Proceedings of the

QR 2023

36th International Workshop on Qualitative Reasoning

Co-located with the

European Conference on Artificial Intelligence (ECAI)

Krakow, Poland – October 1st, 2023

Edited by

Bert Bredeweg

Amsterdam University of Applied Sciences, Faculty of Education, The Netherlands

University of Amsterdam, Faculty of Science, The Netherlands

Johan de Kleer

Palo Alto Research Center, USA

Program Chairs

Bert Bredeweg
Amsterdam University of Applied Sciences, Faculty of Education, The Netherlands
University of Amsterdam, Faculty of Science, The Netherlands

Johan de Kleer
Palo Alto Research Center, USA

Program Committee

- Núria Agell, ESADE – Ramon Llull University, Spain
- Elena Andonova, New Bulgarian University, Bulgaria
- Mehul Bhatt, University of Bremen, Germany
- Gautam Biswas, Vanderbilt University, USA
- Ivan Bratko, University of Ljubljana, Slovenia
- Elizabeth Bradley, University of Colorado, USA
- Maria Chang, IBM, USA
- Tony Cohn, University of Leeds, UK
- Zoe Falomir, University Jaume I, Spain
- Ken Forbus, Northwestern University, USA
- Scott Friedman, Smart Information Flow Technologies, USA
- Joanna Golińska-Pilarek, University of Warsaw, Poland
- Luis Gonzalez-Abril, Universidad de Sevilla, Spain
- Tomoya Horiguchi, Kobe University, Japan
- Kamal Kansou, Centre de recherche Angers-Nantes, France
- Alexandra Kirsch, University of Tübingen, Germany
- Matthew Klenk, Palo Alto Research Center, USA
- Marco Kragten, Amsterdam University of Applied Sciences
- Kate Lockwood, University of St. Thomas, USA
- Andrew Lovett, Northwestern University, USA
- Lleó Museros, University Jaume I, Spain
- Wei Pang, University of Aberdeen, Scotland, UK
- Mónica Sánchez, Universitat Politècnica de Catalunya, Spain
- Carl Schultz, Aarhus University, Denmark
- Qiuang Shen, Aberystwyth University, Wales, UK
- Kazuko Takahashi, Kwansei Gakuin University, Japan
- Louise Travé-Massuyès, LAAS-CNRS, France
- Przemyslaw A. Walega, University of Oxford, UK
- Diedrich Wolter, University of Bamberg, Germany
- Jure žabkar, University of Ljubljana, Slovenia

Preface

Proceedings of the working papers accepted and presented at the 36th International Workshop on Qualitative Reasoning (QR) held on October 1st, 2023, co-located with ECAI in Krakow, Poland.

The QR community is involved with the development and application of qualitative representations to understand the world from incomplete, imprecise, or uncertain data. Our qualitative models span natural systems (e.g., physics, biology, ecology, geology), social systems (e.g., economics, cultural decision-making), cognitive systems (e.g., conceptual learning, spatial reasoning, intelligent tutors, robotics), and more.

The QR community includes researchers in Artificial Intelligence, Engineering, Cognitive Science, Applied Mathematics, and Natural Sciences, commonly seeking to understand, develop, and exploit the ability to reason qualitatively. This broadly includes:

- Developing new formalisms and algorithms for QR.
- Building and evaluating predictive, prescriptive, diagnostic, or explanatory qualitative models in novel domains.
- Characterizing how humans learn and reason qualitatively about the (physical) world with incomplete knowledge.
- Developing novel, formal representations to describe central aspects of our world: time, space, change, uncertainty, causality, and continuity.

The International Workshop on Qualitative Reasoning provides a forum for researchers from multiple perspectives to share research progress toward these goals.

Topics of interest include:

- Qualitative modelling in physical, biological and social sciences, and in engineering.
- Representations and techniques for QR.
- Methods that integrate QR with other forms of knowledge representation, including quantitative methods, machine learning and other formalisms.
- Using QR for diagnosis, design, and monitoring of physical systems.
- Applications of QR, including education, science, and engineering.
- Cognitive models of QR, including the use of existing QR formalisms for cognitive modelling and results from other areas of cognitive science for QR.
- Using QR in understanding language, decision-making, sketches, images, and other kinds of signals and data sources.
- Formalization, axiomatization, and mathematical foundations of QR.

The accepted papers were reviewed by at least two members of the international program committee.

Acknowledgements

We acknowledge the support for organizing the Qualitative Reasoning workshop by the Amsterdam University of Applied Sciences, the University of Amsterdam (The Netherlands), and the Palo Alto Research Center (USA).

We thank the members of the international Program Committee for their valuable work during the reviewing process. We also thank Easychair, which was used to manage paper submissions and the reviewing process.

Contents

Incremental Analogical Learning with Qualitative Representations of Quantity

Will Hancock[1], **Walker Demel**[2], **Kenneth D. Forbus**[2]
Northwestern University

wwhancock@u.northwestern.edu,{walker.demel,forbus}@northwestern.edu

Abstract

Quantities are ubiquitous in our conceptualization of the world, and the ability to learn and reason with them is an important aspect of commonsense reasoning. Existing cognitive models of similarity and generalization often lack sensitivity to quantitative knowledge, and those that are often represent it implicitly, meaning that it is not available for further learning or reasoning. This paper presents an extension to analogical reasoning processes that enables learning from mixed qualitative and quantitative knowledge. This is accomplished by utilizing qualitative representations of quantity, and by leveraging structure mapping to build schemas incrementally, maintaining probability distributions for quantitative knowledge, and then using these distributions to generate predicates that participate in structured generalization. This extension, called **A**nalogical**Q**uantity**E**stimation (AQE) is both incremental and unsupervised, and our results show that AQE performs significantly better than a baseline where quantitative knowledge is not taken into account. In addition, we compare AQE to a standard linear regression estimator, which, despite being batch and supervised, does not perform significantly better than AQE, and in some cases, performs worse.

Introduction

Commonsense knowledge is playing an increasingly important role in the development of AI systems. Many large-scale knowledge bases are emerging that encode general facts about the world using both structured qualitative and quantitative knowledge. Such knowledge is available in large open-domain knowledge bases such as *OpenCyc*, *DBpedia* and *WikiData*.

The ability to learn and generalize from these knowledge sources is therefore useful to any AI agent. Most existing computational models of retrieval and similarity cannot use numerical representations (Forbus et al., 2017; Holyoak and Thagard, 1989; Hummel and Holyoak, 1997), leading to quantitative information being ignored in computation of similarity. There are models in case-based reasoning (Ram and Santamaria, 1997) that use numeric information, but they employ ad hoc similarity metrics that are not psychologically grounded. A major motivation of this work is to generate cognitively plausible symbolic representations of quantity and show that these representations aid in learning.

In this paper, we introduce a novel algorithm, AQE, that improves an existing analogical learner so that it is sensitive to quantity. A similar idea was proposed by Paritosh (2004), which introduced a computational model called *CARVE*. AQE extends CARVE in two ways. First, CARVE's quantity symbolization was external to the analogical learner and needed to be run manually. Second, this symbolization process was batch, meaning that it needed access to an entire dataset before learning, and symbolization needed to be complete before any learning took place. AQE addresses these issues by automatically symbolizing quantities incrementally as new cases are generalized. Additionally, CARVE did not find any regularities in the data it was tested on, whereas our model shows significant improvement over a baseline.

AQE is tested by estimating quantities for two datasets derived from Wikidata; one containing knowledge about countries, and the other knowledge about universities. Wikidata contains vast amounts of knowledge in a wide array of domains, and therefore is a useful resource that contains a wealth of ground facts that can be used for commonsense reasoning (Forbus and Demel, 2022).

We begin by introducing the most relevant related work on systems that used mixed qualitative and quantitative knowledge. Then we present AQE, including the qualitative representation scheme and its incorporation into an existing analogical learner. Finally, we show results for experiments on two Wikidata datasets, ending with conclusions and future work.

Related Work

We give a brief overview of previous computational models that use mixed qualitative and quantitative representations, as well as related models of similarity and retrieval.

Computational Models

There are many examples of representational schemas that combine structured and quantitative knowledge. Hinton's (1979) model of mental imagery combines structured knowledge with numerical properties. Both ACT-R (Anderson, 2009) and SOAR (Laird, 2012) use numerical components in their representations, for example, statistical metadata on recency, frequency, and utility for symbolic structures. There are currently several theoretical frameworks that tightly integrate logic and probability, including Markov Logic Networks (Richardon and Domingos, 2006), while Rosenbloom's (2013) SIGMA cognitive architecture is exploring how to use graphical models to build a complete cognitive architecture, including both symbolic and statistical reasoning.

Many of these models treat quantity implicitly, meaning that it is not available at the level of knowledge. On the other hand, explicit reification is useful because it allows for graceful extension in learning and reasoning, as well as access to the richer semantics of quantity ontologies, such as QP theory (Forbus, 2019).

In addition, these models often require batch learning, which is problematic for cognitive agents because all previous knowledge must be stored. On the other hand, AQE incrementally accumulates distributional knowledge over quantities, meaning that distributions can be updated online as new examples are generalized.

There is converging psychological evidence for structured models of retrieval, similarity, and generalization. One limitation of existing models of analogical processing, e.g., ACME (Holyoak and Thagard, 1989), LISA (Hummel and Holyoak, 1997), ABSURDIST (Goldstone and Rogosky, 2002) is that they do not handle numerical properties adequately. In most of these models, numbers are treated like symbols, so 99 and 100 are as similar/different as 99 and 10000. AQE addresses this issue by automatically symbolizing quantity using the qualitative representations proposed by CARVE, creating new predicates that contribute to similarity in analogical learning.

Background

Next, we overview the analogical learning stack (SME, MAC/FAC, and SAGE) that we are extending, CARVE, a computational model of quantity estimation that we are building on top of, and Wikidata, the source of our data.

Analogical Learning

The generalization mechanism for AQE is built on models inspired by Gentner's structure-mapping theory of analogy and similarity (Gentner, 1983). AQE uses the Structure Mapping Engine (SME; Forbus et. al, 2016) for analogical matching, MAC/FAC for retrieval, and SAGE for analogical generalization. These analogical processes have been used in a wide range of domains, including sketch recognition (Chen et al., 2023), learning to play strategy games (Hancock and Forbus, 2021), and question answering (Crouse et al., 2019), and so we hypothesize that it will be useful for learning with representations of quantity. We summarize each component in turn.

The structure mapping engine (SME) is a domain-general computational model of analogy and similarity, based on Gentner's structure mapping theory. It returns a set of mappings between a base and a target, both structured representations, along with a similarity score for each mapping. Each mapping contains (1) correspondences that map entities and expressions in the base with entities and expressions in the target, (2) a numerical structural evaluation score of the quality of the mapping, and (3) candidate inferences. Candidate inferences are expressions that occur in the base description and not in the target but can be hypothesized to hold in the target.

The MAC/FAC algorithm (Forbus, Gentner, and Law, 1995) is a model of analogical retrieval. MAC/FAC takes as input a probe description (a set of facts) and a set of examples, and returns the example that is most similar to the probe. MAC/FAC stands for many are called, few are chosen. Retrieval is a two-stage procedure. In the MAC stage, each case is represented by a *content vector*. Each dimension in a content vector represents a predicate, and its magnitude corresponds to the number of occurrences of the predicate in that case. The dot product of two content vectors provides a rough estimate of what SME would compute for a similarity score for the corresponding structured representations. This dot product is used as a pre-filter to reduce the number of comparisons made in the FAC stage, which are computationally more expensive. The MAC stage is a map/reduce operation, where a dot product for a content vector of the probe is computed in parallel with the vectors for all items in the case library, with the top three scoring cases passed on to the FAC stage. The FAC stage also is map/reduce but using SME on the probe and the three retrieved cases, keeping the best. The MAC stage provides scalability, since vector dot products are quite fast. The FAC stage provides the sensitivity to structure that human retrieval demonstrates.

The Sequential Analogical Generalization Engine (SAGE; Kandaswamy & Forbus, 2012) is a model of analogical generalization. SAGE learns models of concepts, incrementally, from examples. In SAGE, *generalization pools*, or *gpools*, are used to build up models of concepts. The number of gpools used for learning is determined by the number of concepts in a domain and the learning goals that arise. A gpool is subdivided into clusters of similar examples, or *generalizations*, and outliers that are not similar to any other cases or generalizations. Each generalization can be thought of as a component of a disjunctive model for the concept. In this sense SAGE is like k-means with outliers, except that there is no a priori determination of the number of clusters; the algorithm derives that from the data.

Generalization with SAGE involves assimilating new examples into gpools, and inference involves finding a generalization (or outlier) that is most similar to a probe case. For assimilation, an incoming case is used to retrieve existing outliers and generalizations within a gpool, using MAC/FAC. If the case is sufficiently similar to an existing generalization or outlier, as determined by a fixed assimilation threshold, it is merged with that item and a mapping is returned. Otherwise, a new outlier is created. If merging occurs between the probe and an outlier, then a new generalization is created.

In the case where two items are merged, SAGE uses information computed in a mapping to store metadata about the generalization. Probabilities are updated for aligned facts, reflecting the frequency of that fact within the generalization. For example, facts about international organization membership are included in each country case. After a number of country cases have been assimilated, a generalization will have a lifted facts corresponding to these memberships, and a probability for each fact. For example,
((MemberOfInternationalOrgFn
AllianceofSmallIslandStates) *<?country>): 0.96*

reflects the fact that a country is a member of the Alliance of Small Island States, and has a probability of .96 within the context of one generalization, meaning that 96% of the constituents of that generalization exhibit this attribute.

The <?country> placeholder is a skolem (new unique symbol) that is denoted in knowledge by a non-atomic term (*GenEntFn*). Probabilities for generalizations are updated every time a new example is assimilated. Statements whose probabilities become too low are eventually deleted, based on a fixed probability cutoff threshold.

Quantity Representation in CARVE

AQE builds upon representations of quantity and a computational model, CARVE, developed by Paritosh (2004). CARVE used two distinctions for representation of quantity: *distributional* and *structural* partitions. Distributional partitions map a continuous value to some ordered interval within a probability distribution. More than just the norm, ordered partitions can be defined within the distribution (e.g. small, large) for many quantities, which are construed as a qualitative decomposition of the space. There is psychological evidence that suggests that we can and do accumulate distributions of quantities (Malmi and Samson, 1983; Fried and Holyoak, 1984; Kraus et al, 1993). Distributional partitions are represented by statements of the form

(isa *<?country>* (<?amount> *<?qtype>*))

For example, the USA has a high literacy rate relative to all other countries in the world, represented by:

(isa USA (HighAmountFn LiteracyRate))

Whereas distributional partitions decompose individual quantities, structural partitions highlight how quantities are constrained by what values other quantities in the system take. For instance, GDP tends to increase as a country's population increases, and literacy rates tend to increase with GDP. These constraints represent the underlying mechanisms, or correlations within the domain. *Limit points* decompose values into regions where the underlying correlational story is different (e.g., rich vs poor nations), which induces important and interesting distinctions of quality on the space of quantity.

Wikidata

The AQE algorithm is domain-independent and ontology independent. This work focuses on readily available structured knowledge derived from the Wikidata dataset. Wikidata is a collaborative knowledge graph that serves as a repository of structured data for a wide range of information from many different domains. Like its sibling Wikipedia, Wikidata utilizes the distributed-community model of editors—as of this writing, thousands of editors and bots have made over 1.6 billion edits to over 97 million items. This model allows Wikidata to serve as the downstream aggregate of otherwise independent structured data sources.

Wikidata is organized around items, with each having a unique identifier (QID) and a set of statements about it. Each statement is an RDF triple of <subject property value>. For example, "the United States is a member of the World Health Organization" can be expressed as *<member of (P463), United States (Q30),World Health Organization (Q7817)>* where the terms in italics are the English rendition of the objects whose ids are in parentheses. In QR terminology, items are entities and values are quantity values. In some cases the quantity type is obvious (e.g. Area, Color), while in others it is more opaque (Statistical Population). Any property can specify constraints on its value. Certain properties specify that their values must be a string, number, date, URL, media file, or another Wikidata entity. Other properties, like capital (P36) enforce no more than one value since most states have only one capital. Since Wikidata consists of RDF triples, it can be queried via a SPARQL endpoint (query.wikidata.org). In Wikidata, predicates like point in time (P585) can be used to qualify statements like population (P1082), for which there may be several different assertions that hold in different years. In the case where a country's capital (P36) may have changed, values can be associated with a start time (P580) and end time (P582).

Analogical Quantity Estimation

Recall that SAGE computes progressive structural overlap over incoming cases, resulting in a set of disjunctive generalizations for a concept. For example, in this work a generalization might denote the set of wealthy European nations. In this sense, generalizations can be viewed as structural partitions that describe some latent concept (i.e. rich countries). The goal of structural partitioning is to assign cases to generalizations that correspond to useful distinctions (for instance, groups of developed and underdeveloped nations). Learning for AQE consists of two steps. In the first, quantitative facts are symbolized; that is, continuous quantities are mapped to qualitative distributional partitions, and the resulting new facts are added to the original case. In the second step, this augmented case is added to a separate gpool, which learns structural partitions in the data. We outline each of these steps next.

Distributional Partitioning

The first step for AQE is to encode numeric facts in incoming cases. Many quantity estimators, e.g. regression, assume that

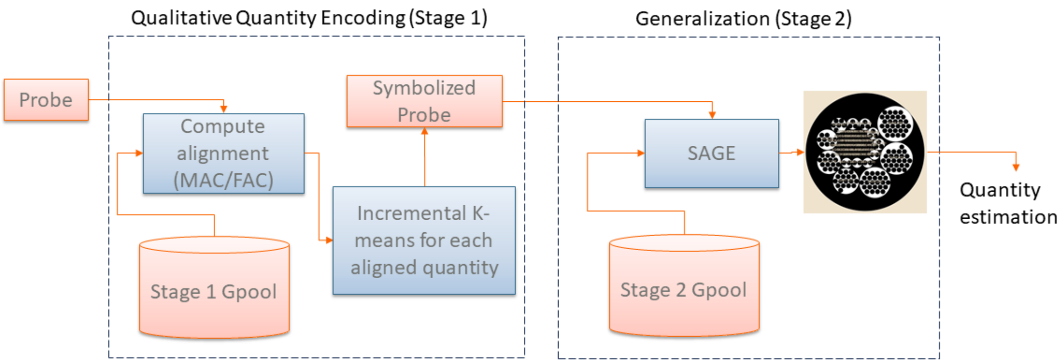


Figure 1: Overview of the AQE encoding, generalization, and inference processes. Stage 1 encodes qualitative facts for each numeric quantity found in a case. This is achieved by accumulating statistical information about each quantity type in the stage 1 gpool. For example, country GDP will have an associated k means. This distributional knowledge is used to encode quantitative knowledge in incoming cases. Stage 2 generalizes the newly symbolized cases, resulting in a set of generalizations (structural partitions), each accumulating statistical information about constituent cases. This model is then used for inference, to estimate quantities for new cases.

incoming data is unstructured, and that attributes are already aligned. Since Wikidata combines structured and unstructured knowledge, this poses an additional challenge to learning. That is, entities and attributes must be aligned before learning can take place. For learners like regression, this is handled outside of the learning mechanism, often manually. One benefit of AQE is that this procedure is handled automatically by computing analogical mappings and is tightly integrated into the learning mechanism. Thus, the first step in symbolizing quantities is to compute attribute alignments (Figure 1, qualitative quantity encoding). Once this is complete, distributions for aligned quantities are used to map continuous quantities to distributional partitions, which we describe next.

First, an incoming quantity must be mapped to a set of previously seen quantities. For example, to symbolize the literacy rate of the USA, which is 99.4 as of 2022, then this quantity should be compared with the distribution for literacy rates of all previously seen countries. This is handled with SAGE by maintaining a gpool that has an assimilation threshold of zero. Recall that the assimilation threshold sets the minimum requirement for two cases to be considered similar. An assimilation threshold of zero means that a gpool will have a single generalization that contains all assimilated cases. While not useful for learning (because it makes no distinctions), this model is useful because it provides a global schema. This schema provides useful metadata about facts in the dataset. For one, the relative frequency of each aligned fact is stored, (e.g. 3.4% of countries border Cameroon). Second, it associates each quantity type with information about the values that that quantity type has taken. The goal is to separate each quantity type into a predetermined number of qualitative partitions. This is achieved with an online k-means algorithm. Given an unseen quantity, it is assigned to one of the K distributions by minimizing the Euclidian distance between the quantity and the norms of each distribution. If less than k quantities have been seen, a new distribution is created, and the new quantity is set as the mean.

For this paper, the number of distributional partitions is set at five, as we have found that this is a good balance between expressiveness and relevance. Too expressive (too many partitions) and all quantities tend towards dissimilarity. Too few distinctions, and all quantities tend towards similarity. Using five partitions results in a quantity space that can be interpreted as (very small, small, medium, large, very large). The number of distributions K can be set at the level of a gpool by by asserting a fact

(kMeansForQuantityAnalysis <?gpool> <?K>)

in the knowledge base.

The next step is to generate qualitative facts based on the assignment of a quantity to one of the K distributional partitions. If fewer than K quantities have been seen, then no fact is generated. Otherwise, a new fact is created, e.g.

(isa USA (HighAmountFn GDP))

and added to the existing case in place of the prior quantitative fact.

Next, the associated distribution is updated to reflect the new quantity. SAGE stores with each distribution a set of statistics: the cardinality, minimum, maximum, mean, and sum of squared error of the constituent quantities. This metadata is used later on for inference, which is detailed below.

Structural Partitioning

Once quantities in a case have been symbolized, the case is given as input to a second SAGE gpool, this time with a non-zero assimilation threshold. The assimilation threshold used for these experiments is .6, which is a standard value that has been successful for learning across many domains. In this step, both existing qualitative as well as the new symbolized

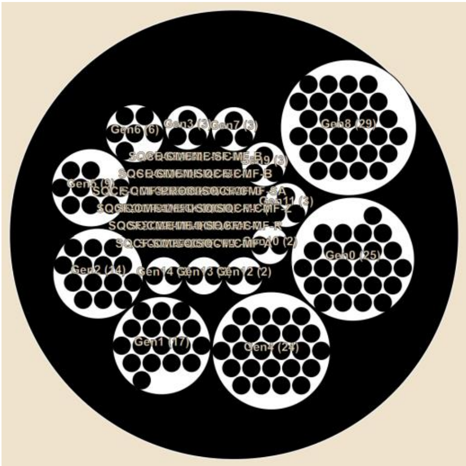


Figure 2: A SAGE Gpool consisting of 15 generalizations (white circles), each containing individual cases (black dots). Generalizations represent groups of similar cases (structural partitions of the dataset).

qualitative facts are taken into account by analogical matching. Figure 2 shows an example gpool with white circles designating generalizations containing similar countries. Each generalization reflects some structural partition in the source dataset. Structural partitions are a reflection the system's understanding of the correlational structure of a set of cases.

The gpool for the second stage accumulates the same statistical knowledge about quantity distributions as the first stage, over quantities of cases within the same generalization. In the next section, we describe how, along with analogy, these statistics contribute to inference in AQE by allowing quantity estimation for quantities in held out cases.

Quantity Estimation

For inference, the idea is to estimate an unseen quantity for some case. First, knowledge about the target quantity is removed from the case. Then, quantity estimation proceeds by first symbolizing all quantities, using the gpool from stage 1. First SAGE retrieves a mapping between the probe case and a generalization. Since the stage 1 gpool has an assimilation threshold of zero, all cases are similar to the single generalization, and a mapping is guaranteed. This mapping aligns quantities in the probe to previously seen quantities from training. For each aligned quantity, the k-means algorithm assigns it to one of K distributions. This assignment is used to generate a qualitative fact, as detailed previously. This fact is added to the probe case, and once this has been performed for all aligned quantities, inference proceeds to stage 2.

Next, SAGE retrieves a mapping from the augmented case to an object in the stage 2 (structural) gpool. If a mapping to a generalization is found, the mean of the target quantity type for that generalization is used as the estimate. If no mapping is found, then the estimate is the marginal average for that quantity type across all cases in the gpool. If the case maps to an outlier, then the quantity from the outlier is used for prediction, or the marginal over all cases in the gpool if the outlier does not have a quantity value for that quantity type.

Evaluation

This paper evaluates AQE on a set of cases that were extracted from Wikidata. Next, we describe this extraction procedure and then discuss how the resulting dataset is used to evaluate AQE.

Case Construction from Wikidata

For learning in AQE, we translate from Wikidata to the open-license knowledge base NextKB[1], which is used for AQE experiments. Data from Wikidata was pulled using the public SPARQL endpoint at query.wikidata.org. For the country dataset, ten quantitative attributes were queried for the year 2022 (population, GDP, GDP per capita, median income, democracy index, life expectancy, fertility rate, area, literacy rate, and human development index) and 6 qualitative attributes (continent, bordering countries, bordering bodies of water, language(s) spoken, international organization memberships, and currency). Overall, 197 cases were generated, having an average of 40 facts each.

For the set of university cases, qualitative attributes are (instanceOf; P31), organizational memberships (P463), and Carnegie Membership Classification (P2643). For quantities, students count (P2196), total assets (P2403), employees (P1128), admission rate (P5822), endowment (P6589), and admission yield rate (P10263) were used. For each university, the latest available quantity for each quantity type was used. All quantities are from 2019 and later, up to the year 2023. This dataset was extracted on July 31, 2023. Cases were generated for universities that were founded prior to 1860, which resulted in 231 university examples. Those that did not have any associated quantitative knowledge were removed, resulting in 194 cases.

Attributes for cases were chosen based on the hypothesis that there is a rich underlying correlational structure that can be learned. These facts were translated into OpenCyc's ontology language for use within NextKB. For some predicates there was a natural correspondence, such as nominal GDP in Wikidata and grossDomesticProduct in OpenCyc. Other predicates were missing from OpenCyc and thus hand ontologized, e.g. human development index as the predicate hdiOfCountry and percentage of applicants admitted as percentApplicantsAdmitted.

For example,

<United States (Q30), population (P1082), 331,449,281>

[1] https://www.qrg.northwestern.edu/nextkb/index.html contains downloadable files in various formats, browsers, and reasoning systems. It uses Creative Commons Attribution 4.0 licensing, compatible with OpenCyc, FrameNet, and other resources.

results in

(populationOfRegion
UnitedStatesOfAmerica
((UnitOfCountFn Person) 331449281)).

Experiment

For the experiments, AQE is evaluated against two baselines: (1): analogical quantity estimation without quantity symbolization and (2): against a standard linear regression estimator. Our hypothesis is that AQE will outperform analogical quantity estimation without qualitative representations of quantity. Additionally, results from a standard linear regression estimator are included. Recall that AQE is both incremental and unsupervised; incremental learners are well known to lack statistical guarantees of their batch counterparts due in part to the stochastic effects of initialization. The results for linear regression are included as a means of comparing learning performance of AQE vs a technique with better learning guarantees.

To run the experiments, standard cross validation is used to partition each dataset into ten folds, each consisting of a train and test set. For countries, there are 197 cases, and 194 for universities, resulting in a test set for each of the ten folds consisting of approximately 20 cases for both datasets. The folds are generated by first randomizing the cases, and then generating ten partitions based on the ordering from this randomization. AQE and the incremental baseline are implemented in Allegro Common Lisp 10.1. A seed for the random state in Allegro Common Lisp is set to 55 for the baseline and AQE conditions, as well as generation of the cross validation set. The linear regression estimator is run using the implementation in Python's ScikitLearn module, using default parameters, also using the same cross validation set that was generated in Allegro Common Lisp. Learning regression models requires vectorizing structured knowledge from each dataset. This is accomplished by manually creating a mapping, where each quantity type is considered a feature, and each unique qualitative attribute (e.g. currency, organization membership) is represented by a one-hot vector. Missing quantities are imputed using Python's impute function from the SciPy module. This results in 883 features across the 197 country cases for universities, and 181 features across 194 cases for the university dataset.

For the country dataset, each condition is tested on four different quantity types: life expectancy (**LE**), human development index (**HDI**), democracy index (**DI**), and nominal GDP (**GDPnom**). For universities, average yield percent (**AYP**), percent applicants admitted (**PAA**), and number of emplyees (**NOE**) are tested.

Results

For countries, our results show that AQE performs significantly better ($p < .05$) than the baseline for every quantity that was tested. Additionally, the regression condition fails to perform significantly better than AQE for any quantity ($p > .05$), and AQE outperforms the regression estimator in one instance (nominal GDP).

	LE	**HDI**	**DI**	**GDPnom**
baseline	60.84	.023	5.15	5.54e8
AQE	20.8	.0088	2.63	**3.79e8**
regression	**19.88**	**.0062**	**1.95**	4.9e8

Table 1: mean squared error across 10 folds for four quantity types (life expectancy, human development index, democracy index, and nominal GDP) across three experimental conditions.

For the university dataset, all experiments were run using the same parameters that were used for the country cases. We tested AQE on average yield percent **(AYP)** (the percentage of students that enroll given acceptance), percent applicants accepted (**PAA**), number of employees (**NOE**), and endowment value (**EV**). For this experiment, AQE outperformed the incremental baseline as well as regression for all quantities tested. Admission yield percent and percent applicants admitted showed significant improvement over the incremental baseline ($p < .05$). The regression condition suffered due to overfitting on certain folds, resulting in large out-of-distribution predictions.

	AYP	**PAA**	**NOE**	**EV**
baseline	.017	.059	1.31e7	3.9e19
AQE	**.011**	**.034**	**8.46e6**	**2e19**
regression	39443	7.5	9.43e9	2.95e19

Table 2: mean squared error across 10 folds for four quantity types (admission yield percent, percent applicants admitted, number of employees, and endowment value).

Explainability

The qualitative representations of quantity used in AQE also result in explainable models, because they are compatible with natural language. The final learned model (stage 2) represents a disjunction over structural partitions of the data. Figure 3 shows a subset of facts from one of these learned structural partitions. In SAGE terms, this corresponds to a

Fact	Prob
((MemberOfInternationalOrgFn AfricanDevelopmentBank) *<?country>*)	1.0
((MemberOfInternationalOrgFn AfricanUnion) *<?country>*)	1.0
((MemberOfInternationalOrgFn InternationalBankforReconstruction-andDevelopment) *<?country>*)	1.0
(isa *<?country>* (CountryTypeFn (VeryLowAmountFn grossDomesticProduct-Nominal)))	.862

Figure 3: Example facts from a single generalization (structural partition) with 29 member cases

generalization, which stores probabilities of individual facts. The depicted generalization in Figure 3 shows that every constituent is a member of the African Development Bank. Furthermore, inspection of the model reveals that 86.2% of the participants have a very low GDP. The nature of these representations means that they are inspectable. In the next section, we discuss possible extensions that use these learned models for further learning.

Discussion and Future Work

This paper introduces AQE, an extension to SAGE that enables learning with quantitative knowledge by automatically symbolizing those quantities into predicate statements that denote distributional partitions. Furthermore, the experiment shows that these representations can assist in estimating quantity by using analogy to learn salient structural partitions of the underlying data in two datasets. Specifically, using qualitative representations significantly improves over a baseline in which these representations are not included. AQE is also compared against a linear regression estimator, which, despite being supervised and batch, does not perform significantly better than AQE in the first experiment, and in the second experiment, performs worse for every quantity.

As Paritosh (2004) points out, relative magnitudes such as *large* are context dependent and thus elude global definition. A person might be tall with respect to the general populace, but short compared to the set of professional basketball players. These ecological constraints surrounding judgements of this kind mean that the ability to quickly estimate from a few examples is especially useful, because new contexts are encountered frequently. The relative data efficiency of AQE is a boon compared to the data-hungry nature of many current statistical learners. This raises the possibility of applying AQE to new domains, such as concept acquisition in situated learning (e.g. learning *near* and *far* by symbolizing quantity).

Furthermore, declarative representations like the ones used in this paper allow for extensibility, in that they can be used as a foundation for other kinds of reasoning. For example, further refinement is possible by connecting the learned representations with the semantics of a more expressive qualitative reasoning framework, e.g. Qualitative Process theory (Forbus, 2019). This opens up the possibility of refining these models by incorporating knowledge from other sources, e.g. language. One way that this might be accomplished is by leveraging causal relationships parsed from language to highlight what is salient for a given estimation task. The idea is that this causal knowledge could improve model accuracy by filtering out noise introduced by non-salient attributes, e.g., extending a quantitative anchoring framework such as KNACK (Paritosh and Klenk, 2006).

Acknowledgements

This research was sponsored by the US Air Force Office of Scientific Research under award number FA95550-20-1-0091.

References

Anderson, J. R. (2009). How Can the Human Mind Occur in the Physical Universe? Oxford University Press.

Chen, K., Forbus, K., Srinivasan, B. V., & Chhaya, N. (2023). Sketch Recognition via Part-based Hierarchical Analogical Learning.

Crouse, M., McFate, C., & Forbus, K. (2018). Learning From Unannotated QA Pairs to Analogically Disambiguate and Answer Questions. Proceedings of the AAAI Conference on Artificial Intelligence, 32(1).

Demel, W., Hancock, W., & Forbus, K. (under review). KNACK v2: Using Analogical Generalization over Qualitative Representations for Quantitative Estimation. *Under review.*

Forbus, K. D. (2019). Qualitative Representations: How People Reason and Learn about the Continuous World. MIT Press.

Forbus, K. D., & Demel, W. (2022). Integrating QR Quantity Representations with the Semantic Web: A Progress Report.

Forbus, K. D., Ferguson, R. W., Lovett, A., & Gentner, D. (2017). Extending SME to Handle Large-Scale Cognitive Modeling. Cognitive Science, 41(5), 1152–1201.

Forbus, K. D., Gentner, D., & Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. Cognitive Science, 19(2), 141–205.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. Cognitive Science, 7(2), 155–170.

Goldstone, R. L., & Rogosky, B. J. (2002). Using relations within conceptual systems to translate across conceptual systems. Cognition, 84(3), 295–320.

Hancock, W., & Forbus, K. D. (2021). Qualitative Spatiotemporal Representations of Episodic Memory for Strategic Reasoning.

Hinton, G. (1979). Some Demonstrations of the Effects of Structural Descriptions in Mental Imagery.

Holyoak, K. J., & Thagard, P. (1989). Analogical Mapping by Constraint Satisfaction. Cognitive Science, 13(3), 295–355.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. Psychological Review, 104(3), 427–466.

Kandaswamy, S. and Forbus, K. (2012). Modeling Learning of Relational Abstractions via Structural Alignment. *Proceedings of the 34th Annual Conference of the Cognitive Science Society (CogSci).* Sapporo, Japan.

Kraus, S., Ryan, C. S., Judd, C. M., Hastie, R., & Park, B. (1993). Use of mental frequency distributions to represent variability among members of social categories. Social Cognition, 11(1), 22–43.

Laird, J. E. (2012). The Soar Cognitive Architecture. MIT Press.

Malmi, R. A., & Samson, D. J. (1983). Intuitive averaging of categorized numerical stimuli. Journal of Verbal Learning and Verbal Behavior, 22(5), 547–559.

Paritosh, P. K. (2004). Symbolizing Quantity. Proceedings of the Annual Meeting of the Cognitive Science Society, 26(26).

Paritosh, P. K., & Klenk, M. E. (2006). Cognitive Processes in Quantitative Estimation: Analogical Anchors and Causal Adjustment. The Proceedings of the 28th Annual Conference of the Cognitive Science Society, Vancouver.

Ram, A., & Santamaría, J. C. (1997). Continuous Case-Based Reasoning. Artificial Intelligence, 90(1–2), 25–77.

Richardson, M., & Domingos, P. (2006). Markov logic networks. Machine Learning, 62, 107–136.

Rosenbloom, P. (2013). The Sigma cognitive architecture and system.

Qualitative Models to learn about Star Properties, Star States, and the Balance between Fusion and Gravity

Bert Bredeweg[1,2], Marco Kragten[1], Joanna Holt[1,3], Dennis Vaendel[3], Joris Hanse[3], Steven Bloemen[4]
[1]Faculty of Education, Amsterdam University of Applied Sciences, Amsterdam, The Netherlands
[2]Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
[3]University of Amsterdam, Faculty of Science, API/NOVA, Amsterdam, Netherlands
[4]Radboud University Nijmegen, Department of Astrophysics/IMAPP, Nijmegen, Netherlands
b.bredeweg@hva.nl

Abstract

This paper presents three qualitative models that were developed for the Stargazing Live! program. This program consists of a mobile planetarium that aims to inspire and motivate learners using real telescope data during the experience. To further consolidate the learning experience three lessons are available that teachers can use as follow up activities with their learners. The lessons implement a pedagogical approach that focuses on learning by creating qualitative models with the aim to have learners learn subject specific concepts as well as generic systems thinking skills. The three lessons form an ordered set with increasing complexity and were developed in close collaboration with domain experts.

1 Introduction

Star formation, stellar properties and the underlying physical laws are fundamental topics in pre-university physics education. However, learning about stars can be challenging for learners, due to a variety of pre-instructional conceptions and learning difficulties. For example, learners often do not know that nuclear fusion provides stars with their energy, allowing them to generate light [1,3]. In addition, they have an incomplete understanding of how stars are formed. When asked how stars differ from each other, learners often mention properties such as size or composition, but less often luminosity, temperature, or lifespan. For example, in Bailey and colleagues' study [2], only 21 of 381 learners named mass as a property that distinguishes stars. Previous research shows that traditional instruction in astrophysics courses is not always sufficiently effective and that there is a need for interventions that stimulate conceptual understanding [3].

The Stargazing Live! project [11,12] uses a mobile planetarium to bring semi-live real scientific astronomy data into the classroom. Planetariums have played a role in the learning of astronomical concepts since their inception [4]. They can provide a unique and enriching learning experience [14] and spark learners interest and excitement for astronomy [16, 13] and help improving retention [19].

Key requirements for an effective learning experience in a planetarium is that viewers are allowed and encouraged to ask questions, participate in simulations, and engage in hands-on activities to deepen their understanding of the concepts [16, 17, 13]. The combination of planetarium and traditional classroom lessons can provide a well-rounded education experience that complements and reinforces each other [14, 15].

To address these issues, the Stargazing Live! program comprises two parts. First, learners are introduced to the idea of the changing universe and associated astronomy concepts during a live and interactive planetarium experience. Shortly thereafter, learners further develop and consolidate their knowledge with lesson activities during which they create and simulate cause-and-effect models using computer-supported modelling software. By constructing a model of a system, learners develop a deeper understanding of its underlying principles and relationships between components. This process helps to build and refine their conceptual model, providing a clearer and more comprehensive understanding of the system [8, 9]. Moreover, constructing a model requires active engagement, as learners think deeply about the information and make connections to their prior knowledge. This form of active learning, where learners are actively involved in the learning process, has been shown to be more effective than passive forms of learning [18].

Three qualitative models were created to serve as a basis for the three *learning by modelling* lessons that the Stargazing Live! program developed. The lessons form an ordered set with increasing complexity. The first lesson, *star properties*, focuses on learners identifying key quantities that characterize stars and establishing the causal dependencies between them. The second lesson, *star states*, follows on from the star properties assignment by adding ranges of qualitative values (represented in quantity spaces) to six key quantities. During this assignment learners learn how stars can be classified according to mass and how that relates to characteristic values for other quantities. The third lesson, *fusion-gravity balance*, focusses on the birth of stars and how a balance emerges between the gravitational force (inwards) and the nuclear fusion force (outwards).

The organization of this paper is as follows. Section 2 describes the planetarium experience. Section 3 introduces the DynaLearn software that was used to create the models for the lessons. Section 4, 5 and 6 each describe one of the three models. Section 7 concludes the paper.

2 Planetarium experience

The planetarium experience has been developed by NOVA (Netherlands Research School for Astronomy) using a Mobile Planetarium (Fig. 1). The semi-live real data are taken from the small optical telescopes MeerLICHT (www.meerlicht.org) and BlackGEM (www.blackgem.org), both operated by Radboud University in the Netherlands. MeerLICHT is stationed in South Africa and performs optical follow-up for the MeerKAT radio telescope. The BlackGEM array is in La Silla, Chile and currently comprises three telescopes. Data from the telescopes are uploaded each night, processed automatically, and made available for use within 20 minutes. To run the lessons the mobile planetarium uses customized scripts in the Digistar 6 software.

Figure 1. The planetarium experience.

The topic of the Stargazing Live! program is 'the changing universe' and discusses a range of transient phenomena in the night sky including (near Earth) asteroids, variable stars, (super)novae and gravitational wave events, such as kilonovae. Each topic is introduced with a discussion around a data set from the telescopes projected onto the correct region of the sky in the planetarium software. Learners are asked to identify changing features in the images and think about what they might be seeing. The various physical processes at work are then explained using custom-made 3D-visualisations and animations. Key curriculum topics for pre-university level astrophysics are also included such as an explanation of how Wien's law connects stellar surface temperature to the observed colour of an object and how the luminosity of a star is related to other measurable parameters.

3 DynaLearn – Learning by representing

The modelling lesson activities within the Stargazing Live! program use the DynaLearn software (https://dynalearn.eu) [6]. This software provides a qualitative vocabulary to represent conceptual models [10]. No quantitative information is used. Instead, logic-based algorithms are used to generate simulations [5]. Models built in DynaLearn can be represented at multiple levels of complexity [6]. Higher levels use a richer vocabulary to express the system and its behaviour. At each level, the software has scaffolds to support learners during their knowledge creating effort. The *norm-based feedback* pinpoints errors made by learners (solving these remains a task of the learner). The *scenario advisor* inspects the status of the model before starting a simulation and automatically highlights missing initial settings as well inconsistent settings. The *progress bar* shows how many ingredients have already been created and how many still need to be created. The working of the software is partly explained in the workbook which guides learners through the assignments, but it is also provided from within the software [7].

4 Star properties (level 2)

Lesson activities were developed to extend the planetarium experience, focusing on key concepts in the Dutch secondary school physics curriculum. A specific request was to focus on conceptual understanding of star formation and star properties and the associated laws (e.g., Wien's law and the Stefan-Boltzmann law).

The *star properties* model is shown in Fig 2. The model is created at level 2 of the software, which is relatively simple for learners in pre-university education. The complexity arises from the number of ingredients that need to be created and connected (26 modelling steps) combined with running various intermediate simulations with various initial values.

Entities are used to represent the objects (or parts) that together form the system. This model comprises three entities, Stars (the overarching object), the inner Core and the outer Surface. Two *configurations* specify that Stars have a Core and Stars have a Surface. *Quantities* represent the dynamic and measurable properties that characterize the stars and their behaviour. Eleven quantities are defined, such as Mass, Gravity, Fusion-energy, etc. *Causal dependencies* specify how the change of one quantity influences the change of another quantity. They can be positive, e.g., more Mass results in more Gravity, or negative, e.g., higher Fusion-energy results in a shorter Lifespan.

Initial settings are required to run a simulation. Mass is the quantity at the beginning of the causal chain and thus the only quantity for which an initial change must be specified. When Mass is set to change, the simulation shows how the remaining quantities change (green arrows in Fig. 3). As can be seen in Fig. 3, when Mass increases, all intermediate quantities also increase, and at the end of the causal chain, Radius and Luminosity also increase while Wavelength and Lifespan decrease.

A workbook is used to guide leaners during the lesson. The workbook presents the lesson in 5 steps, notably (a) Entity stars with two quantities (which focusses on Mass ad Gravity), (b) Properties of the core (which focusses on Pressure, Temperature and Fusion-energy, and how these are causally related as well as related to the quantities created in the first step), (c) Properties of the surface (which focusses on Temperature (of the Surface), Wavelength and Luminosity and how these are causally related as well as related to the quantities created before), (e) What else do we know? (which challenges learners to find and add the still missing quantities (namely Gas pressure, Radius & Lifespan) and their cause-and-effect relations. After each step. Learners are asked to run simulations and process the results (e.g., by answering questions).

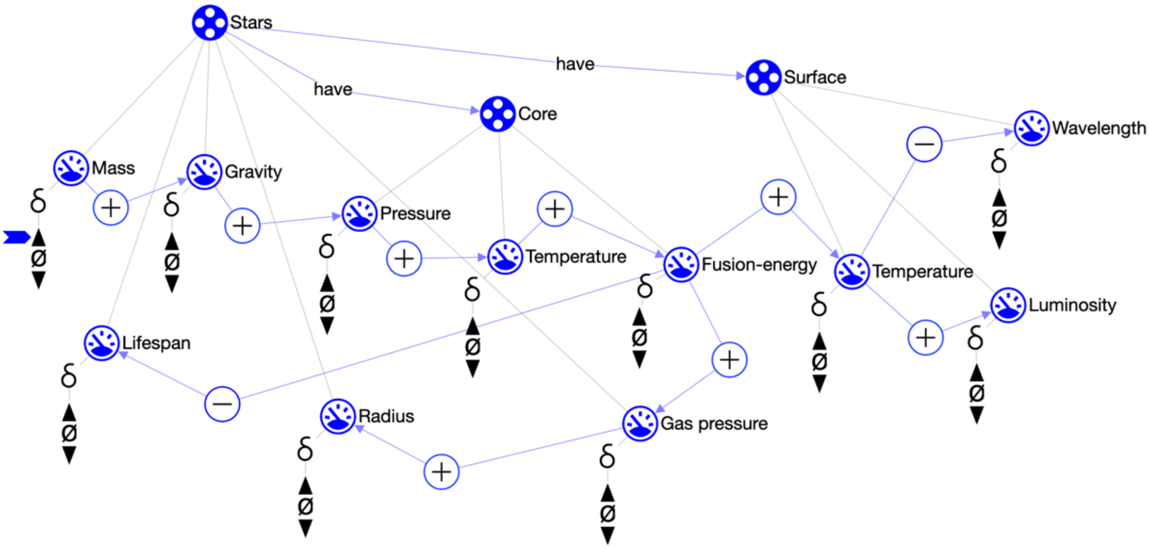


Figure 2. Star properties model with three entities (Stars, Core & Surface), two configurations (2x have), eleven quantities (Mass, Gravity, Pressure, Temperature (of the Core), Fusion-energy, Temperature (of the Surface), Wavelength, Luminosity, Gas pressure, Radius & Lifespan), and ten causal dependencies (2 negative & 8 positive). Mass is set to initially increase (blue arrow).

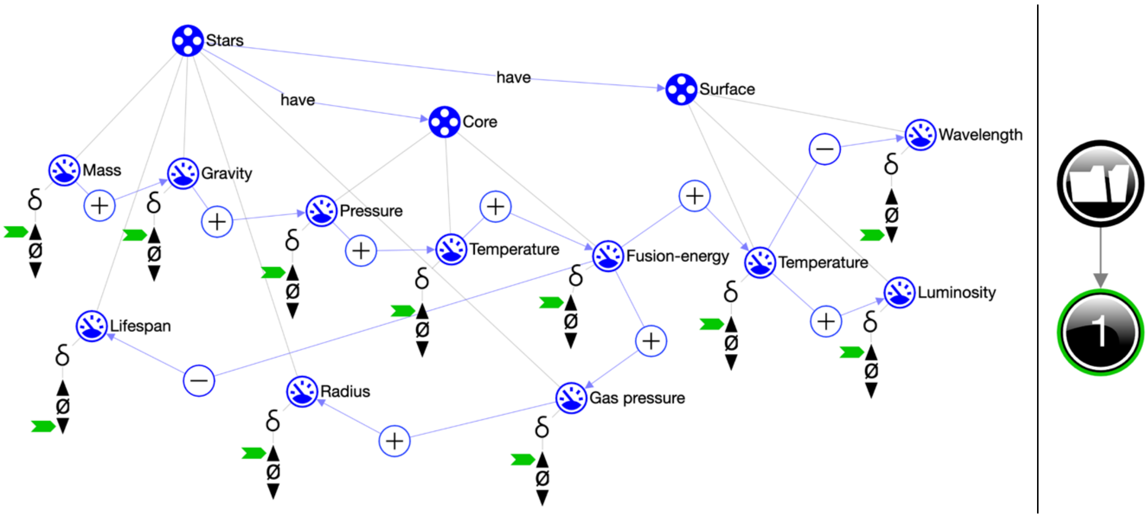


Figure 3. Simulation result for the star properties model shown in Fig. 2. Each quantity has a ∂ which can be decreasing (arrow down), steady (Ø), or increasing (arrow up). Starting with Mass increasing, the simulation shows how other quantities change depending on their proportional relationship with the preceding quantity.

5 Star states (level 3)

The *star states* model (Fig. 4) is created at software level 3. New vocabulary at this level includes *quantity space* (a set of alternating point and interval values that the quantity can take on), *correspondence* (for representing co-occurring values among values from different quantity spaces), and *exogenous quantity behavior* (setting a quantity to keep decreasing, increasing, behave random, etc.) [5]. Correspondences can be directed (only when the source is known, the target can be calculated) or undirected (if one is known, the other can be calculated), and regular (the highest value of one quantity corresponds to the highest value of the other quantity, etc.) or inverse (the highest value of one quantity corresponds to the lowest value of the other quantity, and vice versa).

The *star states* model augments six key quantities from the star properties model with a quantity space, notably Mass, Temperature (of the Surface), Wavelength, Luminosity, Radius & Lifespan. However, to optimally fit the curriculum requirements Wavelength has been replaced by Peak wavelength and Peak colour. Each quantity space holds five values (three intervals separated by two points), and specific values correspond to quantities across the model. For instance, stars with Mass in the red dwarf region (less than 0.5 times the mass of the sun), have a (Surface) Temperature of less than 4000 K, a Lifespan of more than 10^11 years, a Peak wavelength of more than 720 nm, etc.

Learners build the quantity space for each of the key quantities and specify how these values correspond across the model. The lesson is organised as follows. Learners start by creating the quantity space for Mass, run the simulation and discover that they need to apply an exogenous increase to the mass to have the simulation progress through the quantity space fully. Step 2 focusses on the quantity space for Lifespan, and that it inversely corresponds the quantity space of Mass (more Mass corresponds to shorter Lifespan, etc.). Step 3 focusses on Surface Temperature. Step 4 focusses on Peak wavelength and Peak colour simultaneously. Finally, step 5 focusses on Luminosity and step 6 on Radius.

To support learners in determining the values of the quantity spaces the workbook provides short descriptions of each phenomenon. Effectively, all the terms are mentioned in the workbook, but it still requires an effort on behalf of the learners. Specifically, deciding upon the correct terms, their order, and which value is lowest and which value is highest (bottom and top of the quantity space, respectively). Notice that, the norm-based support [7] helps the learners with this challenge. Once a quantity space is in place the next task for learners is to place the correct correspondence, both deciding upon which quantity spaces (of which quantities) to relate and whether the correspondence is regular or inversed.

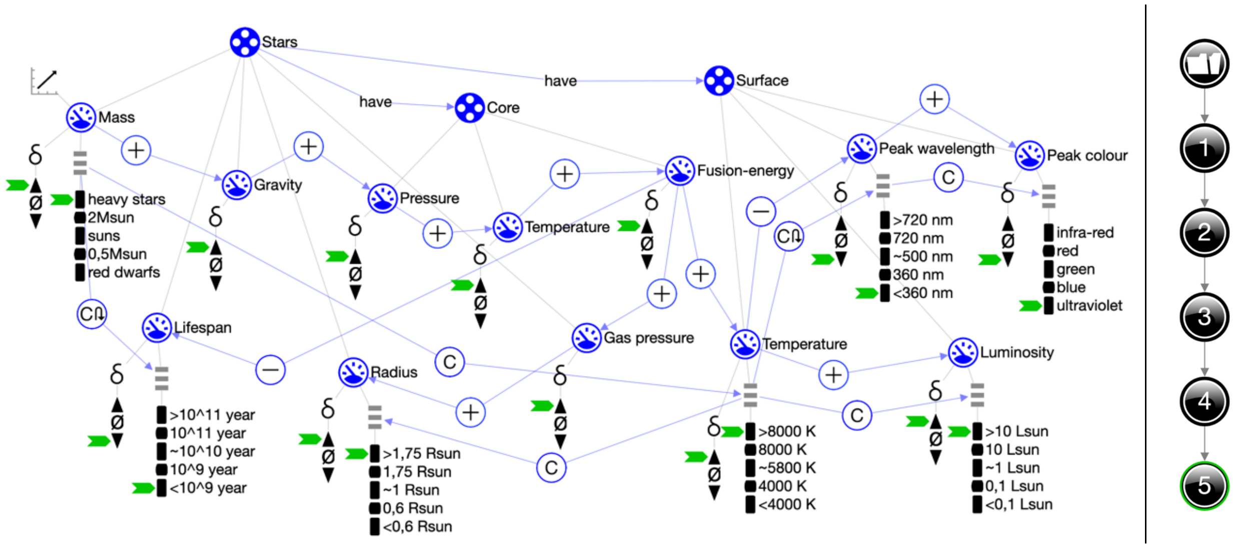


Figure 4. Part of the simulation results for the *star states* model. The simulation started with *Mass=<red dwarfs, +>* (not shown). Following this setting the six key quantities get their initial value via correspondences (C), notably, *Lifespan* started at *<10^9 year, Radius* at *<0.6 Rsun, Temperature* at *<4000 K, Luminosity* at *<0.1 Lsun, Peak wavelength* at *<360 nm,* and *Peak colour* at *infra-red*. The derivatives are calculated using the causal dependencies (–, +). The state-graph (RHS) shows that the simulation progressed through 5 states. State 5 is shown (LHS).

6 Fusion-gravity balance (level 4)

The goal of this *fusion-gravity balance* model is to represent the process of star formation and the consequential fusion-gravity balance that emerges. This model is therefore created at level 4 of the DynaLearn software (Fig. 5). This level introduces *influence* (I+/I–) and *proportionality* (P+/P–) [5,10] to distinguish between processes (I) (initial causes) and the propagation (P) of these through the system. Positive and negative *feedback loops* and *in/equality* (< ≤ = ≥ >) to represent the relative impact of competing processes.

The model starts by distinguishing three entities and their associated quantities: Nebula (Mass & Accretion), Star (Mass, Gravity, Density & Fusion) and Protoplanetary disk (Mass). The model assumes a certain amount of Mass being present in the Nebula <+, ?>, while other quantities are zero <0, ?> (Masses of Star and Protoplanetary disk, and Fusion) or unknown <?, ?> (Accretion, Gravity, and Density). Simulating the model delivers 5 states. Each state representing a unique qualitative behaviour of the system. Table 1 shows the details with for each quantity, in each state, specifying its value and direction of change, represented as a tuple <v, ∂>.

How are these results generated? It starts with the Accretion process, which corresponds (C) and is proportional (P+) to the Nebula's Mass (hence Accretion=<+, –>). Accretion negatively influences (I–) this Mass of the Nebula (hence Mass=<+, –>) and positive influences (I+) the Mass of the Star and the Protoplanetary disk (both <0, +>, see Table 1, state 1). Note that, as soon as Accretion becomes active, it is decreasing because Mass (of the Nebula) is decreasing.

The Gravity of the Star corresponds (C) and is proportional (P+) to the Mass of the Star (Gravity=<0, +>). The Gravity positively influences (I+) the Density, but being zero, has no effect yet in the initial state (state 1). Therefore, Density remains steady, and Gravity in balance with the (not yet active) Fusion (Gravity=Fusion). Note that, to keep the model simple, we choose to not define a quantity space for Density.

State 1 terminates into state 2 in which the Star accumulates Mass (Mass=<+, +>) and consequently the gravitation becomes active (Gravity=<+, +>). Now Density starts increasing and Fusion is about to start (Fusion=<0, +>), but momentarily not yet, therefore Gravity>Fusion.

State 2 progresses into state 3 in which the Fusion becomes active (Fusion=<+, +>), however Gravity still has a stronger impact, hence Gravity>Fusion. State 3 changes into state 4 in which all the Mass from the Nebula has been consumed (Mass=<0, 0>). The Accretion stops (Accretion=<0, 0>) and the Mass of the Star and the Protoplanetary disk stabilise (hence, both <+, 0>). However, Gravity remains active (Gravity=<+, 0>), still outperforms Fusion (Gravity>Fusion), and therefore Density keeps increasing. In state 5 the Fusion catches up with the Gravity and the processes balance (Gravity=Fusion) and the Density stabilises. Fig. 6 shows the simulation results for this final state.

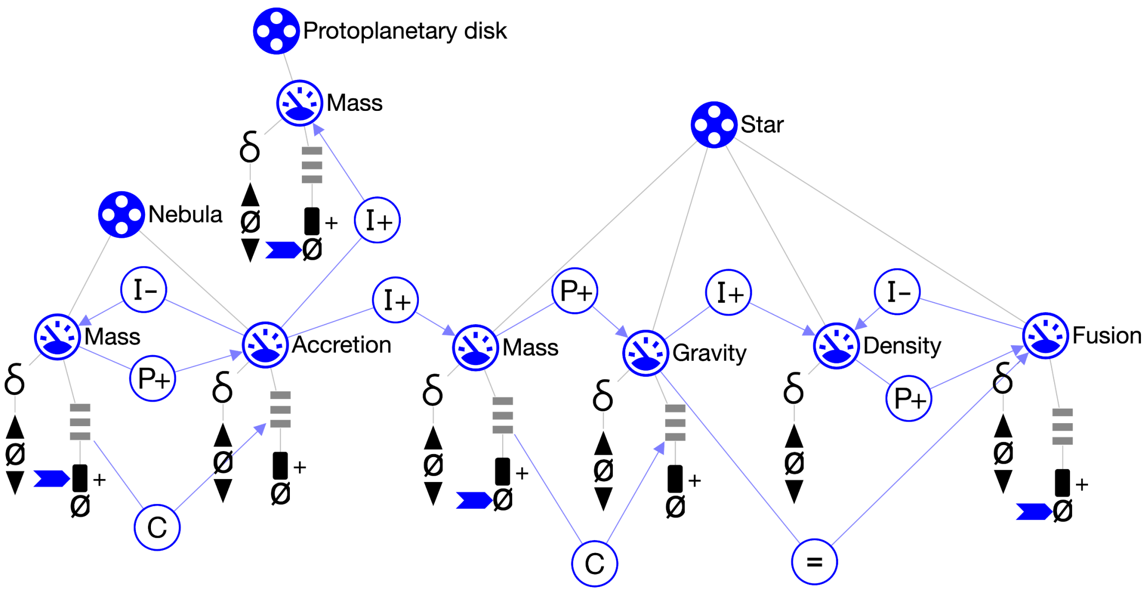


Figure 5. The *fusion-gravity balance* model and its initial setting. The model assumes a certain amount of Mass being present in the Nebula <+, ?>, while other quantities are zero <0, ?> (Masses of Star and Protoplanetary disk, and Fusion) or unknown <?, ?> (Accretion, Gravity, and Density). Note that in this starting state, Gravity=Fusion. In fact, both are still non-existing.

Table 1. Simulation results for the *fusion-gravity balance* model. Quantities have a value and a direction of change, represented as <v, ∂>.

	Nebula		Proto. disk	Star				
State	**Mass**	**Accretion**	**Mass**	**Mass**	**Gravity**	**Density**	**Fusion**	**Gravity ? Fusion**
1	<+, –>	<+, –>	<0, +>	<0, +>	<0, +>	<?, 0>	<0, 0>	Gravity = Fusion
2	<+, –>	<+, –>	<+. +>	<+, +>	<+, +>	<?, +>	<0, +>	Gravity > Fusion
3	<+, –>	<+, –>	<+. +>	<+, +>	<+, +>	<?, +>	<+, +>	Gravity > Fusion
4	<0, 0>	<0, 0>	<+. 0>	<+, 0>	<+, 0>	<?, +>	<+, +>	Gravity > Fusion
5	<0, 0>	<0, 0>	<+. 0>	<+, 0>	<+, 0>	<?, 0>	<+, 0>	Gravity = Fusion

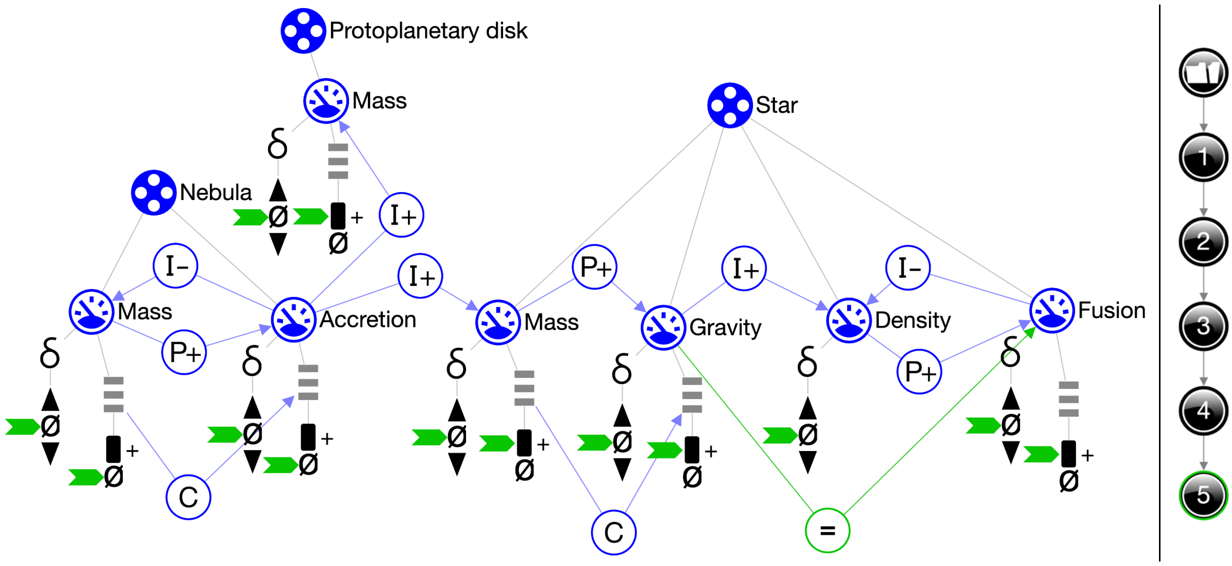


Figure 6. The simulation results for state 5 of the *fusion-gravity balance* model.

To support learners in developing this model, the workbook uses 6 steps. Each construction step is interleaved with simulation activities.

Learners built the full model from scratch and start by adding the Star with its Mass and Gravity, including the quantity spaces, the causal dependency, and the correspondence. Simulations are performed to ensure proper working of this first part. The second step focusses on Density being caused by Gravity and the fact that this is a process (steady gravity causing density to increase). Step 3 focusses on Accretion, but first only on the impact it has on the Mass of the Star. Note that, accretion is also a process. Step 4 includes the Mass of the Nebula and its relationship with Accretion. Step 5 focusses on Fusion and how it counteracts Gravity. Finally, step 6 adds the details regarding the Protoplanetary disk.

In addition to instructing learners in building the model and having them answer questions regarding the mechanisms, the workbook also presents notions of caution and the fact that a model is a simplification. E.g., it explains that the assumption that the mass of the star is zero is not entirely correct. That in fact, the star forms in the nebula. Hence, the moment the collapse of the nebula starts (i.e., accretion starts), the star already contains some material. For simplicity, however, the model assumes that the nebula and the star are separate from each other, so that the mass of the nebula flows into an 'empty' star.

7 Working with experts

Astrophysics experts contributed to creating the models presented in this paper. During each meeting improved versions of the model were presented to the experts for critical reflection. After consensus was reached with the first group, the model was reviewed by two further experts, in three consecutive sessions.

Most of the work focussed on the *star properties* model. In addition to clarifying terms and agreeing on the basic mechanism, most discussion concerned the notion of temperature and pressure before and after the start of nuclear fusion. Two postulates were formulated to reach consensus. Firstly, the model represents a family of stars, those in the *main sequence*, and not the specific behaviour of a single star. Hence, 'changing the mass of a star' (in the *star properties* and *star states* model) refers to comparing stars of different mass in the main sequence. Secondly, the quantities may refer to features at different moments during the lifespan of stars. As such, Pressure and Temperature (of the Core) refer to the features that led to the nuclear fusion starting, while Temperature and Gas pressure (of the Surface) refer to features that result from the nuclear fusion being active.

8 Classroom evolution

Evaluation of the lessons have been carried out (cf. [20]). Specifically, the *star properties* lesson has been evaluated in real classroom settings, the *star states* and *fusion-gravity balance* lessons have been pilot-tested with master students and reviewed by teachers.

Pilot. A pilot version of the three lesson activities were tested with three astrophysics master students, taking about 1 hour to complete a lesson. Students reflected on the activity and suggested improvements to the workbooks. The models remained unchanged.

Teachers. During a 90-minute teacher-training, physics teachers from the participating schools where informed about the three lesson activities and the evaluation study. Teachers agreed to reserve 90 minutes for *star properties* lesson, including a pre- and post-test.

Learners. One hundred and fifty-two learners from 9 classes from three secondary schools (across the Netherlands) participated in an evaluation study of the *star properties* lesson. Learners had no previous experience with learning by constructing qualitative representations. Results obtained during these lessons show that there is a significant positive effect of conceptual modelling on learners' understanding of the causal relationships between quantities of stars in the main sequence and the qualitative vocabulary [20].

9 Conclusion and Discussion

Three models and corresponding lessons have been developed to extend the Stargazing Live! mobile planetarium experience with lesson activities that relate to the Dutch secondary school physics curriculum. The lessons are available and can be taken online via https://dynalearn.eu/.

The *star properties* lesson focuses on learners identifying the key quantities that characterize stars and establishing the causal dependencies between those quantities. The *star states* activity follows on from the star properties lesson by adding ranges of qualitative values to six key quantities. During this lesson, learners learn how stars can be classified according to mass and how that relates to characteristic values for other quantities. The *fusion-gravity balance* model focusses on the birth of stars and how a balance emerges between the gravitational force (inwards) and the nuclear fusion force (outwards).

The lessons have been well-received by astrophysics master students and physics teachers in secondary education. The *star properties* lesson has been successfully evaluated in real classes in secondary education.

As future research we plan to evaluate the lessons on *star states* and on *fusion-gravity* balance. Furthermore, we intend to expand the set of conceptual modelling lessons to include other phenomena discussed in the planetarium lesson. For instance, we are currently developing conceptual modeling lessons related to circular and elliptical orbits of celestial bodies.

References

1. Agan, L. (2004). Stellar ideas: Exploring students' understanding of stars. *Astronomy Edu. Review*, 3(1), 77–97.
2. Bailey, J. M., Prather, E. E., Johnson, B., Slater, T. F. (2009). College students' pre-instructional ideas about stars and star formation. *Astronomy Edu. Review*, 8(1), 010110-1.
3. Bailey, J. M., Johnson, B., Prather, E. E., Slater, T. F. (2012). Development and validation of the star properties concept inventory. *Int. J. of Science Edu.*, 34(14), 2257–2286.
4. Branley, F. M. (1964). Planetariums – an Evaluation. *The Science Teacher*, 31(6), 18-19.
5. Bredeweg, B., Linnebank, F., Bouwer, A., Liem, J. (2009). Garp3 – Workbench for qualitative modelling and simulation. *Ecological Informatics*, 4(5-6), 263–281.
6. Bredeweg, B., Liem, J., Beek, W., Linnebank, F., Gracia, J., Lozano, E., Wißner, M., Bühling, R., Salles, P., Noble, R., Zitek, A., Borisova, P., Mioduser, D. (2013). DynaLearn - An Intelligent Learning Environment for Learning Conceptual Knowledge. *AI Magazine*, 34(4), 46–65.
7. Bredeweg, B., Kragten, M., Spitz, L. (2021). Qualitative Representations for Systems Thinking in Secondary Education. In: *Proc. of the 34th Int. QR Workshop*, Montreal, Canada.
8. Doyle, J. K., Ford, D. N. (1998). Mental models concepts for system dynamics research. *System Dynamics Review*, 14(1), 3-29.
9. Greca, I. M., Moreira, M. A. (2002). Mental, Physical, and Mathematical Models in the Teaching and Learning of Physics. *Science Edu.*, 86(1), 106–21.
10. Forbus, K. D. (2018). *Qualitative Representations. How People Reason and Learn About the Continuous World*. The MIT Press, Cambridge, MA.
11. Holt, J., Hanse, J., Baan, M., Groot, P., Bloemen, S. (2021). Co-creation 3.0: Taking the development of astronomy education resources to the next level in the project Star-gazing Live!. *Proc. of the 3rd Shaw IAU Workshop on Astronomy for Edu*. Int. Astronomical Union.
12. Holt, J., Hanse, J., Vaendel, D., Kragten, M., Bredeweg, B., Bloemen, S., Baan, H. M., Groot, P. (2022). Stargazing Live! Inspiring with semi-live astronomy data; teaching curriculum topics using smart education tools. *Proc. of the 4th Shaw IAU Workshop on Astronomy for Edu*. Int. Astronomical Union.
13. Plummer, J. D., Small, K. J. (2013). Informal science educators' pedagogical choices and goals for learners: The case of planetarium professionals. *Astronomy Edu. Review*, 12(1), 1-16.
14. Plummer, J. D., Kocareli, A., Slagle, C. (2014). Learning to explain astronomy across moving frames of reference: Exploring the role of classroom and planetarium-based instructional contexts. *Int. J. of Science Edu.*, 36(7), 1083-1106.
15. Plummer, J. D., Small, K. J. (2018). Using a planetarium fieldtrip to engage young children in three-dimensional learning through representations, patterns, and lunar phenomena. *Int. J. of Science Edu.*, 8(3), 193-212.
16. Schmoll, S. E. (2013). Toward a Framework for Integrating Planetarium and Classroom Learning (Doctoral dissertation).
17. Schultz, S. K., Slater, T. F. (2021). Use Of Formative Assessment-Based Active Learning By Astronomy Edu-

cators Teaching In Live Planetarium Learning Environments. *J. of Astronomy & Earth Sciences Edu.*, 8(1), 27-38.

18. Tippett, C. D. (2016). What recent research on diagrams suggests about learning with rather than learning from visual representations in science. *Int. J. of Science Edu.*, 38(5), 725-746.
19. Zimmerman, L., Spillane, S., Reiff, P., Sumners, C. (2014). Comparison of student learning about space in immersive and computer environments. *J. and Review of Astronomy Edu. and Outreach*, V1, pA5-A20.
20. Bredeweg, B., Kragten, M., Holt, J., Vaendel, D., Hanse, J., Bloemen, S. (2023). Stargazing Live! Inspiring with Real Data in a Mobile Planetarium and Learning Through Conceptual Modelling. In: *ITS 2023,* LNCS, 13891, 257-269. Springer, Switzerland.

AI Birds: Obstacles for Reaching Human-Level Performance and a New Role for Qualitative Reasoning

Diedrich Wolter[1], **Felix Haase**[1], **Alexandra Kirsch**[2]

[1] University of Bamberg, Germany [2] Independent Scientist

Abstract

Since 2012 the AI Birds competition hosted at major AI conferences sets out to challenge humans by fostering the development of autonomous agents that can outperform human players in a single-player physical simulation game. Unlike several other games, AI agents have not yet come close to human performance, let alone defeated average human players. In this paper we analyze what makes acting in physical environments hard and why computers show poor performance in open-world tasks. By evaluations performed on our agent that currently dominates the competition we aim to pinpoint to fundamental challenges which AI needs to face to ready itself for entering the open world. Our results show that the shortcomings are due to a lack of dynamics in common architectures. We then outline how qualitative reasoning can be applied to achieve a dynamic interplay of AI components.

1 Introduction

The AI Birds competition[1] (Renz et al. 2015) is carried out annually at major AI conferences since 2012. The aim of this competition is to assess the progress in AI towards problem solving in open domains whilst avoiding the challenges of working with technical systems such as robots and their limitations, thus putting a stronger focus on problem-solving skills (Renz et al. 2019). In short, the competition is based on the physical simulation game *Angry Birds* and requires an autonomous agent to catapult birds at structures protecting enemies in order to destroy them (see Figure 1). In a survey among AI researchers, AI Birds was estimated to be one of the next milestones of AI accomplishments in which an AI system will defeat humans by around 2022 (Grace et al. 2018). Since 2016, the BamBirds agent developed at the University of Bamberg participates in the competition and has won the competition three times so far. Like most agents participating, the code of the BamBirds agent is made publicly available.[2] We can thus use the BamBirds agent as a basis to discuss progress in the AI Birds competition. Also, the BamBirds agent can serve as a baseline to explore general shortcomings of today's AI approaches for open-world problem solving. The aim of this paper is to, first, give a description of the BamBirds agent and, second, to identify principle shortcomings that need to be addressed in order to make significant progress towards open-world problem solving in physical domains. A particular focus of this paper is to discuss possible contributions of qualitative reasoning. We also substantiate a claim that the problem areas identified encompass crucial gaps that need to bridged in order to reach for human-like performance in open words.

The remainder of this paper is structured as follows. In Section 2 we first introduce the AI Birds competition and discuss the challenges it encompasses for AI. Thereafter, Section 3 presents the BamBirds agent and discusses the contribution of distinct modules to successful performance in the competition. In Section 4 we then analyse principle limitations of the BamBirds agent that are symptomatic for current AI architectures. We identify research gaps and discuss means to overcome today's limitations. The paper concludes by summing up our key observations and claims.

2 The AI Birds Competition

In the AI competition, agents are confronted with a set of previously unseen levels. Within a set time limit, the agent has to gather as many points as possible by solving a level. The competition is run in several rounds: in the final, the two agents scoring highest in the semi-finals compete with one another; in the semi-final the four agents scoring highest in the quarter-finals compete; and so on, depending on the amount of agents participating. In each round a new set of unseen levels is used. In the finals, agents are typically given 20 minutes to solve 8 levels, allowing them to re-try each level about 2–5 times, depending on the complexity of the levels and agent speed. Each level (see Figure 1 for an example) comprises a set of target objects (green pigs), objects of different kinds, and a sequence of birds that can be launched from a slingshot by performing a drag-and-release operation. Once released, a bird is catapulted from the sling towards the area where pigs are positioned. By placing shots appropriately, all pigs must be destroyed, either by direct hit, or by any other physical impact of sufficient strength. When launching bird after bird once the physical scene has stabilised from the previous impact, a single level may take up to 2–3 minutes to play, depending on how many birds are available and shot. An agent is awarded points only if all pigs are destroyed. Points are determined by the game

[1] aibirds.org

[2] https://github.com/dwolter/BamBirds

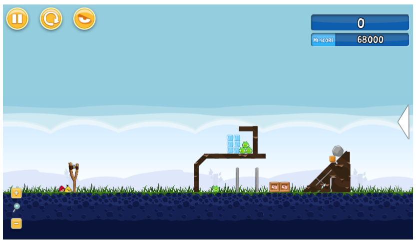


Figure 1: Example levels from AI Birds competition. To solve the level shown at the top, the blue bird (in the sling) has to be shot at the blue ice blocks in order to clear the path for the red bird (third in sequence). The yellow bird (second in sequence) must be shot at the wood structure that prevents the round stone on the right from rolling towards the TNT boxes on the ground. Explosion of the TNT boxes triggers a domino effect on the stone pillars, eventually destroying the bottom pig.

through some undisclosed formula that awards points to objects destroyed and a large amount of points to each unused bird when a level gets cleared. If an agent clears a level several times, it is awarded the maximum of points it has scored. The more damage is inflicted and the lower number of birds fired, the more points an agent receives. To make the game attractive to humans, several birds comprise special functions (e.g., blue birds can split into three smaller birds, allowing to hit multiple targets at once) and have unique effects when shot at particular game objects (e.g., yellow birds penetrate wood particularly well). Also, the game includes elements with special properties, in particular indestructible obstacles and explosives, which allow a great variety of levels to be constructed.

On a technical level, every agent communicates with the game through a network interface. Agents may request screenshots from the game and can issue click and click-drag-release (shot) actions. Moreover, agents can restart a level or select any of the levels from the current round. Also, agents can pan the view (for large levels) and control the zoom level. In later rounds of the competition, an agent may inquire the current best scores for each level. The AI Birds game is executed in a web browser window and can only be accessed via this interface. This setup has the following implications:

- game mechanics are concealed, i.e., physical simulation is performed with parameters unknown to the agent and can only be estimated from observations
- the interface is real-time, e.g., agents cannot quickly gather training data, even outside competitions

The competition challenges agents in two regards: solving individual levels and maximising the overall score.

2.1 Solving Levels

In order to solve a level, each agent has to interpret the screenshot and locate relevant objects. It is particularly important to identify the location of the slingshot and scale of the scene precisely in order to perform goal-directed shots as the game calculates flight trajectories with respect to the slingshot. If the pivot point of the sling is not estimated to within a few pixels accurately, no shot will be performed, or the bird drops off the sling. In order to clear a level, an agent has to plan a number of shots (two to five, typically) in an uncertain physical environment. Due to the lack of a reliable forward model and the sometimes chaotic reaction (e.g., how large structures collapse), uncertainty in action outcomes cannot be neglected.

2.2 Maximising Score

Typically, agents participating in the competition are not able to solve every level, at least not within the given time limit. As points are only awarded for levels solved, it is important to use some strategy for selecting which level to try next. Agents have to balance between re-playing a level already solved in order to improve the high-score, trying to crack a previously unsolved level, and not wasting time on levels unsolvable to them.

2.3 When Games are not Toys

AI has always considered games as benchmarks, be it for the public impact (like IBM's Deep Blue defeating Garri Kasparow or Google's AlphaGo defeating Lee Sedol), or for what Schaeffer called "microcosms of AI research" (Schaeffer 2014). Games may offer a convenient platform for conducting research as the rules of the game are fixed and clear—no bias by committing research to individual assumptions is at risk and results of different groups are easily comparable. Nevertheless, we believe one should reflect on a commitment to work on a game rather than a "real" problem that promises direct rewards for the society. As a physical simulation game, Angry Birds present a simplification of physical manipulation required for versatile service robots that eventually will assist humans in their everyday tasks, e.g., by getting dishes from cupboards, preparing meals for humans within an environment designed by humans for humans. For most labs, hardware for such versatile robots is beyond reach and even where such systems are available, technical challenges are manifold. For example, research on manipulation required for preparing meals like opening a bottle of milk, retrieving flour from their typical paper containers, etc. is hardly possible while contemporary progress in versatile robots is still involved with opening cupboards in kitchens, see (Kazhoyan et al. 2021). Above all, differences between robotic platforms used and the specific tasks considered hinder a direct comparison. In the light of the versatility of problems that can easily be constructed in a simple 2D physical simulation game (cp. (Stephenson, Renz, and Ge 2020)), the AI Birds competition thus constitutes a viable option for fundamental AI research that has prospects to improve future robotic applications. In particular, the physical nature of AI Birds is well-aligned with fundamental tasks and goals of qualitative reasoning (cp. (Forbus 2019)).

3 Synopsis of BamBirds Agent

The BamBirds agent is developed at the University of Bamberg, Germany. Its development is significantly supported by

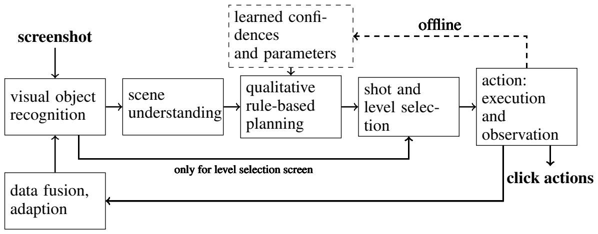
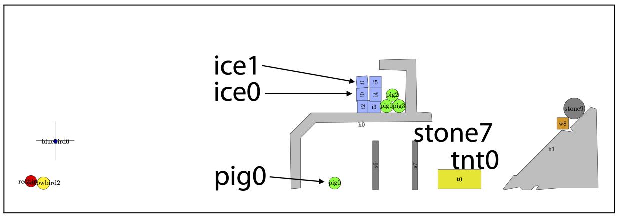


Figure 2: Architecture of BamBirds agents




```
belongsTo(ice0,struct2).
expl_affects(tnt0,stone7).
isOver(hill0,pig0).
protects(struct2,pig1).
slingshotPivot(145.2,332.8).
supports(ice0,ice1).
...
```

Figure 3: Example level from AI Birds competition depicted in Figure 1 as interpreted by BamBirds agent an excerpts from the respective scene descriptions.

student projects and thesis works. Within individual study groups, selected AI techniques that were expected to improve the agent are developed, implemented, and evaluated. Successful components are then integrated into the BamBirds agent. By design, BamBirds integrate GOFAI (Good Old-Fashioned AI) approaches like symbolic state space representation with probabilistic methods and lightweight machine learning. An explicit hybrid representation comprising quantitative and qualitative knowledge about levels is central to the design of the agent. BamBirds comprise the building blocks we detail in the following, and shown in Figure 2.

3.1 Visual Object Recognition and Scene Understanding

All planning hinges on a description of the situation the agent faces, in particular the objects within a level, their whereabouts, and the overall level scale (which can be derived from the size of the sling in pixels due to its constant size in spatial units). A precise representation is required for delivering precise shots at chosen targets.

Scene understanding itself is largely based on simple methods provided by the AI Birds organisers for visual object recognition to detect primitive objects; due to the graphic nature of the Angry Birds game, a simple approach already yields sufficiently good results in most cases. The visual recognition provided by the organisers and used in the BamBirds agents provides polygonal outlines of objects and a classification into the object types (wood, stone, ice, etc.). As an example, see Figure 3 for the example level from Figure 1 as seen by the BamBirds agent.

Visual object recognition is also responsible for detecting the game state, in particular to recognise that a ‘level won’ or ‘level lost’ screen appears and the agent is expected to select a new level.

From the geometric description of the scene, the agent derives qualitative spatial and physical relations that allow basic strategies to be grounded. To this end, two techniques are used. First, qualitative spatial relations (above, below, left of, etc.) are instantiated based on the semantics grounded in the location of objects. Second, a physical simulation using a 2D physics simulator is consulted to determine whether objects weigh on one another (for inferring stability) and to foresee selected effects of actions performed in the game. As physics simulation under uncertain start conditions is susceptible to noise and may easily yield wrong results, the component is only consulted for basic prediction of forces. In order to construct this component, we consulted the IHSEV agent which also includes physical simulation and used regression to fit parameters to the game (M. and Buche 2013). Generally, physical simulation is not robust due the nearly chaotic nature of how complex arrangements response to impact in conjunction with inevitable uncertainty in parameter estimation and visual object recognition. To illustrate, we point to the score of the IHSEV agent relying on physical simulation from the 2016 competition (AI Birds 2016) as reported in Table 1. As can be seen in the table, advancing from the quarter final to the final, the increasing level complexity correlated with the performance of IHSEV wrt. BamBirds decreasing (we note that absolute points are not comparable due to different amount of points that can be reached). Similar numbers can be observed in the 2019 competition where BamBirds defeated the simulation-based agent SimbaDD.

Table 1: score of BamBirds vs. IHSEV agent per round in 2016 competition, level complexity increases towards final

round	BamBirds	IHSEV
quarter final	280, 390	470, 940
semi final	406, 200	562, 820
final	451,250	288, 720

In short, the competition taught us that physical simulation is not reliable beyond simple inferences in semi-complex and complex environments. We therefore critically assess a much recognised argumentation for physical simulation in scene understanding (Battaglia, Hamrick, and Tenenbaum 2013), beyond grounding qualitative primitives on simple force calculations, e.g., `rests_on`.

As the output of the scene understanding module, a scene description in Prolog syntax is generated which contains both quantitative information (in pixel coordinates) and qualitative relations (see Figure 3 for an excerpt).

3.2 Qualitative Rule-Based Planning

The second and most involved component is responsible for determining possible shot candidates, given a scene description. By obtaining an explicit representation of qualitative relationships such as, for example, from `isOver(pig,ice)` and `supports(ice,stone)`, it is possible to design rules that serve as heuristics for identifying (potentially) useful shots. One of these rules states that by destroying an object that supports another, the now unsupported object will fall down and be likely destroyed. In the example above, aiming at the ice object could thus be a viable plan since the unsupported stone will fall onto the pig, destroying it. Until now, the rule base of BamBirds has been designed manually. Although BamBirds does not perform physical simulation for shots, the symbolic method is augmented with a quantitative estimator, e.g., to estimate the likelihood of penetrating objects by a single shot or the likelihood that a tower of objects will collapse when shooting at it. Also, an estimate is given whether the shot is expected to succeed. For example, a direct shot at a freely reachable pig is given full confidence, whereas a shot against a wall of objects to bounce off into the direction of some goal is given low confidence. To obtain functions for estimations, machine learning and regression has been performed on selected parameters from recorded games.

As a last step in the shot heuristic, a simple partial order planning is performed. In particular, shots are decreased in confidence if a later bird will be better suited to reach the goal, and shots are increased if the current bird is more useful for achieving some (intermediate) goal than forthcoming birds. A level taken from Stephenson and Renz (2018) that challenges lookahead planning is depicted in Figure 4. The player has a blue and a yellow bird, the blue bird must be shot first. The yellow bird can penetrate both wooden (yellowish) pillars, directly hitting the pig. The blue bird can only destroy one pillar, making the stones fall down and render the level unsolvable. Here, the agent has to waste the blue bird (e.g., shooting it over the structure or against the stone blocks) in order to finish the level with the yellow bird.

3.3 Shot Planning and Level Selection

The third component of our agent implements the shot selection from the set of candidates computed by the shot heuristic module. We approach the problem as heuristic search in a tree whose edges represent shots. For every shot performed we monitor its effect (e.g., the points awarded, pigs destroyed). When retrying a previously unsolved level the algorithm aims to find an alternative to a previously tried shot sequence. Consider again the example from Figure 4. Our agent lacks a forward model to anticipate that shooting the blue bird at the wooden pillars or the ice bar is a bad idea. However, once it has tried that shot (and noticed it has no plan for finishing the level with the yellow bird), the shot is discarded and an alternative is tried when revisiting the level. In absence of promising alternatives, the agent soon tries shooting at one of the stone objects (without much effect, if any) and is then able to finish the level with the yellow bird. Unlike classic game settings previously studied in AI, it is not possible to explore a significant portion of the search tree since exploration requires to engage in the game; only very few retries are possible during the competition.

In the 2021 competition, a clone of the BamBirds 2019 agent has won the competition which has chosen a parameter in favour of more exploration. Although winning 257,330 to 168,290 against BamBirds 2021 in the grand final[3], the BamBirds 2021 agent defeated its clone 312,910 to 270,200 in the previous round. We may therefore conclude that shot selection is critical but not sufficiently well evaluated in a single competition.

In a fourth and last step, once a level has been played, we decide which level to try next. We select the level that is expected to yield the largest reward considering information about the type of level, the number of previous attempts, the points that might be earned, the set of shot candidates not yet tried. We apply machine learning (offline) to obtain an estimator function that predicts the probability distribution for the performance of our agent based on previous attempt and features of the level. We then apply a randomised selection balancing potential gain with probability of success.

3.4 Action Execution and Monitoring

For all shots performed in the game, we monitor the effects to collect data and to determine when a level has stabilised after a shot, allowing the agent to plan its next shot.

3.5 Modul Interconnections and Adaption

For the most part, modules are executed one after another along the main horizontal axis shown in Figure 2. However, there is one notable exception to the linear flow, which is found in the module "adaption".

Precise visual object localisation is required for delivering precise shots at chosen targets. We found our agent to be too limited when relying on the visual object recognition techniques available. Therefore, we use data from the observed flight parabolas to improve estimates of scene scaling and sling position using regression on a per-level basis while the agent is playing. Most importantly, we trace the flight parabolas of each bird shot and, using regression, we adapt slingshot location and scaling parameters to align predicted shot parabolas to the observed one.

4 What is Missing in AI Problem Solving?

As we have seen, the typical "AI" approach of transforming real-life problems into optimisation problems fails for all but the simplest configurations in AI birds. On the level of physical processes alone, it seems to be unsolvable with current techniques. Of agents relying on machine learning, the agent DQ-Birds (Nikonova and Gemrot 2021) using a Deep Q-Network trained from about 115,000 situations was the best-performing agent so far with being able to solve 3 out of 8 levels scoring 185,869 points in the 2017 quarter finals, the last round it participated. Other learning-based agents have performed worse and teams decided to quit participating in the competition. By contrast, BamBirds scored 290,020

[3] http://aibirds.org/past-competitions/2021-competition/results.html

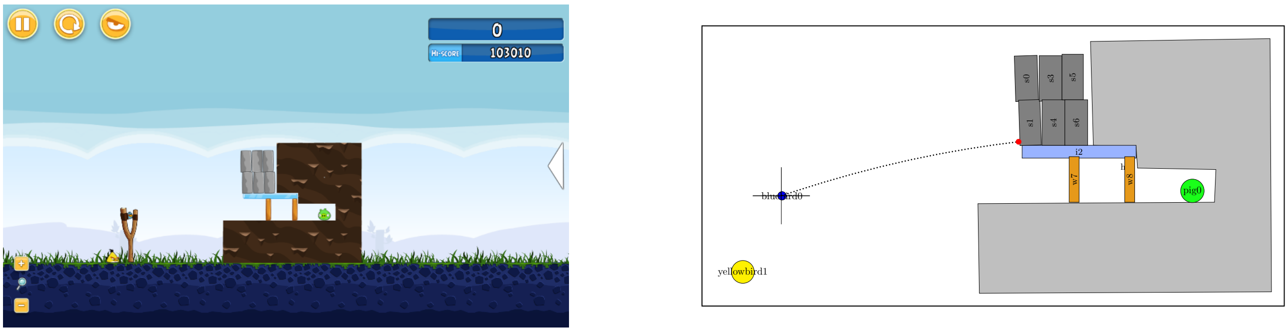


Figure 4: Example of level that requires lookahead planning

points in the 2017 quarter finals and the best-performing agent in that round 405,260 points.

Approaches relying on rules, inference, or planning on an abstract symbolic representation work for specific cases, but despite the fragile basis of symbols grounded in perception, they are still missing (and that regards all computational systems today) the ability to switch strategies, to "step out of the system" (Hofstadter 1979) and reconsider one's own understanding of the situation and the strategy to be applied.

AI has a long tradition on abstract representations: logic (Nebel 2001; McCarthy 2000), frame-based representations (Minsky 1974), and qualitative abstraction (Forbus 2019). Together with the representations we have powerful reasoning mechanism for, e.g., qualitative reasoning (Forbus 2019), analogical reasoning (Falkenhainer, D.Forbus, and Gentner 1989), and planning (Ghallab, Nau, and Traverso 2016).

It has turned out that none of these representations alone can represent and solve realistic problems (Forbus, Nielsen, and Faltings 1991). Ideas of how to combine different representations on a problem have been around for a long time as well, such as the mental image of a society of mind (Minsky 1986). Blackboard systems (Engelmore and Morgan 1988) have tried to provide an architectural basis for combining different representation and reasoning strategies.

Such approaches have been around for a while now and it feels this must be the direction to go. Still, none of them have made the step from hand-crafted systems to self-aware systems that understand the situation and act according to it.

4.1 Knowledge Representation and Transformation

From our experience, the main bottleneck is the interaction between different representations, especially symbolic and subsymbolic representations, often referred to as *symbol grounding*. Of course, there have also been attempts to do this, especially in robotics (see contributions in special issue (Coradeschi, Loutfi, and Wrede 2013)).

The fundamental flaw with all of these approaches is that they treat the task as a mathematical mapping from one representation to another. But there is no such mapping. When we transform subsymbolic information to a symbolic representation, we lose detail (usually numerical information), but we add interpretation. This interpretation always comes with some arbitrariness. The same numerical state representation can be part of different situations. Sitting in front of a black screen can mean that the computer is switched off, or that the screen is broken, or that the computer simply shows a fully black screen to name just a few possible interpretations. At this moment, there is no way to fully understand the situation based on perception. Some disambiguation can be done by including memory (remembering having switched on computer and screen), some may be possible after waiting some time (the computer showing something else than black), but others may need some interaction between reasoning, action and re-observation, like switching the screen off and on again.

The other way around we encounter the same problem. When some reasoning process has come to a conclusion like a "good" shot, it has done so with impoverished information, since it had abstracted from numerical values. Abstraction is great to focus a problem and keep the state space small. The problem is just that settling on one specific solution makes the whole process extremely fragile. It is up to luck whether the one solution settled on will really fit the situation. And when we transform the one abstract solution into a lower-level command, we again have to guess, this time the numerical values that are necessary for action execution, but that are not part of the output of the abstract solution process. If we then observe an action to fail, it may be due to a poor plan or a poor transformation.

4.2 Dynamics

We propose that the way out of the dilemma lies in a more dynamic view on computation. Even if the problem itself is static (as is the case for Angry Birds, at the moment the agent has to decide on an action, nothing changes in the setup), the solution process needs to be dynamic.

The basic idea is that we should replace the computational pipelines that are used today with a network of interacting modules, each of which is running its own decision-making process in a way described in (Kirsch 2019): the module would continually 1) consider a set of alternatives (which may be the output of some other module) and 2) evaluate and rank those alternatives (again a service that may be provided by other modules). Differently from the pipeline approach, no module would have to settle on one single solution. Of course, at some point an agent should act. This could be done if enough modules have converged to a stable solution, meaning that their most favoured alternative is not changing by any further decision iterations. Actions could also just be executed with a certain frequency, using the best-ranked alternative in some action module.

For example, during shot planning multiple shots are output which aim at the same object but at a different target point, each of them being a candidate for one specific goal (e.g., tossing over some object). If we know that one of these shots has succeeded, there is no need to consider the alternatives—if one shot goes terribly wrong by destroying the object, other shots at the same object will suffer from the same problem. The problem our agent is facing is that there are too many potential alternatives to consider. We could counter-act this problem by structuring the suggested shots and providing means to dynamically move within this structure. To this end, feedback information about a shot tried is required, revealing *how* it failed and *why* it might have failed. Qualitative relations allow us conveniently to describe how an action failed by comparing the actual outcome with the expected outcome. With respect to grasping causality, we again have to acknowledge that it will not be possible to single out the one reason, but only to rank alternative explanations.

In further steps, modules could even be added and removed (or switched on and off) depending on the situation (there would be a need for special modules to decide on the module configuration). The machine would not have to invent completely new modules, but it could decide to run modules with the same task, but being instantiated with different sets of parameters.

Why should this work? A clear argument for trying more dynamic processes is nature. There is no doubt that human thinking is dynamic, both on the neuronal level (Hawkins and Blakeslee 2004), on behavioral or problem-solving levels (Hayes-Roth and Hayes-Roth 1979; Newell and Simon 1972), also in the development of language and abstract concepts (Lakoff 1987). The reason why human thinking processes are dynamic, is surely the complex and dynamic world around us (Rittel and Webber 1973; Taleb 2010; Varela, Thompson, and Rosch 2016). Previous attempts by the authors in this direction have shown promising results (Kirsch 2017). We back up our claim by the following experiment: In Bambirds, we have a very simple form of dynamics implemented by proposing a certain type of risky shots (termed 'last resort shots') only if no other shots can be found. To study the effect of this simple form of dynamics, we compare it against a variant of Bambirds that always considers last resort shots. Running the agent on the 131 competition levels with a time limit of 5 minutes per level (about four tries per level), the agent in the dynamic condition was not able to solve 32 levels. When always considering last resort shots, 36 levels remained unsolved. Put differently, the agent performed better when dynamically increasing its set of plan candidates as compared to considering all plans at once. This experiment suggests that a dynamic interaction between shot planning and other modules is helpful.

Engineering Fears Interacting modules is about the last thing an engineer wishes for. While single modules are easy to control and debug, interaction always comes with uncertainty. A change in one module may break the whole system. It is exactly this kind of complexity that engineers try to avoid by a module pipeline.

Interaction, however, does not necessarily imply parallelism. In previous work we have explored interacting modules for robot navigation (Kirsch 2017). The modules were run sequentially, the resulting behavior was "rather" deterministic (since the study was run in a physical robot simulation, navigation tasks could be exactly reproduced, but the physical parameters still introduced some non-deterministic behavior in the actions).

Even with modules running in parallel, the behavior can be stable without extreme engineering overhead. In a retrospective of the Hearsay II blackboard architecture, the authors report: "A surprising result was that system performance, in terms of accuracy, was as good with the synchronization disabled as its performance with the full synchronization." (Lesser and Erman 1977, p. 797)

And there are theories around how to deal with dynamic systems, e.g., cellular automata (Wolfram 2002). It is just that the type of stability shown by dynamic systems is different than the exactly predictable input–output pairs we are used to from chaining functions in a processing pipeline. As soon as the environment exhibits uncertainty and dynamics, the pipe(line) dream comes to an end anyway. Instead of trying to force environments into our engineering wishes, we should rather accept the challenge and learn to deal with it.

Interfaces When different modules use their own representations, we need to find a way to combine them. Blackboard architectures (Engelmore and Morgan 1988) are an attempt to channel the complexity of interacting modules to a central memory where all modules communicate. This makes the information flow easy to track and to debug (all the relevant information is in the central blackboard memory). But it also makes it hard or almost impossible to find the one representation that fits all modules. The experiences described on the Hearsay II architecture (Lesser and Erman 1977) confirm what can be expected: at the end, one does construct special pieces of information that are only relevant and useful for some modules.

Therefore, instead of trying to put all pieces of information in one central memory in a unified language, we suggest to try networks of interconnected modules. Each module must support communication interfaces to its neighbors, in the way known from current pipeline structures. Such a network constrains the options for adding and removing modules, but as stated above, we do not expect to generate fully new modules any time soon. Additional modules could be clones from other modules and would have a matching place in the network of modules.

We want to emphasize that modules form a network, not a hierarchy. One module may use a more abstract representation than another, but that does not put it "above" the other module. An observation from a neuroscientist friend: When we look at graphical representations of modules in the human brain, each scientist will put the module she is working on in the center of any diagram, but if you were to draw the full picture, there is no ‘upper module’. All the pieces are connected, and *in all directions*. A network only makes sense with information passing in both directions, otherwise we would be back at a processing pipeline.

4.3 A New Role for Qualitative Reasoning

The considerations above motivate us to propose a new approach to qualitative reasoning in agents. Rather than only using QR in the classical form of describing a process abstractly, we advocate to use QR to describe the interplay of modules. To give an example, the feedback loop in the Bambirds agent that adapts parameters for visual scene recognition from observations could be described using qualitative rules that explain how scene parameters must be changed (i.e, increased, or decreased), given how a shot missed the target anticipated. QR techniques can then be applied to govern the convergence process of modules, similar to how QR rules about throwing objects like “reduce launch speed if throwing too far” make action selection converge more quickly (Wolter and Kirsch 2015).

5 Summary and Conclusion

This paper presented the BamBirds agent, which has won the AI Birds competition three times. The agent is based on several modules that are involved with visual object recognition and scene understanding, shot planning, shot selection, an action module, and feedback components that allow the agent to improve (during gameplay by adapting parameters, during development by learning estimators from recorded data). We discuss why we believe a physical simulation game cannot be tackled with existing AI techniques such as machine learning or QR alone, but motivates basic AI research. Despite the survey among AI experts (Grace et al. 2018) that projected the arrival of an agent defeating humans in Angry Birds around the year 2022, we are pessimistic that an agent will come close to human performance in the near future. We argue that a severe limitation of today’s approaches is due to static architectures of independent modules that lack the ability to reflect their decisions and to reach their output in a dynamic process of interacting with other modules. In order to let the modules step out of their static roles, researchers must also step out of their beaten path of higher degrees of specialisation in AI research and focus more on AI architectures and how they allow existing techniques to be integrated. Rather than aiming for a precise method to govern module interactivity, we argue that QR techniques are of interest which steer convergence processes, but allow components to interact in a dynamic manner.

References

AI Birds. 2016. Results from 2016 Competition. http://aibirds.org/past-competitions/2016-competition/competition-results.html. Last accessed 2022-05-22.

Battaglia, P. W.; Hamrick, J. B.; and Tenenbaum, J. B. 2013. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45): 18327–18332.

Coradeschi, S.; Loutfi, A.; and Wrede, B., eds. 2013. *KI special issue “Symbol Grounding”*, volume 2/2013. Springer.

Engelmore, R.; and Morgan, T., eds. 1988. *Blackboard Systems*. Addison-Wesley Publishing Company.

Falkenhainer, B.; D.Forbus, K.; and Gentner, D. 1989. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41(1): 1–63.

Forbus, K. D. 2019. *Qualitative Representations: How People Reason and Learn about the Continuous World*. Cambridge (MA), USA: MIT Press.

Forbus, K. D.; Nielsen, P.; and Faltings, B. 1991. Qualitative Spatial Reasoning: The Clock Project. *Artificial Intelligence*, 51(1-3): 417–471.

Ghallab, M.; Nau, D. S.; and Traverso, P. 2016. *Automated planning and acting*. Cambridge University Press.

Grace, K.; Salvatier, J.; Defoe, A.; Zhang, B.; and Evans, O. 2018. When will AI exceed human performance? *Journal of Artificial Intelligence Research*, 62.

Hawkins, J.; and Blakeslee, S. 2004. *On Intelligence*. Times Books.

Hayes-Roth, B.; and Hayes-Roth, F. 1979. A Cognitive Model of Planning. *Cognitive Science*, 3(4): 275–310.

Hofstadter, D. R. 1979. *Gödel, Escher, Bach: an eternal golden braid*. VintageBooks.

Kazhoyan, G.; Stelter, S.; Kenfack, F. K.; Koralewski, S.; and Beetz, M. 2021. The Robot Household Marathon Experiment. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Kirsch, A. 2017. A Modular Approach of Decision-Making in the Context of Robot Navigation in Domestic Environments. In *3rd Global Conference on Artificial Intelligence (GCAI)*, 134–147.

Kirsch, A. 2019. A Unifying Computational Model of Decision Making. *Cognitive Processing*, 20(2): 243–259.

Lakoff, G. 1987. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. The University of Chicago Press.

Lesser, V. R.; and Erman, L. D. 1977. A retrospective view of the Hearsay-II architecture. In *IJCAI’77: Proceedings of the 5th international joint conference on Artificial intelligence*.

M., M. P.; and Buche, C. 2013. Towards A Theory-Of-Mind-Inspired Generic Decision-Making Framework. In *IJCAI 2013 Symposium on AI in Angry Birds*.

McCarthy, J. 2000. concepts of logical AI. In Minker, J., ed., *Logic-Based Artificial Intelligence*, 37–58. Dordrecht, Holland: Kluwer.

Minsky, M. 1974. A Framework for Representing Knowledge. Technical report, MIT-AI Laboratory.

Minsky, M. L. 1986. *The society of mind*. Simon and Schuster.

Nebel, B. 2001. Logics for Knowledge Representation. In *International Encyclopedia of Social and Behavioral Sciences*. Elsevier.

Newell, A.; and Simon, H. 1972. *Human Problem Solving*. Upper Saddle River, New Jersey: Prentice Hall.

Nikonova, E.; and Gemrot, J. 2021. Deep Q-Network for Angry Birds. Technical Report 1910.01806v2 [cs.AI], arXiv.

Renz, J.; Ge, X.; Gould, S.; and Zhang, P. 2015. The Angry Birds AI Competition. *AI Magazine*, 36(2): 85–87.

Renz, J.; Ge, X. Y.; Stephenson, M.; and Zhang, P. 2019. AI meets Angry Birds. *Nature Machine Intelligence*, 1: 328.

Rittel, H. J. W.; and Webber, M. M. 1973. Dilemmas in a General Theory of Planning. *Policy Sciences*, 4.

Schaeffer, J. 2014. The Games Computers (and People) Play. In *Proceedngs of AAAI*.

Stephenson, M.; and Renz, J. 2018. Deceptive angry birds: towards smarter game-playing agents. In *FDG '18: Proceedings of the 13th International Conference on the Foundations of Digital Games*, 1–10. Article No. 13.

Stephenson, M.; Renz, J.; and Ge, X. 2020. The computational complexity of Angry Birds. *Artificial Intelligence*, 280: 103232.

Taleb, N. N. 2010. *The Black Swan*. Penguin Books.

Varela, F. J.; Thompson, E.; and Rosch, E. 2016. *The Embodied Mind: Cognitive Science and Human Experience*. MIT Press. Revised edition.

Wolfram, S. 2002. *A New Kind of Science*. Wolfram Media.

Wolter, D.; and Kirsch, A. 2015. Leveraging Qualitative Reasoning to Learning Manipulation Tasks. *Robotics*, 4: 253–283. Special Issue Representations and Reasoning for Robotics, N. Bellotto, N. Hawes, M. Sridharan, D. Nardi (eds).

Qualitatively Constrained Control Policy Learning

Domen Šoberl,[1] **Jure Žabkar** [2]

[1] Faculty of Mathematics, Natural Sciences and Information Technologies, University of Primorska, Slovenia
[2] Faculty of Computer and Information Science, University of Ljubljana, Slovenia
domen.soberl@famnit.upr.si, jure.zabkar@fri.uni-lj.si

Abstract

This paper introduces a novel approach that leverages qualitative reasoning to enhance reinforcement learning in physical domains. Traditional reinforcement learning methods often suffer from sample inefficiency and lack of explainability, especially in complex, physics-driven environments. Our proposed approach addresses these challenges by integrating qualitative induction and qualitative planning to learn a control strategy. Our method enables faster convergence of the learning process and yields an interpretable and physically plausible model of the environment. It employs a unique feedback loop mechanism that iteratively improves the qualitative model of the environment based on the observed outcomes of the executed plans, allowing continual refinement of the system's understanding and actions. Through an extensive set of experiments, we demonstrate a superior performance of our method compared to a state-of-the-art deep reinforcement learning method.

Introduction

Reinforcement Learning (RL), especially when used in Deep Learning (DL), has excelled in various environments, from discrete-action games to continuous control in robotics (Sutton and Barto 2018; Mnih et al. 2015; Silver et al. 2017; Vinyals et al. 2019; Silver et al. 2014; Lillicrap et al. 2015; Ibarz et al. 2021). However, deep architectures typically require large training data and consequently take a long time to converge. The obtained models offer minimal insight into their internal control policies. Model-based RL methods partly address these issues by combining learning with planning and using state transition models to enhance efficiency and explainability (Moerland et al. 2023; Plaat, Kosters, and Preuss 2023; Milani et al. 2022).

When training an agent within a physical domain (e.g. in robotics - either real-world or simulated), integrating some degree of physical knowledge into the training mechanism can significantly speed up the training process. In RL, domain knowledge is typically coded in the reward function that guides the agent faster towards the goal (Grzes 2017). Reward shaping is limited to the form of a real-valued function and cannot encode generalized laws of physics. Instead of dealing with physics directly, the agent tries to maximize the received rewards, handcrafted by a domain expert that considered certain laws of physics when designing it.

Qualitative representations (Forbus 2019) show a promising direction towards sample efficient learning (Šuc 2003; Žabkar et al. 2011), learning explainable models (Bratko 2008, 2011; Košmerlj, Bratko, and Žabkar 2011), and learning explainable control strategies (Šoberl and Bratko 2019). These representations can bridge the gap between numerical and symbolic machine learning, and can therefore be used for symbolic learning and planning in continuous robotic environments (Žabkar, Bratko, and Mohan 2008; Wiley, Bratko, and Sammut 2018; Šoberl and Bratko 2020).

In this paper, we propose a qualitative approach to learning continuous control policies, where instead of a rewarding mechanism, qualitative reasoning about the physical environment is used to constrain the search space and guide the system toward the goal state. A qualitative model is built and a qualitative state space determined during the first few episodes of training and then being continuously refined over new observations as the training continues. The training is guided along qualitative plans, which are devised from the observed state transitions, and so avoiding spurious solutions. We evaluate the performance of our proposed method on the swing-up problem for the inverted pendulum and compare it to the performance of the Deep Deterministic Policy Gradient (DDPG) on the same problem (Lillicrap et al. 2015).

In comparison to similar qualitative control methods, the contributions of this paper are the following:

- The learned qualitative model, the inferred qualitative state space and the execution parameters are being continuously refined, while the existing methods refine only the execution parameters.
- The structure of the qualitative state space is inferred from the observed numerical behaviors instead of from the rules of qualitative simulation, which eliminates the possibility of spurious plans and confines the computational complexity of planning to the complexity of the breadth-first search algorithm.
- Qualitative control is demonstrated on the swing-up benchmark domain and its performance is compared to the performance of the deep reinforcement learning algorithm DDPG.

Related work

First attempts at solving physical and mechanical problems qualitatively date back to the 1980s, when the concept of *Qualitative Differential Equations* (QDE) as a simplified alternative to ordinary differential equations (ODE) was introduced (Forbus 1984; De Kleer and Brown 1984). Kuipers devised a way to use QDEs to simulate dynamic systems and introduced an algorithm for *qualitative simulation* called QSIM (Kuipers 1986). Forbus formalized the concept of *action* in the context of qualitative simulation (Forbus 1989), which complied with the paradigm of classical planning, where actions are deterministic and instantaneous. Such a formulation was used by Sammut and Yik (Sammut and Yik 2010) to devise a qualitative plan for a bipedal robot walking, which was executed through trial-and-error (Sammut and Yik 2010). A similar approach was used by Wiley et al. to train a multi-tracked vehicle to climb the stairs (Wiley, Bratko, and Sammut 2018).

When first introduced, QDEs were abstracted from ODEs manually. However, when a numerical model of the domain is not known or cannot be easily devised by intuition, there is a tendency to learn it from numerical traces. This was largely made possible with the introduction of Multivariate Monotonic Qualitative Constraints (MMQC) (Wellman 1991). These types of constraints can be induced directly from numerical data and represented in the form of decision trees (Bratko and Šuc 2003), or abstracted as qualitative partial derivatives and constructed into a model with any chosen classifier (Žabkar et al. 2011). Because these qualitative models abstract away most of the numerical information and present the learned theory in the form of increasing and decreasing intervals, they better comply with the human intuition that the traditional numerical models (Bratko 2008).

Practical applications demonstrated that by autonomously interacting with the physical world, a robot can learn qualitative representations that could intuitively be interpreted as *obstacle*, *stability* and *movability* (Leban, Žabkar, and Bratko 2008; Košmerlj, Bratko, and Žabkar 2011). The robots were interacting with high-level actions and observations. Mugan and Kuipers (Mugan and Kuipers 2012) proposed learning on the motor level and developed an unsupervised learning algorithm QLAP (Qualitative Learner of Action and Perception) that uses qualitative representations as a way to discretize the input data. The learned models are represented as Dynamic Bayesian Networks (DBNs).

Using learned qualitative models on the motor level to execute robotic tasks requires a method for resolving the effects of actions on a qualitative level. This approach alone allows a reactive execution of tasks, simple enough to be solved without planning. This was demonstrated on the problems of pursuing a goal, avoiding collision and pushing a box (Šoberl, Žabkar, and Bratko 2015; Šoberl and Bratko 2017). Employing qualitative planning, explainable control strategies can be devised, which was demonstrated on the problem of balancing the inverted pendulum (Šoberl and Bratko 2019). Combining qualitative planning with reactive execution, a quadcopter was able to learn an explainable controller to navigate through space and plan its way around a simple maze (Šoberl and Bratko 2020). A domain-independent framework was proposed soon after (Šoberl 2021).

Learning qualitative control vs. Reinforcement learning

Existing qualitative approaches to learning motor-level control learn a qualitative model first, then devise a qualitative plan, which is finally refined through execution. The learned model and the qualitative plan remain unchanged for the remainder of the execution. The solution is either a quantified qualitative plan (Sammut and Yik 2010; Wiley, Bratko, and Sammut 2018) or a qualitative plan with a numerically fine-tuned execution policy (Šoberl and Bratko 2019, 2020). In both cases, learning is model-based and goal-oriented. This is fundamentally different from reinforcement learning, where the learned policy is refined continuously as new observations are collected. In the case of the model-free Q-learning method, no model of the environment is learned, while the learned policy aims to maximize the reward function, typically without any notion of a goal state.

Each of the two approaches has certain advantages over the other: Reinforcement learning can be used with a wider range of discrete and continuous environments and requires a large set of training samples, obtained over a large number of training episodes. The learned control policy is purely numerical, typically in the form of a deep neural network. The latter methods do not provide any explanation of why a certain action is taken in a certain state. The policy is numerically bound to the training environment and requires retraining in case the environmental parameters change.

The fundamental idea behind learning qualitative control is to narrow down the search space by constraining it in two ways: (i) with qualitative constraints that arise from the laws of physics, and (ii) with domain-specific qualitative constraints learned through experimentation with the environment. The first type of constraints restrict the use of qualitative control methods to continuous real-world environments — typically to robotic domains. The reasons to constrain the search space qualitatively instead of numerically are: (i) the generality of qualitative physics, and (ii) sample efficient learning of qualitative models. Qualitative physics is more general than conventional physics in the sense that it abstracts away numerical constants and works with symbolic time. It, therefore, predicts a succession of qualitative states instead of exact numerical states at precise times. This makes the devised qualitative solutions transferable to environments with the same qualitative dynamics, but different numerical parameters. The execution parameters need to be re-tuned, but qualitative models and plans remain unchanged. Moreover, qualitative abstractions comply with the way humans reason about the physical world and are therefore a feasible basis for explainability (Bratko 2008; Šoberl and Bratko 2019). The key differences between reinforcement learning and learning qualitative control are summarized in Table 1.

The presumption taken in previous research on learning qualitative control is that constraining the search space qual-

	Reinforcement learning	Learning qualitative control
Environment types	Arbitrary mechanics	Real-world physics
Trainable entity	(Deep) neural networks	Qualitative constraints
Background knowledge	Reward function (explicit)	Qualitative physics (implicit)
Reward function	Crucial for success	Used as performance metric
Goal	Maximize the reward	Reach a goal state
Sample efficientcy	Low	High
Explainable policy	No	Yes, through qualitative behaviors
Transferable policy	No	Yes, qualitative models and plans

Table 1: Key differences between reinforcement learning and learning qualitative control.

itatively reduces the training time. It is a reasonable premise, considering the fact that model-free reinforcement learning usually attempts many absurd and non-intuitive actions, before finding a working solution. However, this has — to the best of our knowledge — not yet been demonstrated and evaluated on a common benchmark domain. One of the reasons for the lack of such a comparison is the fundamentally different ways in which the two methods approach a problem and hence the lack of a common evaluation metric. In this paper, we introduce certain adaptations to the qualitative control method, that bring it closer to the paradigm of reinforcement learning. Training is executed over multiple episodes of limited duration and the received rewards are used as a performance metric, although they are not utilized by the qualitative method for training.

The proposed method

Any qualitative method of acting in a numerical environment would inevitably assume at least the following four phases: (1) data collection, (2) qualitative abstraction, (3) qualitative reasoning, and (4) numerical implementation. Qualitative abstraction lifts numerical data to a qualitative level so that qualitative reasoning can take place, while numerical implementation acts in reverse: it quantifies a qualitative solution so that it can be executed in the numerical environment. Our method denotes these four phases as:

1. *Data collection*. Sensory data is collected either by random or systematic experimentation or by motor babbling.
2. *Qualitative abstraction*. A qualitative model is induced from the collected numerical data.
3. *Qualitative planning*. A qualitative plan is found from the current state to a goal state.
4. *Plan execution*. The obtained qualitative plan is executed reactively — one action at a time in a closed control loop.

Existing approaches to qualitative control employ similar four phases, but mostly in linear succession, with the execution phase being the only one done in a closed loop. We propose expanding the sensory feedback also to the phases of qualitative induction and qualitative planning. In such a non-linear setting, these can no longer be deemed *phases*, but rather *levels* of acting. *Experimentation* is the only phase of acting done separately from the rest. It is performed during the first few episodes of training to collect the minimum required samples to induce a qualitative model. In reinforcement learning, the first few episodes are often also purely explorational. Samples are then collected for the remainder of the training and used to refine the parameters on each level.

Data collection

The objective of the initial data collection is to provide enough numerical samples to induce a qualitative model. The goal is to model how actions affect the observable numerical state. For example, modeling the behavior of a simple pendulum, we model how the force $F \in \mathcal{X}$, applied to the pendulum, affects its radial acceleration $\ddot{\theta} \in \mathcal{Y}$ at a certain position and radial velocity $\theta, \dot{\theta} \in \mathcal{C}$. When parts of the state space exhibit different operational principles than others, samples should be collected in all operating regions (as in (Šoberl, Žabkar, and Bratko 2015; Wiley, Bratko, and Sammut 2018)).

Qualitative induction

Qualitative induction is a process of generating qualitative models from numerical data (Bratko and Šuc 2003). In this paper, we consider qualitative models that capture monotonically increasing and decreasing intervals of continuous functions. For instance, function $y = x^2$ has two such intervals — it is monotonically decreasing in all $x < 0$ and monotonically increasing in all $x > 0$. Point $x = 0$ is considered a *critical point*, a border between two *operating regions*. In this paper, we use Padé (Žabkar et al. 2011), which allows us to systematically introduce into the modeling the action variables $\mathcal{X}$, dependant variables $\mathcal{Y}$, and conditional variables $\mathcal{C}$. We do this in the following way:

$$\begin{matrix}\text{numerical}\\\text{samples}\end{matrix} \xrightarrow{\mathcal{X},\mathcal{Y}} \text{Padé} \xrightarrow{\mathcal{C}} \text{classifier} \longrightarrow \begin{matrix}\text{qualitative}\\\text{model}\end{matrix}$$

In the case of only one operating region, the resulting qualitative model would consist of qualitative constraints of the form $\mathcal{Y} = Q^{\{+,-\}}(\mathcal{X})$, which are valid everywhere within the domain. If there is more than one operating region, the model would contain multiple sets of such constraints, each set conditioned by values from $\mathcal{C}$.

Let us recall the pendulum example from the previous section. Since we only have direct control over the output variable $F \in \mathcal{X}$, we want to model the effect of F on the radial acceleration $\ddot{\theta} \in \mathcal{Y}$. Assuming that this effect may be qualitatively different depending on $\theta, \dot{\theta} \in \mathcal{C}$, we use Padé to find qualitative dependencies of the form $\ddot{\theta} = Q^{\{+,-\}}(F)$ and a classifier (typically a decision tree learner) to learn operating regions determined by θ and $\dot{\theta}$.

Qualitative planning

Qualitative planning considers a search through the *qualitative state space* from an initial to a goal qualitative state; a *qualitative plan* is a qualitative behavior, permitted by the given qualitative model (Wiley, Bratko, and Sammut 2018; Šoberl and Bratko 2020). The definition of the qualitative state space has been adopted from Kuipers' QSIM (Kuipers 1986) and adapted for planning by extending it with the concept of action. We adopt the same definition of the qualitative state space, but base its internal structure on qualitatively abstracted observations, instead on the QSIM's theoretical framework. For instance, suppose that a robot is moving uphill toward some critical point. The robot's path can in such case be qualitatively abstracted as *the interval before the critical point* and *the interval after the critical point*. Because of the continuity of the path, QSIM would presume the possibility of transitioning between the two intervals. However, the hill may in reality be too steep for the robot to reach the critical point and transition to the next qualitative state. Our method, therefore, presumes the possibility of transitioning between two qualitative states only if the transition has already been observed.

To promote some level of exploration, maximum depth d is given as a parameter and all qualitative plans up to depth d are constructed. A randomly chosen plan is then selected for execution. If no plan is found, a random action is executed and planning is repeated from the new state, hopefully with new observations allowing for a wider set of state transitions. In contrast to reinforcement learning, where at every step a single action is chosen either randomly or by the learned policy, our method commits to plans rather than to single actions.

Plan execution

Execution of qualitative plans is the problem of implementing a continuous transition between two consecutive qualitative states $S_i \rightarrow S_{i+1}$ of the plan within a particular numerical environment. We consider reactive control: the agent selects and executes an action several times per second after observing the current numerical state with the same frequency. At each reactive step, the executor is solving two problems: (i) determining which qualitative action will produce the most desirable effect towards state S_{i+1}, and (ii) translating the qualitative action to output numerical signals.

The question of how to resolve the effects of qualitative actions through qualitative models has been addressed in (Šoberl and Bratko 2017). *Qualitative action* has been defined as *an instruction on the directions of change of control variables*. Formally, let $c_{\{1...m\}} \in \mathbb{R}$ be control variables that represent output signals (e.g., signals to control the motors). Then a qualitative action A formalized as $A = [c_1 : \mathrm{dir}_1, \ldots, c_m : \mathrm{dir}_m]$, where $\mathrm{dir}_{\{1...m\}} \in \{\mathrm{inc}, \mathrm{dec}, \mathrm{std}\}$. Action $A = [c_1 : \mathrm{inc}, c_2 : \mathrm{dec}, c_3 : \mathrm{std}]$ would therefore instruct the value of c_1 to be increased, the value of c_2 decreased, and c_3 kept at its current value. There is no information on the rates of change at this stage.

The process of action prioritization is mathematically formulated as follows. Let variables x_i for all $i \leq k$ and some k have a target value, and let variables x_i for all $i > k$ be numerically constrained. Numerical constraints are optional[1] and determine the fail states, e.g. numerical constraint $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$ specifies the allowed interval for variable θ. Let e_i be the respective time estimates of variables x_i. Define the function:

$$f(e_1, \ldots, e_n) = \sum_{i=1}^{k} \frac{{e_i}^2}{2} + \\ + \sum_{i=k+1}^{n} \Big((1+e_i)^{-1} \cdot (1-e_i)^{-1} - 1 \Big) \quad (1)$$

and denote its gradient as:

$$\nabla f = \left(\frac{\partial f}{\partial e_1}, \ldots, \frac{\partial f}{\partial e_n} \right). \quad (2)$$

Let $E = [\mathrm{dir}_{x_1}, \ldots, \mathrm{dir}_{x_n}]$ be the vector of qualitative effects of action A on variables $x_1, \ldots, x_n$ as deduced through the qualitative model, where qualitative directions $\mathrm{dir}_{x_i} \in \{\mathrm{inc}, \mathrm{std}, \mathrm{dec}\}$ are mapped to integers as $\mathrm{inc} \mapsto 1, \mathrm{std} \mapsto 0, \mathrm{dec} \mapsto -1$. The priority $Q(A)$ of action A is then computed as

$$Q(A) = -\nabla f \cdot E. \quad (3)$$

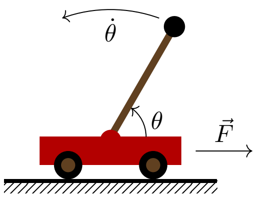
In previous work (Šoberl and Bratko 2020; Šoberl 2021), it was presumed that acceleration is constant over the entire domain. If the agent was observed to accelerate at a certain rate, it was presumed that it can decelerate at the same rate at any location. However, there are domains where this is not the case. A pendulum, for instance, will exhibit different accelerations under the same applied force when in different positions, due to the force of gravity. Therefore, in this paper, we propose predicting the maximum possible deceleration through linear regression. From the collected numerical samples, we select the points near the goal state to model the relation between the control variables and the observed accelerations. Using linear regression, we then predict the maximum deceleration rates at the goal position. The predictions get refined as new samples are collected throughout training episodes.

The experimental domain

To demonstrate our approach, we simulated a classic benchmark control problem of swinging up the inverted pendulum. A freely rotating pole is attached to a cart as shown in

[1] In this paper we do not use numerical constraints, but they can be useful in other control domains, e.g. (Šoberl, Žabkar, and Bratko 2015; Šoberl and Bratko 2019; Šoberl 2021).

Figure 1. The cart can be moved either left or right by applying a force in the direction of motion. Consequently, the pole accelerates rotationally either clockwise (CW) or counterclockwise (CCW). The pendulum is initially in the downward position ($\theta = -180°$). The goal is to lift the pendulum upward ($\theta = 0°$) and maintain it in this state. We do not impose any goals or constraints on the position of the cart. We orient the coordinate system so that the force F is positive in the right direction, and the pole's angle θ increases in the CCW direction. The length of the cart is $1\,\mathrm{m}$ and weighs $10\,\mathrm{kg}$. The pole is $1\,\mathrm{m}$ long and weighs $1\,\mathrm{kg}$. The force F is applied to the cart with the frequency of $50\,\mathrm{Hz}$ and can be between $-100\,\mathrm{N}$ and $100\,\mathrm{N}$. We limit the maximum speed of the pole to $360°/s$ in any direction. We did not simulate noise.



θ — The angle of the pole.

$\dot{\theta}$ — The angular velocity of the pole.

$\vec{F}$ — The force applied to the cart.

Figure 1: The state $(\theta, \dot{\theta})$ and the action $(\vec{F})$ in our cart-pole domain.

The observable state of the system is $(\theta, \dot{\theta})$, while $\ddot{\theta}$ can be derived from the observed $\Delta\dot{\theta}$ and Δt. The initial state is $(\theta = -180°, \dot{\theta} = 0)$ with $F = 0$, and the goal state is defined as $(\theta = 0°, \dot{\theta} = 0)$. We use the reward function

$$R(\theta, \dot{\theta}) = -\left(\theta^2 + 0.1 \cdot \dot{\theta}^2\right), \tag{4}$$

which is similar to the reward function used in the OpenAI Gym pendulum environment (Brockman et al. 2016), except for the torque component ($+0.001 \cdot u^2$), which we here omit. In the Gym pendulum domain, force F is applied directly to the tip of the pole, so the translation of the force to the torque is straightforward. In our inverted pendulum domain, the force is transferred via the cart to the base of the pole, which makes the torque not directly observable. Moreover, the impact of the torque on reward values is small due to a low coefficient of 0.001. The two methods compared in this paper – reinforcement learning and our qualitative control method — are hence both driven by the same observable quantities θ and $\dot{\theta}$, the former through optimizing the received rewards and the latter by pursuing the goal state. Recall that our qualitative method does not employ a reward function; we use it here only to be able to compare the performance of both methods.

Reinforcement learning is usually done over multiple *episodes*. After each episode, the system is reset to its initial state and the training is repeated with the updated policy. The DDPG method updates the weights and biases of the actor and the critic neural networks (as described in (Lillicrap et al. 2015)), while our qualitative control method updates (i) the qualitative model, (ii) the set of state transitions and (iii) the linear regression model to predict accelerations. We set the length of each episode to 6 seconds, which is 300 steps with the $50\,\mathrm{Hz}$ action frequency. We identify no fail states in this domain, although, in general, a fail state would also terminate an episode.

Results

In this section, we separately present the results of qualitative induction, qualitative planning, and execution that we obtained in our simulated inverted pendulum domain. We compare the results of execution with those obtained by the DDPG algorithm that we ran in the same simulator and under the same conditions.

Qualitative induction

The training of the inverted pendulum started purely experimentally. Random forces F were applied to the cart with random durations between 20 and 500 ms. Numerical samples $(F, \theta, \dot{\theta})$ were captured 50 times per second, and $\ddot{\theta}$ computed as $\Delta\dot{\theta}$ under the constant time step $\Delta t = 20$ ms between two consecutive observations. What we aim to model is how the force F affects $\ddot{\theta}$ in any given state $(\theta, \dot{\theta})$. Since $\ddot{\theta}$ is the time derivative of $\dot{\theta}$ and the latter of θ, the effect of F on those higher integrals would be simulated by the qualitative planner.

After 3 episodes, 900 samples were obtained, which sufficed for inducing a qualitative model with an average numerical error below $4°$ for θ. Considering that the maximum speed of the pole $\dot{\theta}$ is $360\,°\,\mathrm{s}^{-1}$ and that observations are collected with the frequency of 50 Hz, up to 7.2° input error is possible just from temporal resolution. The collected numerical samples and the induced qualitative model are shown in Figure 2. It can be seen from the model that two operating regions have been learned: $\ddot{\theta} = Q^{+}(F)$ for $-89.94° \geq \theta < 93.53°$ and $\ddot{\theta} = Q^{-}(F)$ outside that interval. The theoretically correct boundaries are $-90.0°$ and $90.0°$ respectively. The error decreased with further sampling, dropping below $1.2°$ after 30 episodes. The learner determined that $\dot{\theta}$ plays no role in specifying the operating regions.

The obtained qualitative model can be interpreted in the following way:

If the pole is in the upright position (above the horizontal line), increasing the force on the cart will increase the acceleration of the pole, and vice-versa.

If the pole is in the downright position (below the horizontal line), increasing the force on the cart will decrease the acceleration of the pole, and vice-versa.

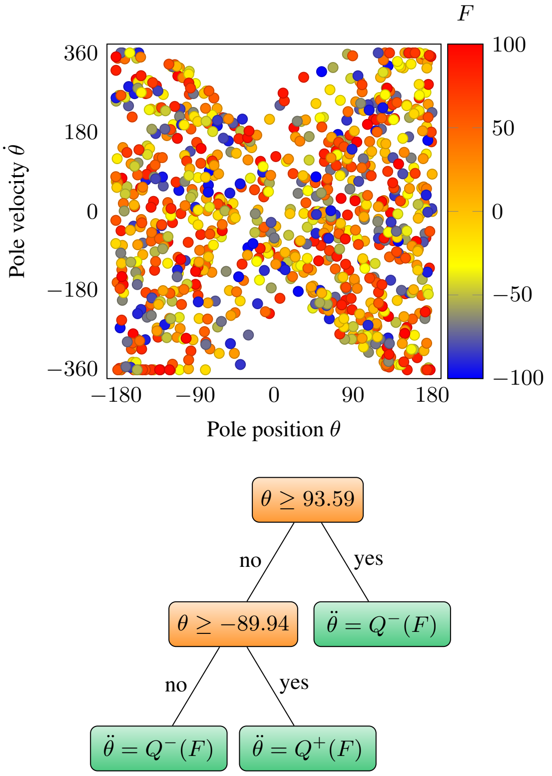


Figure 2: Qualitative induction from numerical samples. Above: Samples collected during the experimentation episodes. Below: A qualitative model induced after processing the numerical samples with Padé.

Qualitative planning

The qualitative model was passed to the qualitative planner, together with the initial state ($\theta = -180°, \dot{\theta} = 0$) and the goal state ($\theta = 0°, \dot{\theta} = 0$), the planner immediately determines the landmarks of the configuration space $(\theta, \dot{\theta})$:

$$\begin{aligned} \theta &: \{-180, -90, 0, 90, 180\}, \\ \dot{\theta} &: \{\min, 0, \max\}. \end{aligned} \tag{5}$$

For clarity, we here write the theoretical boundaries -90 and 90 for the two operating regions, although the actual values with the induced model are -89.94 and 93.53. The *min* and *max* landmarks are symbolic representations of the minimum and maximum values, which at this point are not yet known.

These landmarks determine 24 possible qualitative states. The possible qualitative values for θ are:

$$[-180], [-180..-90], [-90], [-90..0], [0], [0..90], [90], [90..180],$$

and for $\dot{\theta}$:

$$[\min..0], [0], [0..\max],$$

to which we will also respectively refer to as *neg*, *zero*, and *pos*.

Transitions between the qualitative states are deduced from numerical observations. Most of the possible transitions were abstracted from the 900 numerical samples used to induce the qualitative model. From these, the circular topology ($-180° = 180°$) of θ was also discovered and incorporated into the initial space partitioning (5). As seen from Figure 2, the 900 samples are more densely concentrated around the initial position $\theta = -180°$, while being sparse at the goal position $\theta = 0°$. For this reason, the transitions abstracted from these samples were also denser around the initial position, while most transitions around the goal position were discovered in later episodes. A typical scenario occurring during the early episodes of training would be making the pole rotate a full circle after overshooting the goal, because the possibility of stopping the pole close to the goal and reversing its direction had not yet been observed. Still, the early plans did tend to bring the pole closer to the goal more often than random exploration, hence new transitions were eventually discovered and with them the possibilities of new plans.

Figure 3 shows how a qualitative plan was found to swing up the pendulum. The shorter plan, leading from the initial state $[-180/\text{zero}]$ to goal state $[0/\text{zero}]$ failed to transition from $[-180..-90/\text{zero}]$ to $[-90/\text{zero}]$ due to the lack of momentum to overcome the force of gravity. A longer plan was then deduced which takes a different path after the point of failure. This way the concept of swinging to build up the momentum was discovered.

Figure 4 shows a plan deduced after the goal position $\theta = 0°$ is overshot. Instead of transitioning from $[0..90/\text{neg}]$ to $[0/\text{zero}]$, the pendulum would overshoot to state $[-90..0/\text{neg}]$. A plan would then be devised to reverse the direction of the pendulum back to the goal position.

Execution

We compared the performance of our qualitative execution with the performance of DDPG, which we configured as follows: the *actor* and the *critic* neural networks both contained two hidden layers of 64 and 32 nodes, both using the ReLu activation function. The actor took two inputs, θ and $\dot{\theta}$, and output via the Tanh activation function a continuous action within $[-1, 1]$, which was scaled to $F \in [-100, 100]$ before being executed. The critic took the same input as the actor, together with the actor's output, and output the Q-value via the linear activation function. Both networks used the Adam optimizer with the learning rate $\alpha = 10^{-3}$. The Q-learning discount factor was set to $\gamma = 0.99$. The training was done in batches of 32 samples with unlimited replay memory size. We tried various other DDPG configurations with different neural network architectures but achieved the best results using the described settings.

The training performance of both algorithms is shown in Figure 5. The plots were obtained by running each algorithm 100 times and averaging the episode rewards obtained in each training episode. The episode reward itself is the average of the 300 rewards received during an episode (recall that the duration of each episode is 300 steps). Both algorithms eventually converged toward the same strategy — build up the momentum by swinging and then balance

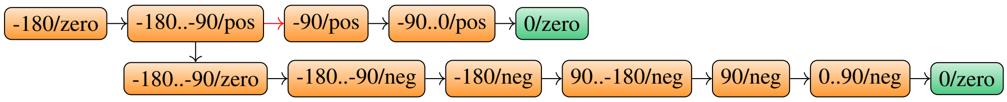


Figure 3: A qualitative plan found to swing up the pendulum. The alternative (lower) branch is deduced when the transition marked with the red arrow fails to execute.

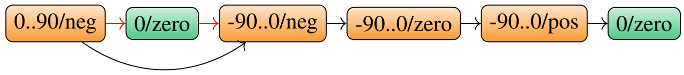


Figure 4: A qualitative plan found to correct a goal overshoot. The alternative (lower) branch is deduced when the pendulum fails to stop exactly at the goal position.

at the goal position. However, our qualitative algorithm converged significantly faster. The level of performance reached by DDPG after about 150 episodes, was achieved by our qualitative executor around episode 50 in the worst case.

Conclusion

This paper aims to bridge the gap between reinforcement learning and learning qualitative models, which have previously been researched separately and, to the best of our knowledge, never evaluated on the same control problem. The type of qualitative modeling and reasoning that we focus on in this paper is based on *monotonic qualitative constraints*, which are more sample efficient to learn, have better noise resiliency and offer a higher level of explainability than traditional numerical models.

We demonstrated our method on a single benchmark control problem and compared it to state-of-the-art deep reinforcement learning method. The approach should easily be transferrable to other control domains. We find it somewhat more difficult to implement than a typical reinforcement learning algorithm, since it contains a three-layered feedback loop, the complete Padé learner, a qualitative physics engine, and a non-trivial execution mechanism that is capable of dynamically adapting to the environment. It also cannot be used in (simulated) environments that do not comply with Newtonian physics.

References

Bratko, I. 2008. An assessment of machine learning methods for robotic discovery. *Journal of Computing and Information Technology - CIT*, 16(4): 247–254.

Bratko, I. 2011. Autonomous Discovery of Abstract Concepts by a Robot. In *Adaptive and Natural Computing Algorithms*, 1–11.

Bratko, I.; and Šuc, D. 2003. Learning qualitative models. *AI magazine*, 24(4): 107–119.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. arXiv:1606.01540.

De Kleer, J.; and Brown, J. S. 1984. A qualitative physics based on confluences. *Artificial Intelligence*, 24(1): 7–83.

Forbus, K. D. 1984. Qualitative Process Theory. *Artificial Intelligence*, 24(1–3): 85–168.

Forbus, K. D. 1989. Introducing Actions into Qualitative Simulation. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'89, 1273–1278. Morgan Kaufmann Publishers Inc.

Forbus, K. D. 2019. *Qualitative Representations: How People Reason and Learn about the Continuous World*. MIT Press, 1st edition.

Grzes, M. 2017. Reward Shaping in Episodic Reinforcement Learning. In *Adaptive Agents and Multi-Agent Systems*.

Ibarz, J.; Tan, J.; Finn, C.; Kalakrishnan, M.; Pastor, P.; and Levine, S. 2021. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5): 698–721.

Košmerlj, A.; Bratko, I.; and Žabkar, J. 2011. Embodied Concept Discovery through Qualitative Action Models. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 19: 453–475.

Kuipers, B. 1986. Qualitative simulation. *Artificial Intelligence*, 29(3): 289–338.

Leban, G.; Žabkar, J.; and Bratko, I. 2008. An Experiment in Robot Discovery with ILP. In *Proceedings of the 18th International Conference on Inductive Logic Programming*, 77–90.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N. M. O.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971.

Milani, S.; Topin, N.; Veloso, M.; and Fang, F. 2022. A Survey of Explainable Reinforcement Learning.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M. A.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518: 529–533.

Moerland, T. M.; Broekens, J.; Plaat, A.; and Jonker, C. M. 2023. Model-based Reinforcement Learning: A Survey. *Foundations and Trends® in Machine Learning*, 16(1): 1–118.

Mugan, J.; and Kuipers, B. 2012. Autonomous Learning of High-Level States and Actions in Continuous Environ-

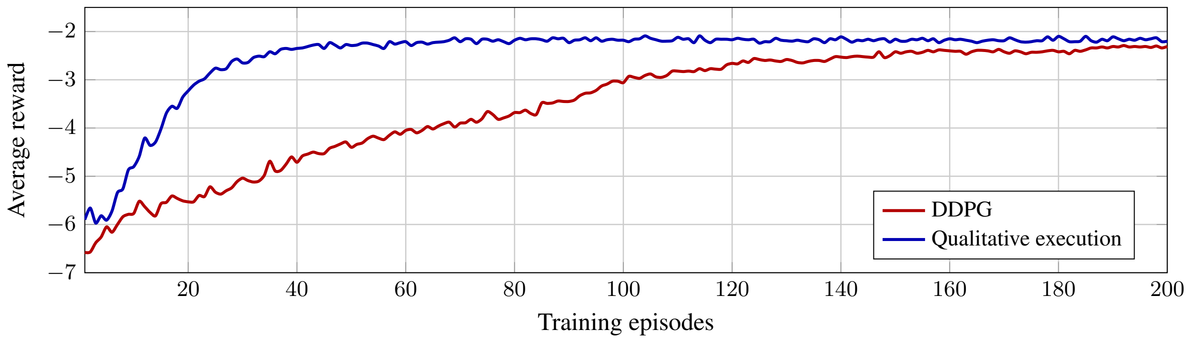


Figure 5: The performance of the two evaluated algorithms. The plots show how the reward converges over the number of training episodes. The plotted values are the averages of 100 repetitions.

ments. *IEEE Transactions on Autonomous Mental Development*, 4(1): 70–86.

Plaat, A.; Kosters, W.; and Preuss, M. 2023. High-accuracy model-based reinforcement learning, a survey. *Artificial Intelligence Review*.

Sammut, C.; and Yik, T. F. 2010. Multistrategy Learning for Robot Behaviours. In *Advances in Machine Learning I: Dedicated to the Memory of Professor Ryszard S.Michalski*, 457–476. Berlin, Heidelberg: Springer Berlin Heidelberg.

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic Policy Gradient Algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, 387—395. JMLR.org.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; Chen, Y.; Lillicrap, T.; Hui, F.; Sifre, L.; van den Driessche, G.; Graepel, T.; and Hassabis, D. 2017. Mastering the game of Go without human knowledge. *Nature*, 550(7676): 354–359.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; Oh, J.; Horgan, D.; Kroiss, M.; Danihelka, I.; Huang, A.; Sifre, L.; Cai, T.; Agapiou, J. P.; Jaderberg, M.; Vezhnevets, A. S.; Leblond, R.; Pohlen, T.; Dalibard, V.; Budden, D.; Sulsky, Y.; Molloy, J.; Paine, T. L.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Wu, Y.; Ring, R.; Yogatama, D.; Wünsch, D.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T. P.; Kavukcuoglu, K.; Hassabis, D.; Apps, C.; and Silver, D. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 1–5.

Wellman, M. P. 1991. Qualitative Simulation with Multivariate Constraints. In *Second International Conference on Principles of Knowledge Representation and Reasoning*, 547–557.

Wiley, T.; Bratko, I.; and Sammut, C. 2018. A Machine Learning System for Controlling a Rescue Robot. In *RoboCup 2017: Robot World Cup XXI*, 108–119.

Šoberl, D. 2021. *Automated planning with induced qualitative models in dynamic robotic domains*. Ph.D. thesis, University of Ljubljana.

Šoberl, D.; and Bratko, I. 2017. Reactive Motion Planning with Qualitative Constraints. In *Advances in Artificial Intelligence: From Theory to Practice*, 41–50. Springer International Publishing.

Šoberl, D.; and Bratko, I. 2019. Learning Explainable Control Strategies Demonstrated on the Pole-and-Cart System. In *Advances and Trends in Artificial Intelligence. From Theory to Practice*, 483–494. Springer International Publishing.

Šoberl, D.; and Bratko, I. 2020. Learning to Control a Quadcopter Qualitatively. *Journal of Intelligent & Robotic Systems*.

Šoberl, D.; Žabkar, J.; and Bratko, I. 2015. Qualitative Planning of Object Pushing by a Robot. In *Foundations of Intelligent Systems*, 410–419. Springer International Publishing.

Šuc, D. 2003. *Machine Reconstruction of Human Control Strategies*. Frontiers in Artificial Intelligence and Applications. IOS Press, Inc.

Žabkar, J.; Bratko, I.; and Mohan, A. C. 2008. Learning qualitative models by an autonomous robot. In *22th International Workshop on Qualitative Reasoning*, 150–157.

Žabkar, J.; Možina, M.; Bratko, I.; and Demšar, J. 2011. Learning qualitative models from numerical data. *Artificial Intelligence*, 175(9–10): 1604–1619.

The Difficulty of Novelty Detection in Open-World Physical Domains: An Application to Angry Birds

Vimukthini Pinto[1], Cheng Xue[1], Chathura Gamage[1], Matthew Stephenson[2], Jochen Renz[1]
[1]School of Computing, The Australian National University, Canberra, Australia. [2]College of Science and Engineering, Flinders University, Adelaide, Australia.
vimukthini.inguruwattage@anu.edu.au

Abstract

Detecting and responding to novel situations in open-world environments is a key capability of human cognition and is a persistent problem for AI systems. In an open-world, novelties can appear in many different forms and may be easy or hard to detect. Therefore, to accurately evaluate the novelty detection capability of AI systems, it is necessary to investigate how difficult it may be to detect different types of novelty. In this paper, we propose a qualitative physics-based reasoning method to quantify the difficulty of novelty detection focusing on open-world physical domains. We apply our method in the popular physics simulation game (PSG) Angry Birds and conduct a user study across different novelties to validate our method. Results indicate that our calculated detection difficulties are in line with those of human users.

1 Introduction

With the increasing reliance on autonomous systems such as self-driving cars and planetary robots, detection and adaptation to novel situations have become important capabilities for such AI systems. For example, if an autonomous car is not trained on slippery roads, the car may fail to detect when the friction is reduced and adjust the speed accordingly. Open-world learning is an emerging research area that attempts to address the challenge of detecting and adapting to novel situations (Langley 2020). Open-world learning research requires adequate evaluation protocols to capture the performance of agents under the two tasks: detection and adaptation (Senator 2019). This paper focuses on creating a difficulty measure for novelty detection to aid the evaluation of novelty detection by disentangling agents' performance from the intrinsic difficulty of novelties.

The novelties we encounter in an open world can take various forms (Boult et al. 2021). In this paper, we focus on *structural transformation*, a very common type of real-world novelty where an unknown object is encountered or a previously known object changes one/more of its properties (Langley 2020). For example, this could be a new vehicle type on the road, a new type of product in the supermarket with new packaging, a previously empty box filled with goods, or an abnormally heavy ball in a billiards game. As these examples suggest, only some of the novelties can be identified from appearance. Novel objects with different appearances can be detected by observing the change in color or shape. Quantifying the difficulty of detecting them can be addressed with the use of concepts presented in color science (Giesel and Gegenfurtner 2010) and research conducted on object shapes and sizes (Perner 2018). However, the difficulty of detecting novel objects with the same appearance but different physical parameters (e.g., mass, friction, bounciness) is not addressed so far. It is also not straightforward as one needs to interact with the objects and observe changes in their movements. Moreover, the detectability of such novelty depends on several factors: the physical parameter that is changed or the number and arrangement of novel objects in the environment.

This paper presents a qualitative-physics based method to quantify the difficulty of detecting novel objects with the same appearance but altered physical parameters (compared to previously seen versions of the objects). The proposed method aids a thorough evaluation by disentangling agents' performance from the difficulty of detecting the novelty. For example, if the novelty cannot be identified from the appearance and occurs in an object that is not reachable to interact, then the novelty cannot be detected. Therefore, the difficulty of novelty detection should be considered before making conclusions on the detection ability of an agent. The method we propose is agent-independent and enables us to evaluate an agent's performance (both detection performance and task performance) at different levels of difficulty (that can be categorized as easy, medium, and hard). We apply our method to the popular PSG Angry Birds, as it has semi-realistic physics and provides an ideal platform to introduce novelty (Gamage et al. 2021). We then conduct a user study experiment with human participants to verify that the calculated novelty detection difficulty values are in line with those of humans.

2 Background and Related Work

This section presents the notion of difficulty and novelty in the context of physical worlds and AI. We also discuss the related work in qualitative physical reasoning and a brief description of our experimental domain.

Difficulty Assessing difficulty is a popular research area in neuroscience where researchers are interested in quantifying the difficulty of tasks or decisions (Franco et al. 2018).

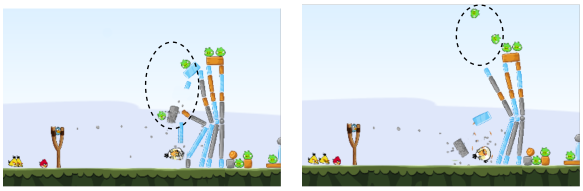


Figure 1: Examples from Angry Birds. The figure in the left (a) has original parameters whereas the figure in the right (b) has an increased *bounciness* parameter for pigs. The two figures show the difference in pig's movement for the same shot in the original (a) and increased *bounciness* (b) of pigs.

Measuring difficulty is also a main topic of discussion when measuring the intelligence of AI systems (Chollet 2019). It is also a widely studied topic in the gaming industry to make games interesting to players. The flow-theory, one of the most prevalent models in the game design, suggests that the games should not be too easy or too difficult to maintain player enthusiasm (Takatalo et al. 2010). Considering the difficulty of detection, researchers have studied this in areas such as emotion detection (Laubert and Parlamis 2019) and missing content detection (Yom-Tov et al. 2005). However, to our best knowledge, the difficulty of novelty detection in physical domains is not studied so far and is important when evaluating the detection capabilities of agents.

Novelty In the context of AI, novelty is described as situations that violate implicit or explicit assumptions about the agents, the environment, or their interactions (SAIL-ON-BBA 2019). Similarly, Boult et al. formalize a theory of novelty for open-world environments and Langley explains different types of environmental transformations that can be considered as novelty. Following these ideas, the novelties we consider in this paper occur as a result of changed physical parameters of objects. It could be the mass, friction, elasticity, etc. These novelties do not change the appearance of the object but cause it to behave differently after an interaction. For example, in the real-world, a novelty could be a new tennis ball with higher bounciness than the balls seen before, a previously empty bottle now filled with water, or a box of goods with less weight due to a manufacturing defect. Figure 1 shows differences in the observed movements of objects after physical parameters have been changed in the research clone of Angry Birds (Ferreira and Toledo 2014).

Qualitative Physics As discussed previously, novelties based on physics parameters are not detectable from appearance alone. Therefore, one needs to interact with the objects and observe any difference in their expected movements. Humans are often unaware of the exact physical parameters such as density, friction, and mass distribution of objects and do not need to solve complex differential equations when reasoning about their movements, instead relying on spatial intelligence (Walega, Zawidzki, and Lechowski 2016).

To analyze object movements, a qualitative physics approach was proposed to approximate structural stability based on the extended rectangular algebra (ERA) (Zhang and Renz 2014). ERA comprises 27 interval relations based on the approximated center of mass of the object and offers more flexibility than the original 13 interval algebra relations (Allen 1983). Ge, Renz, and Zhang point out that ERA is more suitable to approximate the stability of a single object rather than a structure and extends the use of ERA by proposing two qualitative stability analysis algorithms: *local stability* and *global stability*. A similar algorithm, *vertical impact* is proposed by Walega, Zawidzki, and Lechowski, which combines the concepts of *local stability* and *global stability* into one algorithm. They also introduced the algorithm *horizontal impact*, to provide a heuristic value to the interaction based on force propagation. Our difficulty measure also uses the algorithm *vertical impact* along with new reasoning components which reason about the nature of the object movements that are necessary to detect novelty.

Experimental Domain Our experimental domain, Angry Birds is a PSG where the player shoots birds at pigs from a slingshot. These pigs are often protected by different physical structures that are made up of three types of materials: wood, ice, and stone. There are also static platforms, which are indestructible objects that are not affected by forces. The task of an agent that attempts to detect novelty is to identify if anything has changed from the normal game environment by shooting at game objects. As the original Angry Birds game by Rovio Entertainment is not open source, we conduct our experiments using a research clone of the game (Ferreira and Toledo 2014). One example of novelty in Angry Birds is displayed in figure 1. As Figure 1a shows, the agent who is familiar with the normal game environment expects the pigs to fall down without bouncing up after an interaction. However, when the bounciness parameter is increased, the agent observes a change in the pigs' movement as shown in Figure 1b.

We selected Angry Birds as our experimental domain due to three reasons. First, solving an Angry Birds game instance (game level) requires reasoning about object movements in complex physical structures (Zhang and Renz 2014). Second, there are many game levels and level generators (Stephenson et al. 2019) that enable us to evaluate our difficulty measure on a diverse set of levels. Third, this is an ideal platform to vary different physics parameters and add the type of novelties we are investigating in this study. Moreover, Angry Birds is a very popular domain among AI researchers with several long-running competitions associated with it (Renz et al. 2015, 2019).

3 Overview of the Difficulty Measure

This section presents a high-level overview of our difficulty measure formulation for novelty detection in physical domains. We define the following to aid our explanations.

- Each object consists of a set of appearance-related parameters and a set of physical parameters. There is a predefined many-to-one mapping from appearance parameters to physical parameters (objects with the same appearance always have the same physical parameters and two or more objects with different appearance can have the same physical parameters). Objects with the same appearance are referred to as an object type. The number of object types is predefined.

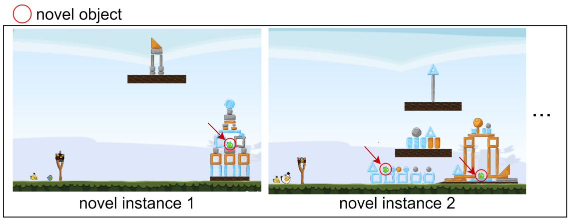


Figure 2: A set of novel instances. Each instance contains one/-more novel pigs denoted by the red circle and a set of normal objects. Note that, this paper focuses on novel objects with the same appearance as non-novel objects but with different physical parameters. Therefore, novel objects cannot be distinguished visually.

- *normal object*: An object that preserves the predefined mapping between appearance and physical parameters.
- *novel object*: An object that violates the predefined mapping between appearance and physical parameters.
- *normal instance*: An arrangement with a collection of *normal objects*.
- *novel instance*: An arrangement with a collection of *normal objects* and *novel objects*. (See Figure 2)

Our measure has three properties. Our difficulty measure:

1. is instance-based, i.e., we provide the difficulty of detecting novelty for a specific novel instance.
2. quantifies the difficulty of detecting novelty when an agent encounters the novel instance for the first time (the agent does not attempt the instance multiple times).
3. is agent independent (i.e., we do not collect data from agents to develop the measure).

Given below are three assumptions we make.

1. As designers of the difficulty measure, we have full understanding of the novel instance (i.e., the novel object, location of the novel object, the changed parameters, and the value of the parameters).
2. The agent has a full understanding of the object dynamics in the normal environment. The agent is fully aware of how objects move without novelty and the agent can detect that the environment is novel if a change in movements happens in the novel environment (perfect detection). We made this assumption because the detection difficulty can be different across agents; therefore, our measure is based on the lower bound of the detection difficulty by assuming perfect detection.
3. The agent attempts to detect novelty using a sequence of interactions. i.e., the agent cannot have multiple interactions at the same time. For example, in the billiards game, an agent can move only one ball at a time.

Figure 3 shows the components of our difficulty measure formulation.There are two inputs, the initial state of an instance (i.e., the state of an instance before any interactions) and the novelty present in the instance. Novelty present can be a set of objects with their changed physical parameter (e.g. {(wood objects, mass), (stone objects, friction)}). Our first module, the *Target Determining Module* takes the above

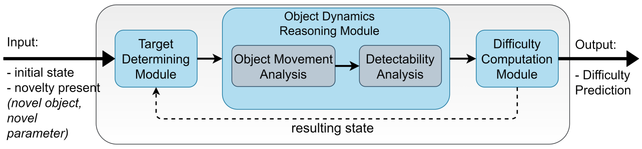


Figure 3: Overview of the method to compute the difficulty of novelty detection.

two inputs and searches possible target objects, i.e., the objects an agent can interact with. This module outputs all possible target objects in the given state.

The second module, *Object Dynamics Reasoning Module* has two components, the *object movement analysis* component and the *detectability analysis* component. The *object movement analysis* component takes each target object and identifies other objects that are moved due to the interaction with the target object. Next, the *detectability analysis* component determines if the interaction has caused the novel object to move in a detectable way. For example, when a novel object has a different sliding friction value, an interaction that causes the novel object to fall from above would not make the novel object detectable. In contrast, an interaction that causes the novel object to slide on a surface would make the novel object detectable.

Knowing the target objects that make detectable movements, the *Difficulty Computation Module* quantifies the difficulty of novelty detection to the given initial state. If the algorithms in the difficulty computation module require the next state to predict the difficulty, the updated states (i.e., state after an interaction) are sent to the *Target Determining Module* (dotted arrow in Figure 3) and the process iterates until the difficulty for the instance can be calculated.

4 Difficulty Measure Applied to Angry Birds

This section presents each component of Figure 3 in detail by considering the domain of Angry Birds. Novelties in Angry Birds can appear in any game object. When explaining our difficulty measure formulation, we do not consider the novelties that appear in the birds' game object, as such novelties can usually be identified directly after a single shot by observing birds' behavior.

The first input is the initial state of the instance without any interaction. In our example domain, this is the game instance before shooting any birds. To represent the game scene, we use a 2D coordinate system where the *x*-axis is horizontal and oriented to the right while the *y*-axis is vertical and oriented to the top (Figure 4). P denotes all pixel points in a scene. For a pixel p_i, $x(p_i)$ and $y(p_i)$ denote its x and y coordinates. O represents all objects in the environment. Each object o_j (s.t. $o_j \in O$) comprises a set of pixels that can be mapped to a specific object (e.g. square wood).

The second input is the novelty present in the instance. In our example domain, novelties may appear in different object categories (i.e., wood, ice, stone, pigs) and the novel property could be any physics parameter (e.g. mass, friction, bounciness, etc). Thus, an example of the input is *(stone blocks, mass)*. These inputs are sent to the *target determining module* to search for possible target objects.

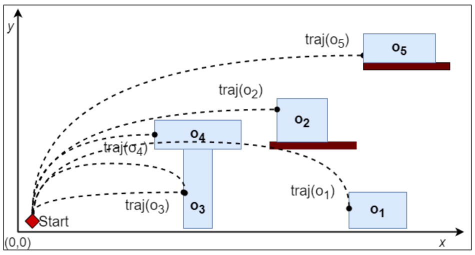


Figure 4: Representation of the object space. o_2, o_3, o_4 and o_5 satisfy the *left-of* relation to o_1. The trajectories to each object are denoted by the dotted line. A dot in the line represents a pixel point $p_i \in P$. o_2, o_3, o_4, and o_5 satisfy the *target* predicate. o_1 is not a target as the *traj*(o_1) intersects with o_4, which is in left to o_1. o_3 supports o_4. If o_3 moves, o_4 also moves: *impacted*(o_3,o_4) is true.

Target Determining Module

This module is used to identify the target objects. We consider the target objects as the objects that are directly reachable to interact. We do not consider platforms as target objects as they are static. We use the following predicates to determine the targets in our example domain.

- *left-of* (o_i, o_j): if object o_j is in left of object o_i (Figure 4). *left-of*(o_i, o_j) $\equiv o_i \neq o_j \wedge x_{max}(o_i) > x_{min}(o_j)$,
  where: $x_{max}(o_i)$ and $x_{min}(o_j)$ are the maximum pixel coordinate of object o_i in *x* direction and minimum pixel coordinate of o_j in *x* direction respectively.
  $x_{max}(o_i) = max(x(p_k) \; \forall \; p_k \in o_i)$,
  $x_{min}(o_j) = min(x(p_k) \; \forall \; p_k \in o_j)$
- *traj*(o_i): trajectory from a starting point to object o_i.
  As shown in Figure 4 for object o_3, a trajectory may contain multiple connections starting from a fixed position (slingshot in Angry Birds) to the connection point of the object. The connections can be represented using a set of points denoted by the dotted lines in Figure 4. We define:
  traj(o_i) = $\{t^i_1, t^i_2, ..., t^i_n\}$
  where, $t^i_k = \{p_{1k}, p_{2k}, ..., p_{nk}\}$
  i.e., a set of points that belong to one of the parabola trajectories and only $p_{n\cdot} \in o_i$.
- *target*(o_i): if object o_i is a target object.
  o_i is a target if the object is directly reachable, i.e., there is at least one trajectory to o_i such that trajectory points do not intersect with any object with *left-of* relation to o_i according to our domain.
  target(o_i) $\equiv (\exists \; t^i \in traj(o_i)) \wedge t^i \notin o_j \; \forall \; \{o_j : left-of(o_i, o_j) \; \forall o_j \in O\}$

Similar to the above *left-of* relation, we can define *right-of, below,* or *above* relations according to the requirement in each domain. We can also define *traj*(o_i) and *target*(o_i) specific to each domain. For example, if the way to interact with the objects is to drop an object from above, *traj*(o_i) should be defined according to the path taken by the object in gravitational free fall and *target*(o_i) is determined by the trajectories that do not intersect with the objects in *above* relation to *target*(o_i) $\equiv (\exists \; t^i \in traj(o_i)) \wedge t^i \notin o_j \; \forall \; \{o_j : above(o_i, o_j) \; \forall o_j \in O\}$

Object Dynamics Reasoning module

After target objects are determined, it is necessary to identify the objects that can be moved due to the interactions with the target objects. This is achieved by using the *object movement analysis* component. We instantiate all components using our proposed qualitative physics algorithms. If the novel objects are among the impacted objects identified (defined below) or the target objects, the *detectability analysis* component captures if the novel objects move in a detectable way. We first define the following to aid the explanations of the methods used in the two components.

- *novel-object*(o_i): if object o_i is a novel object. As defined earlier, o_i is a novel object if it violates the predefined mapping between appearance and physical parameters. i.e., object has changed physical parameter values.
- *impacted*(o_i, o_j): if o_j is moved due to the interaction of a bird with the target object o_i. For example, if o_i supports o_j and o_i is hit by a bird, o_j also moves (See o_3 and o_4 in Figure 4. The reasoning for the identification of such objects is presented under *object movement analysis*).
- *detectable*(o_i, o_j): if o_j moves in a detectable way due to the interaction of a bird with the target object o_i. detectable(o_i, o_j) returns true when o_j is a novel object and *impacted*(o_i, o_j) is true and o_j is impacted by the target object o_i in a detectable way. A case-based exploration of the detectability is conducted in *detectability analysis*.

Object Movement Analysis This section presents the qualitative physics approach used in identifying the objects that satisfy the *impacted* predicate presented above. i.e., we identify which objects move after an interaction with a target object. We use two algorithms 1) based on the stability, 2) based on the force propagation in the horizontal direction (Algorithm 1). We used the algorithm *vertical impact* proposed by Walega, Zawidzki, and Lechowski to reason about the stability of the objects. We also propose a new algorithm, *approximate horizontal influence* to check the impact on the objects located in the horizontal direction.

Vertical impact: This algorithm recursively checks the objects in a structure starting from the object that is directly impacted and returns a list of objects that may fall.

It exploits the rule which is the basis for stability investigation, "an object is stable if the vertical projection of the centre of mass of the object falls into the area of support base" (Zhang and Renz 2014). The algorithm contains eight steps where at each step object relationships are examined and substructures are constructed. The stability of objects is examined by approximating the center of mass of substructures and their supporters. A clear explanation of the algorithm is available in the work of Walega, Zawidzki, and Lechowski and is diagrammatically summarized in the extended version of our paper (Pinto et al. 2023). At the end of the eight steps, the algorithm returns the list of objects that may fall after the interaction with a target object.

Approximate horizontal influence: This algorithm examines the impact a target object can cause due to the force propagation on the objects located horizontally to the target.

We start by analysing if the target object can get destroyed due to the interaction. If it is not destroyed, we check if the object will slide or it will flip as a result of the interaction (collision). Destruction of the target object heavily depends on the materials and the types of the two colliding objects and the velocity at the collision. In our example domain, we define the following predicate by considering the object materials (e.g., wood, ice, stone, pig) and the bird (e.g., red, blue, yellow). We approximate the velocity at the collision by using the law of energy conservation.

object-destroy$(o_i) \equiv o_i^{life}$ *– damage* < 0. o_i^{life} is the object life and it depends on the material of the object and type of it (e.g. square wood-block, rectangular ice-block). This is a constant value for a specific object. *damage* depends on the type of the bird used and the relative velocity at collision. Damage caused by a bird type is a fixed value for a specific object, $o_i^{bird\_damage}$. *damage* can be approximated as $o_i^{bird\_damage}$ × *relative-velocity* at collision. *relative-velocity* can be approximated using the law of energy conservation. Thus, the final formulae for the *object-destroy*(o_i) predicate can be given as, *object-destroy*$(o_i) \equiv$ $(o_i^{life} - o_i^{bird\_damage}\sqrt{k1 \times (y_{start} - y_{target}) + k2_{bird}}) < 0$

where, $k1$ is an experimentally fixed constant value, and $k2_{bird}$ is a value based on the initial kinetic energy of the bird (In Angry Birds, the value only depends on the bird mass as the initial launch velocity is fixed because agents use the slingshot with full stretch). $(y_{start} - y_{target})$ is the height difference between slingshot and the target object.

If the *object-destroy*(o_i) predicate is false, we check the *object-flip*(o_i) predicate by considering object dimensions.

object-flip$(o_i) \equiv \frac{y_{max}(o_i) - y_{min}(o_i)}{x_{max}(o_i) - x_{min}(o_i)} > k_{flip}$,

where: $y_{max}(o_i)$ and $y_{min}(o_i)$ are the maximum pixel coordinate of object o_i in *y* direction and minimum pixel coordinate of o_j in *y* direction respectively. The k_{flip} is an experimentally fixed constant value.

k_{flip}= *flipping threshold*,
$y_{max}(o_i)$ = *max* $(y(p_j)\ \forall\ p_j \in o_i)$,
$y_{min}(o_i)$ = *min* $(y(p_j)\ \forall\ p_j \in o_i)$,
and $x_{max}(o_i)$, $x_{min}(o_i)$ are as defined previously.

These predicates hold the basis for the *approximate horizontal influence* algorithm. A pseudo-code of the process is demonstrated in Algorithm 1 and Figure 5 explains the terms *falling-arc*(o_i) and *sliding-path*(o_i) used in Algorithm 1.

- For a circle C, with centre $(x_{max}(o_i), y_{min}(o_i))$ and radius $(y_{max}(o_i) - y_{min}(o_i))$, let $q1$ be the set of pixel points in the first quadrant of C. *falling-arc*(o_i) returns the list of objects within the falling arc of object o_i (See Figure 5a). We define *falling-arc*(o_i) as follows:
  falling-arc$(o_i) \equiv \{o_j \in O \mid o_j \neq o_i \wedge (o_j \cap q1) \forall o_j \in O\}$
- *sliding-path*(o_i) returns the list of objects within the path the object o_i slides (See Figure 5b). We define *sliding-path*(o_i) as follows:
  sliding-path$(o_i) \equiv \{\ o_j \in O \mid o_i \neq o_j$
  $\wedge\ (x_{max}(o_i) < x_{min}(o_j) < x_{max}(o_i) + k_{sliding\_constant})$
  $\wedge\ ((y_{min}(o_i) < y_{max}(o_j) < y_{max}(o_i)) \vee (y_{min}(o_i) < y_{min}(o_j) < y_{max}(o_i)))\ \forall o_j \in O\}$
  where, $k_{sliding\_constant}$ is an experimentally determined distance that approximates the distance an object can slide after a collision.

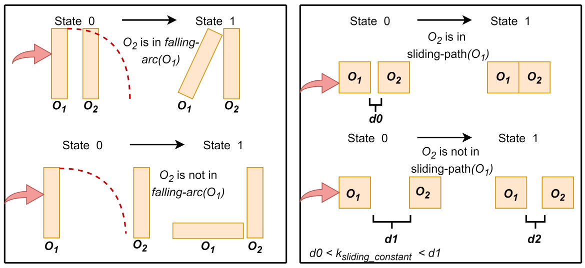


Figure 5: Left Figure (a) shows examples for *falling-arc*(o_1) and the right Figure (b) shows examples for *sliding-path*(o_1)

Algorithm 1: Approximate horizontal influence

Input: State representation of objects, target object (o_i)
Output: List of impacted objects

1: Initialize *horizontal-propagation(HP)_impact_list*
2: **if** ¬ (*object-destroy*(o_i)) **then**
3:    **if** *object-flip*(o_i) **then**
4:       *pending_list* = *falling-arc*(o_i)
5:    **else**
6:       *pending_list* = *sliding-path*(o_i)
7:    **end if**
8:    *closest_object* = $o_j \mid min(x_{min}(o_j) - x_{max}(o_i)\ \forall\ o_j \in$ *pending_list*$)$
9:    Add *vertical impact*(*closest-object*) to *HP_impact_list*
10: **end if**
11: **return** *HP_impact_list*

In Algorithm 1 (line 8), we only limit to a single *closest_object* obtained from either the *falling-arc* or *sliding-path* according to the experimentation with our example domain. However, this can be altered according to the domain. The output of the *object movement analysis* module is the list of impacted objects obtained from the *vertical impact* algorithm and the *approximate horizontal influence* algorithm.

Detectability Analysis This section presents the case-based exploration in identifying the *detectable* predicate shown above. Once the set of impacted objects is available, we can categorize each object into at least one of the below cases that represent observable features in a physical world. The observable movement of the directly-hit object (i.e., target object) can be explained using the first three cases.

- Case 1: Directly hit and destroys
- Case 2: Directly hit and flips
- Case 3: Directly hit and slides

Apart from these three special cases, all objects subject to at least one of the following six cases. Case 4 and 5 focus on object rotation. We assumed that rotation of the impacted objects directly above and very close to static structures (ground or a platform) is hardly observable. Other objects could rotate due to the collisions with objects and there is a chance of observing the rotation when objects fall.

- Case 4: Falls from the top without rotating
- Case 5: Falls from the top while rotating

Case 6 and 7 focus on the objects that slide. The object may slide and stop, or it might fall if it's located above the ground based on the sliding path.

- Case 6: Slide and stop
- Case 7: Slide and fall down

Case 8 and 9 focus on the objects which flip. Similar to the above two cases, it may either fall or stop based on location.

- Case 8: Flips and stop
- Case 9: Flips and fall down

The nine cases cover the majority of observable movements in Angry Birds (More details in (Pinto et al. 2023)).However, there could be situations that may be not captured using the nine cases. To evaluate if the novel object is detectable, we check if the object is moved in a detectable manner by considering the changed attribute along with the object type. Consider the following examples:
Example 1: Novelty in "friction" of stone blocks - If at least one impacted stone block satisfies the requirements for case 3, 6, or 7, we can detect the novelty (as friction changes can be observed when the object slides).
Example 2: Novelty in "bounciness" of wood objects - If at least one impacted wood object satisfies the requirements for case 2, 3, 4, 5, 6, 7, 8, or 9, we can detect the novelty (as bounciness can be observed when objects collide).

The output of this module enables to capture the objects that satisfy *detectable* predicate for each target object.

Difficulty Computation Module

This component quantifies the difficulty of detecting novelty for each game instance. We propose two algorithms to calculate the detection difficulty. Factors including the novelty in the object, the placement of the objects, the number of detectable objects, the number of reachable objects, and the number of interactions available (number of birds in Angry Birds) are considered when developing both methods.

We define the following to identify the most influential target object to interact with (i.e., the target object that gives the most information about objects movements. We refer to this as the *best-target*).

- *impact-score*(o_i): The heuristic impact score of *target*(o_i) is defined based on the objects moved and the novelty.

  Example 1: If the novelty is in only one object in the instance, the *score* per each object moved = 1

  Example 2: If the novelty is in objects with the same material (wood, ice, stone), the *score* per material moved=1

  Example 3: If the novelty is in object types and if the player is informed that the wood objects are not novel, the *score* per each wood object moved = 0, the *score* per other types of objects moved = 1

  impact-score(o_i) = $\sum_{o_j \in O | impacted(o_i, o_j)} score_{o_j}$

- *best-target*: The target object with the highest *impact-score*. If there are multiple objects with the same impact-score, the first object from all objects is selected.

  best-target $\equiv o_i \mid$ *target*(o_i) $\wedge$ *impact-score*(o_i) = *max(impact-score*(o_i)*)* $\forall\, o_i \in O$

Algorithm 2: Probabilistic interaction difficulty

```
Input: State representation of objects (O) Output: PID
Initialize PID = 0
for i in total_number_of_interactions do
    N_i = | { o_j ∈ O | target(o_j) ∀ o_j ∈ O } |
    n_i = | { o_j ∈ O | (target(o_j) ∧ ∃ o_k ∈ O s.t. novel-
        object(o_k) ∧ detectable(o_j, o_k)) ∀ o_j, o_k ∈ O } |
    M_i = (N_i − n_i) / N_i
    PID += M_i
    if M_i ≠ 1 then
        break
    else
        Shoot at the best-target
        Update state of objects
    end if
end for
PID = PID / total_number_of_interactions
return PID
```

Probabilistic interaction difficulty (PID) Algorithm 2 is based on the intuition that the probability of novelty detection depends on the number of novel objects available. Intuitively, if the probability of finding a target that impacts the novel object in a detectable way is lower, the difficulty is higher. *PID* is initialized at zero, and the algorithm loops over the number of possible interactions (i.e., number of birds available in Angry Birds) while updating the *PID*. To proceed to the next interaction, it is assumed that the agent shoots the best-target and the objects in the environment are updated along with the search space (which objects to explore next). The terms, N_i is the total number of target objects and n_i represents the total number of target objects which makes the novel object move in a detectable way in the given state. Thus, M_i is the proportion of targets that do not yield a detectable movement. At the end of the computation, *PID* is normalized to [0,1] (1 indicates the highest difficulty, and *PID* is unitless). One limitation of this algorithm is that it only considers the *best-target* when updating the next state instead of considering all possible targets. This is due to time restrictions and works under the assumption that an intelligent agent would always select the *best-target*.

Best-shot interaction difficulty (BID) Algorithm 3 is inspired by an intelligent human-like agent and is based on the interaction which yields the most information. Here we try to maximize the chance of novelty detection by making the most influential interaction (i.e., always shooting at the *best-target:* o_k*). The algorithm loops over the number of possible interactions that can be made: if the novelty is undetectable by shooting at the best-target, it proceeds after updating the environment, the search space (which objects to explore next), and *BID*. Similar to Algorithm 2, *BID* is normalized to [0,1], where 1 indicates the highest difficulty and is unitless.

These two difficulty algorithms can be used separately or collectively according to the suitability of the study.

5 Experimental Evaluation

As the difficulty measures we proposed is general, we examined the relationship between our proposed difficulty measure and human perception. We conducted an experiment approved by the Australian National University hu-

Algorithm 3: Best-shot based interaction difficulty

Input: State representation of objects **Output**: *BID*

Initialize *BID = 0*
Initialize *detection_flag = False*
for i in *total_number_of_interactions* **do**
    BID = BID + 1
    if $(\exists o_j \in O \mid novel - object(o_j) \wedge detectable(o_k*, o_j))$ ***then***
        detection_flag = True
        break
    else
        Shoot at the best-target
        Update state of objects
    end if
end for
if *detection_flag = False* **then**
    BID = total_number_of_interactions + 1
end if
BID = (BID - 1) / total_number_of_interactions
return *BID*

man ethics committee (protocol-2020/717). We gathered data from 20 voluntary participants (aged 20-35, including males and females) with no prior knowledge of the tested novelties. Participants played 10 instances without novelty (generated from Angry Birds levels generator (Stephenson and Renz 2017)) to familiarize themselves with the game physics and dynamics. Then, they played 15 instances, each featuring one of three different novelties. We measured the difficulty of detecting each novelty using our proposed approach. Each participant was allowed to play the novel instance only once to detect if there is any novelty in the game objects. If the novelty was detected, we recorded the number of interactions (number of shots) the participant used to detect that novelty. We requested the participant to provide a simple description of the observation to validate the results. Each participant took approximately 40-50 minutes to complete the experiment. The novelties we generated are:

- **Type 1 (T1):** The parameter *gravity scale* of pigs is decreased twice the original value. Pigs fall down slower due to this novelty.
- **Type 2 (T2):** The parameter *bounciness* of wood objects is increased by four times the original value. This makes the wood objects bouncier.
- **Type 3 (T3):** The parameter *life* of stone objects increased by five times. This makes stone blocks difficult to destroy.

Game Instance Selection A set of 100 game instances was generated from the state-of-the-art (SOTA) level generator (Stephenson and Renz 2017) and the novelty game instances were created for each novelty type. We then computed difficulty using the two algorithms, *PID* and *BID* for each instance. We combined the two values: *Difficulty Value* = α*PID* + (1-α)*BID*, where $\alpha \in [0,1]$, can be adjusted based on the importance of the two algorithms in an experiment. In our experiment, we got $\alpha = 0.5$ to give equal importance. Game instances within each novelty type were then classified into three categories: *easy (e), medium (m), hard (h)*. Game instances with values < the value at 33.33% percentile, 33.33% - 66.67%, and values > 66.67% were considered as *e*, *m*, and *h* instances respectively. The game

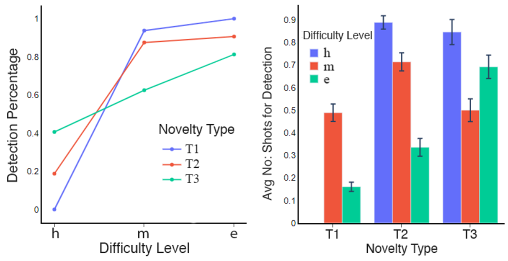


Figure 6: Experiment results from human participants. The left figure (a) shows the percentage of novelty detection and the right figure (b) shows the average normalized number of shots for novelty detection for each difficulty level. Error bars represent the standard error. *e,m,h* indicate easy, medium, and hard categories.

instances used for the experiment were selected randomly from each category. However, techniques such as harmonic mean/clustering methods could also be utilized to categorize based on the data available.

Results According to our difficulty measure, we expect the percentage of novelty detection to decrease in the order *e, m*, and *h* (Algorithm 2). Ideally, if the novelty is detected, we expect a lower number of interactions to detect the novelty in the category *e* and a higher number of interactions in the category *h* (Algorithm 3). Figure 6a illustrates the percentage of human participants who correctly detected the novelty for each novelty type in the three difficulty levels. In line with our hypothesis, the lowest percentage of detection is recorded in the category *h* and the highest is recorded in the category *e*. This observation is consistent for all three experimented novelty types. For the T1 novelty type, none of the participants were able to detect the novelty in the category *h*, while all the participants detected it in category *e*.

Figure 6b summarizes the average normalized number of shots needed for detection for each difficulty level for the three novelty types. That is, for each participant, the number of shots taken for detection is normalized by the total number of possible interactions (i.e., the number of birds in the game instance). For novelty type T1, the category *h* is not presented as none of the participants detected the T1 novelty type. The *m* and *e* categories follow our expectation by producing a lower value for the category *e*. Similarly, T2 results are also consistent with our expectation. For T3, while the category *h* gives the highest normalized interactions for detection, the category *m* is lower than the category *e*. According to our observations, some participants used more shots to confirm that stone-blocks have a higher health value even though they already detected this novelty earlier and some participants did not notice the change in stone-blocks at all. Overall, the difficulty of novelty detection for human participants falls in line with the calculated difficulty values.

6 Discussion and Conclusion

Detecting novelty is an important capability for an intelligent system in an open-world environment. In real-world situations, an agent needs to reason about physics in order to detect novel objects with different physical parameters. These novelties often vary in their difficulty of detection and have not been studied before this paper. However, understanding this difficulty can be an important aspect of conducting a robust and fair evaluation. Thus, we have proposed a method to quantify the difficulty of novelty detection using qualitative physics. Our method is agent-independent and can be used to make more accurate conclusions about the detection capabilities of different agents. This measure was applied in the Angry Birds domain, and validated by comparing the results of the proposed measure with the performance of human participants. To define the physical reasoning predicates, we have used quantitative thresholds based on domain knowledge.

The different components and algorithms that were introduced in this paper can also be applied to other research problems. When formulating our novelty difficulty measure, we proposed the algorithm *approximate horizontal influence* that could also be used as a component for agents to predict the influence of moving an object. This is an improvement to the prior work (Zhang and Renz 2014; Walega, Zawidzki, and Lechowski 2016) as it considers objects that are disconnected in the horizontal direction. Our difficulty formulation can also be used to create novel game instances at a predefined difficulty of novelty detection. It can be used as a component in the SOTA novelty generation framework for Angry Birds (Gamage et al. 2021) to generate novel game instances with a predefined difficulty. This facilitates research in open-world learning agent development by creating different instances with different levels of difficulty.

We plan to extend our study to address limitations such as generalizing our presented qualitative reasoning algorithms in *object movement analysis* to other domains. Moreover, we have discussed how the difficulty formulation can be applied to PHYRE (Bakhtin et al. 2019) in the extended version of this paper (Pinto et al. 2023) and we plan to extend the framework to suit a wider variety of novelties and be applicable to a wider range of domains. In this paper, we laid a foundation for quantifying the difficulty of novelty detection that aids to conduct a sound open-world evaluation.

Acknowledgments

This research was sponsored by the Defense Advanced Research Projects Agency (DARPA) SAIL-ON program.

References

Allen, J. F. 1983. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26(11): 832–843.

Bakhtin, A.; van der Maaten, L.; Johnson, J.; Gustafson, L.; and Girshick, R. B. 2019. PHYRE: A New Benchmark for Physical Reasoning. In *NeurIPS*.

Boult, T.; Grabowicz, P. A.; Prijatelj, D.; Stern, R.; Holder, L.; Alspector, J.; Jafarzadeh, M.; Ahmad, T.; Dhamija, A. R.; Cli; Cruz, S.; Shrivastava, A.; Vondrick, C.; and Scheirer, W. 2021. A Unifying Framework for Formal Theories of Novelty: Framework, Examples and Discussion. In *AAAI*.

Chollet, F. 2019. On the Measure of Intelligence. *arXiv e-prints*, arXiv:1911.01547.

Ferreira, L.; and Toledo, C. 2014. A search-based approach for generating Angry Birds levels. In *2014 IEEE Conference on Computational Intelligence and Games*, 1–8.

Franco, J. P.; Yadav, N.; Bossaerts, P.; and Murawski, C. 2018. Where the really hard choices are: A general framework to quantify decision difficulty. *bioRxiv*.

Gamage, C.; Pinto, V.; Xue, C.; Stephenson, M.; Zhang, P.; and Renz, J. 2021. Novelty Generation Framework for AI Agents in Angry Birds Style Physics Games. In *COG*.

Ge, X.; Renz, J.; and Zhang, P. 2016. Visual Detection of Unknown Objects in Video Games Using Qualitative Stability Analysis. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(2): 166–177.

Giesel, M.; and Gegenfurtner, K. R. 2010. Color appearance of real objects varying in material, hue, and shape. *Journal of Vision*, 10(9): 10–10.

Langley, P. 2020. Open-World Learning for Radically Autonomous Agents. In *AAAI*.

Laubert, C.; and Parlamis, J. 2019. Are You Angry (Happy, Sad) or Aren't You? Emotion Detection Difficulty in Email Negotiation. *Group Decision and Negotiation*, 28: 377–413.

Perner, P. 2018. Determining the Similarity Between Two Arbitrary 2D Shapes and Its Application to Biological Objects. *IJCSE*.

Pinto, V.; Xue, C.; Gamage, C. N.; Stephenson, M.; and Renz, J. 2023. The Difficulty of Novelty Detection in Open-World Physical Domains: An Application to Angry Birds. *arXiv: 2106.08670*.

Renz, J.; Ge, X.; Gould, S.; and Zhang, P. 2015. The Angry Birds AI Competition. *AI Magazine*, 36: 85–87.

Renz, J.; Ge, X.; Stephenson, M.; and Zhang, P. 2019. AI meets Angry Birds. *Nature Machine Intelligence*, 1.

SAIL-ON-BBA. 2019. Broad Agency Announcement, Science of Artificial Intelligence and Learning for Open-world Novelty (SAIL-ON).

Senator, T. 2019. Science of Artificial Intelligence and Learning for Open-world Novelty (SAIL-ON).

Stephenson, M.; and Renz, J. 2017. Generating varied, stable and solvable levels for Angry Birds style physics games. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, 288–295.

Stephenson, M.; Renz, J.; Ge, X.; Ferreira, L.; Togelius, J.; and Zhang, P. 2019. The 2017 AIBirds Level Generation Competition. *IEEE Transactions on Games*, 275–284.

Takatalo, J.; Häkkinen, J.; Kaistinen, J.; and Nyman, G. 2010. Presence, Involvement, and Flow in Digital Games. *Evaluating User Experience in Games*, 23–46.

Walega, P. A.; Zawidzki, M.; and Lechowski, T. 2016. Qualitative Physics in Angry Birds. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(02): 152–165.

Yom-Tov, E.; Fine, S.; Carmel, D.; and Darlow, A. 2005. Learning to estimate query difficulty: Including applications to missing content detection and distributed information retrieval. *Proc. ACM SIGIR*, 512–519.

Zhang, P.; and Renz, J. 2014. Qualitative Spatial Representation and Reasoning in Angry Birds: The Extended Rectangle Algebra. In *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'14, 378–387. AAAI Press.

Describing the Characteristics of Circular and Elliptical Motion using Qualitative Representations

Marco Kragten[1] **and Bert Bredeweg**[1,2]
[1]Faculty of Education, Amsterdam University of Applied Sciences, Amsterdam, The Netherlands
[2]Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
{m.kragten, b.bredeweg}@hva.nl

Abstract

Circular and elliptical motion are fundamental topics in physics education, yet learners often struggle to grasp them. We investigate how interactive qualitative representations can be used to describe the characteristic behavior of circular and elliptical motion. We use the vocabulary and algorithms known as qualitative reasoning, which make it possible to represent the distinct features of these systems in a conceptual way. Leveraging the close alignment between qualitative reasoning and human reasoning about dynamic systems, these representations have the potential to enhance understanding in this domain.

1 Introduction

Circular motion is a fundamental concept in physics that describes the motion of an object moving in a circular path. The direction of velocity (but not the speed) of an object in circular motion changes due to the centripetal force which causes centripetal acceleration. Centripetal force is directed towards the center of the circle. In the case that an object is orbiting another object (e.g., a planet orbiting a star) centripetal force is equal to gravitational force.

Celestial bodies generally follow elliptical orbits, although circular orbits are often used as a simplified approximation for easier understanding. Additionally, certain celestial bodies, like moons, exhibit nearly circular orbits around their parent planets. The elliptical motion of celestial bodies is governed by Kepler's laws of planetary motion, which can be explained by the gravitational forces exerted between celestial bodies. The strength of gravity depends on the distance between the bodies. As the distance changes within an elliptical orbit, gravity varies, resulting in different acceleration at different points along the orbit. The elliptical shape of the orbits arises from the balance between the gravitational force and the momentum (the product of the mass and velocity) of the object in motion.

In physics education, circular and elliptical motion is often explained on the basis of mathematical formulas. Learners then work through exercises involving calculations using these formulas to process and learn this knowledge. The use of supporting software is limited. Particularly, the conceptual knowledge that explains the working of the mechanisms is not available in an interactive format. This issue poses a challenge in physics education, as there have been numerous reported difficulties associated with understanding circular and elliptical motion [e.g., Alonzo & Steedle, 2009; Barniol & Zavala, 2014; Canlas, 2016; Liu & Fang, 2016].

In this paper we focus on describing circular and elliptical motion using interactive qualitative representations [Bredeweg *et al.*, 2023a]. For the work presented in this contribution we use the software Dynalearn [Bredeweg *et al.*, 2013]. This software is implemented as a server-based architecture deploying the Garp3 reasoning engine [Bredeweg *et al.*, 2009]. The front-end is web-based and provides a diagrammatic approach for users to construct and articulate their thoughts. Learning through the construction of qualitative representations has proven to be a successful approach [Bredeweg *et al.*, 2023a; Kragten & Bredeweg, 2023], highlighting the potential of the representations described in this contribution to enhance understanding.

2 Circular motion

To represent circular motion qualitatively, the following notions have to be addressed: entities, quantities, possible values and direction of change, causal dependences, correspondences, and finally simulation consisting of qualitatively distinct states and transitions between them.

2.1 Direction of change and values of quantities

Entities represent the physical objects that constitute the system. Let's assume we model a moon orbiting a planet. In that case, the qualitative representation will have two entities: Moon and Planet. **Quantities** represent the measurable properties of entities, as such, the entity Moon has a position, a velocity, etc.

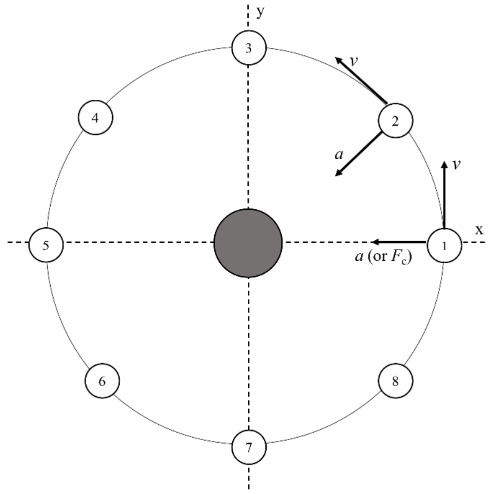


Fig. 1. Circular motion of a moon orbiting a planet. A system manifesting circular motion has eight qualitatively distinct states.

In a qualitative representation, each quantity has a value and a **direction of change**, represented as a tuple <v, ∂>. The possible values are represented in a quantity space, also for ∂. For instance, the direction of change can be captured by {−, 0, +}, referring to decreasing, steady, and increasing, respectively. However, the exact meaning of this depends on the context. To represent the dynamics of circular motion, we project the system on an x- and y-coordinate plane (Fig. 1). With regard to *Position*, ∂=+ is used to refer to 'increasing' on the x−axis (moving to the right) or on the y-axis (moving upward), while ∂=− refers to decreasing on these axes, and ∂=0 refers to remaining steady (no movement).

A similar quantity space can be used for the **possible values**, namely {min, −, 0, +, max}. If we consider *Position*, then 'min' refers to most-negative point on the x-axis (or y-axis), '−' refers to a negative interval between 'min' and '0', '0' refers to the origin of the plane, '+' refers to a positive interval between '0' and 'max', and 'max' refers to the most-positive point on the x-axis (or y-axis). Notice that, 'min', '0' and 'max' are points, while '−' and '+' are intervals. It turns out that the extreme values 'min' and 'max' are not needed for representing all the possible behaviors. This is because if the direction of change is zero within the negative and positive intervals, i.e., <−, 0> and <+, 0>, they also represent the minimum or maximum. Hence, we leave them out and work with the quantity space {−, 0, +}. Also note that, the planet is located at the origin of the coordinate plane.

2.2 Expected qualitative states

In a qualitative representation, each qualitatively distinct behavior of the system is represented as a **state**. Consequently, each state has a unique set of tuples <v, ∂> for the quantities describing the system. Given that the system is projected on a coordinate plane, the horizontal and vertical position, centripetal force, acceleration and velocity are the characteristic quantities. Together they describe the system using eight qualitatively distinct states (Fig. 1).

Table 1 shows the values and directions of change for each of the quantities in the eight states. Consider the position of the moon in state 1, in which case x=<+, 0> and y=<0, +>. The moon is at its most-right position (somewhere in the positive interval, hence '+') and there is no further change in the horizontal direction, hence ∂x=0. The y-coordinate is '0', but the moon is in an upward motion so there is a positive change in the vertical direction, hence ∂y=+.

In state 1, the centripetal force (F_c) and thereby the acceleration (a) is directed to the left. To describe the change of velocity we decompose the vectors of acceleration (and velocity) into a horizontal (a_x) and vertical component (a_y). For the horizontal acceleration holds a_y=<−, 0>, which represents that a_y is at its most-negative value (the vector is directed to the left at its maximum value) and momentarily steady (for an infinite small moment). There is no vertical acceleration but there is a negative direction of change, hence a_y=<0, −>. There is no horizontal velocity and the change is negative, thus v_x=<0, −>. The vertical acceleration is at its maximum, thus v_y=<+, 0>.

Table 1. Eight qualitative states of circular motion. Quantities are position: x-axis (x) and y-axis (y), acceleration: horizontal (a_x) and vertical (a_y), and velocity: horizontal (v_x) and vertical (v_y). Each quantity has a value and a direction of change, shown as <v, ∂>. Force corresponds to acceleration. Force is not shown in this table.

	State							
Quantity	1	2	3	4	5	6	7	8
x	<+, 0>	<+, −>	<0, −>	<−, −>	<−, 0>	<−, +>	<0, +>	<+, +>
y	<0, +>	<+, +>	<+, 0>	<+, −>	<0, −>	<−, −>	<−, 0>	<−, +>
a_x	<−, 0>	<−, +>	<0, +>	<+, +>	<+, 0>	<+, −>	<0, −>	<−, −>
a_y	<0, −>	<−, −>	<−, 0>	<−, +>	<0, +>	<+, +>	<+, 0>	<+, −>
v_x	<0, −>	<−, −>	<−, 0>	<−, +>	<0, +>	<+, +>	<+, 0>	<+, −>
v_y	<+, 0>	<+, −>	<0, −>	<−, −>	<−, 0>	<−, +>	<0, +>	<+, +>

Note that state 1 is a point. The quantities only have these values and directions of change at this specific x- and y-coordinate in the system. In fact, state 1 has an infinite small duration. The system instantaneously moves into state 2, which has a duration. The values and directions of change in state 2 are true for the interval between state 1 and 3. State 3, 5 and 7 are also points. State 2, 4, 6 and 8 are intervals (with duration).

2.3 Adding dynamics to the representation

The next challenge is to add dynamics to the qualitative representation so that the latter can be simulated and successive states calculated from the information in the preceding states. Let us focus on the motion in horizontal direction. The implementation of this part is shown in Fig. 2. As discussed before, the entity Moon has four quantities to represent this part of the behavior: x, F_x, a_x and v_x. All quantities have the quantity space {−, 0, +}. The direction of change is donated with ∂.

Two types of **causal dependencies** are distinguished: *proportionality* and *influence* [Bredeweg *et al.*, 2013]. When two quantities have a proportional relationship (P), a change in one quantity (the cause) results in a change in the other quantity. A proportional relationship can be positive (P+), where both quantities change in the same direction, or negative (P−), where the quantities change in the opposite direction. The relationship between the quantities x and F_x is negative proportional (P−). Note that F_x is the horizontal component of the centripetal force which in this system is equal to the gravitational force, i.e., if the moon moves towards the origin of the coordinate plane (the location of the planet) the gravitational pull in the horizontal direction decreases (but increases in the vertical direction). The relationship between the quantities F_x and a_x is positive proportional (P+). This denotes that acceleration changes when the force applies changes.

Causal dependencies of type influence (I+, I−) can be added to represent the relationship between a process (also represented as a quantity) and another quantity. A process adds or removes something to the system per time unit. If an influence is positive (I+), a positive value of the process results in a change in the positive direction of the affected quantity, a negative value results in a change in the negative direction. The relationship between a_x and v_x is of the type positive influence (I+) (if a_x=− then δv_x=−, if a_x=0 then δv_x=0 and if a_x=+ then δv_x=+). For instance, if the acceleration in the horizontal direction is ‘0’ than there is no change in velocity. The relationship between v_x and x is also a positive influence (I+) (if v_x=− then δx=−, if v_x=0 then δx=0 and if v_x=+ then δx=+). For instance, if the velocity in the horizontal direction is negative ‘−‘, than the moon moves towards the negative side of the x-axis in the coordinate plane.

To determine the potential states of the system, **correspondences** (C) can be incorporated to describe the relationship between co-occurring values. In the present system, the values of x and F_x are dependent, they correspond inversely (if x=− then F_x=+, if x=0 then F_x=0 and if x=+ then F_x=−). The values of F_x and a_x are also dependent, they correspond regularly (if F_x=− then a_x=−, if F_x=0 then a_x=0 and if F_x=+ then a_x=+). The correspondences between x and F_x, as well as F_x and a_x, are directed, suggesting one-way dependencies between the values. To represent these directed correspondences, an arrow pointing in one direction is used (Fig. 2).

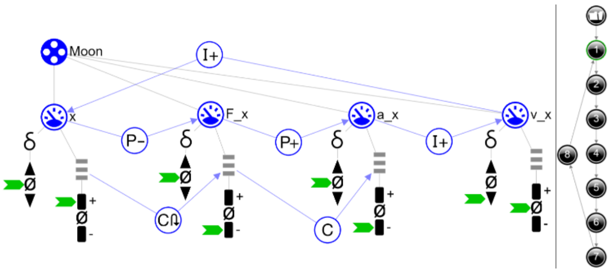


Fig. 2. Qualitative representation of the motion of a moon in horizontal direction. Quantities are position (x), force (F_x), acceleration (a_x) and velocity (v_x) (in the text we use F_x, a_x and v_x). The representation is simulated with initial settings: <+, ?> and velocity <0,?> (not shown in the figure; ? refers to undefined). The simulation generates 8 states, as show on the RHS in the figure. The simulation result of state 1 is shown. From the representation it can be inferred that: x=<+, 0>, F_x=<−, 0>, a_x=<−, 0> and v_x=<0, −> (show in green). Correspondences are represented by the symbol C.

2.4 Simulation of horizontal motion

Fig. 2 shows the simulation for the horizontal motion, as it can be computed from the details discussed so far. The initial settings for this simulation are: x=<+, ?> and v_x=<0, ?> (? refers to undefined). All the other information can be inferred from this. The state graph (Fig. 2, RHS) shows that the system has eight states. The simulation result of state 1 is shown.

In state 1, the moon has no horizontal velocity (v_x=0), as determined by the initial settings. The causal dependency between v_x and x is of type positive influence (I+) and therefore the horizontal position of the moon does not change (if v_x=0 then δx=0). This results in x=<+, 0>, indicating that x is at its maximum. There is an inversed correspondence between x and F_x, indicating that the horizontal gravitational force on the moon is to the left (if x=+ then F_x=−). The correspondence between F_x and a_x indicates that the moon its horizontal acceleration is also to the left (if F_x=− then a_x=−). There is a negative proportional relationship (P-) between x and F_x and a positive proportional relationship (P+) between F_x and a_x. The horizontal position of the moon does not change and as a result gravitational force in the horizontal direction does not change (if δx=0 then δF_x=0). Consequently, acceleration in the horizontal direction does not change (if δF_x=0 then δa_x=0). Therefore, in state 1,

$F_x=<-, 0>$ and $a_x <-, 0>$. Both quantities are maximal in the negative interval, i.e., both vectors (F_x and a_x) have their maximal value (or magnitude) and are directed to the left. The causal dependency between a_x and v_x is of type positive influence (I+). The horizontal acceleration is to the left and as a result the direction of change of the horizontal velocity is to the left (if $a_x=-$ then $\delta v_x=-$), i.e., $v_x=<0, ->$.

In state 2 (Table 2), the moon its velocity in the horizontal direction is to the left and increasing, i.e., $v_x=<-, ->$. As a result, the moon is on the right of the y-axis and moving towards the left, i.e., $x=<+, ->$. As the moon moves closer to the x-origin of the coordinate plane, the gravitational pull, and consequently, the acceleration in the horizontal direction towards the left, decreases, i.e., $F_x=<-, +>$ and $a_x <-, +>$.

The changes from state 2 propagate onwards, continuing until state 8. Upon reaching state 8, the values resemble those of the simulation's initial settings, initiating the repetition of circular motion.

2.5 Completing the model

Thus far we have managed to represent the movement of the celestial body in the horizontal direction. For this, it is important to see that the causal dependencies between quantities that describe vertical motion are similar to those of the horizontal direction. But how to represent the pendulum movement of the moon between its most-negative and most-positive position in the horizontal *and* vertical direction? Both pendulum movements have 8 possible states and without further information this results in 64 (8 x 8) possible states. For instance, the motion in the horizontal direction can go through all its 8 states while the motion in the vertical direction is still in its first state. An important insight is to realize that the pendulum movements in both directions are dependent.

Table 2 shows the correspondences between the values of the quantities in both directions when describing circular motion. All correspondences are bi-directional and apply to the entire quantity space. It is important to note that due to the bi-directional nature of correspondences, they also apply in the opposite direction. Table 2 includes six correspondences, namely between:

- x and a_x. When the moon is positioned on the left side of the y-axis, its acceleration in the horizontal direction is towards the right (if $x=-$ then $a_x=+$). If the moon crosses the y-axis, there is no acceleration in the horizontal direction (if $x=0$ then $a_x=0$). When the moon is located on the right side of the y-axis, its horizontal acceleration is towards the left (if $x=+$ then $a_x=-$).
- x and v_y. When the moon is positioned on the left side of the y-axis, its vertical velocity is downward (if $x=-$ then $v_y=-$). If the moon crosses the y-axis, there is no vertical velocity (if $x=0$ then $v_y=0$). When the moon is located on the right side of the y-axis, its vertical velocity is upward (if $x=+$ then $v_y=+$).
- a_x and v_y. When the moon its acceleration in the horizontal is directed towards the left, its vertical velocity is upward (if $a_x=-$ then $v_y=+$). If the moon has no acceleration in the horizontal direction, there is no vertical velocity (if $a_x=0$ then $v_y=0$). When the moon its acceleration in the horizontal direction is toward the right, its vertical velocity is downward ($a_x=+$ then $v_y=-$).
- y and a_y. When the moon is positioned below the x-axis, its acceleration in the vertical direction is upward (if $y=-$ then $a_y=+$). If the moon crosses the x-axis, there is no acceleration in the vertical direction (if $y=0$ then $a_y=0$). When the moon is located above x-axis, its vertical acceleration is downward (if $y=+$ then $a_y=-$).
- y and v_x. When the moon is positioned below the x-axis, its horizontal velocity is towards the right (if $y=-$ then $v_x=+$). If the moon crosses the x-axis, there is no horizontal velocity (if $y=0$ then $v_x=0$). When the moon is located above the x-axis, its horizontal velocity is towards the left (if $y=+$ then $v_y=-$).
- a_y and v_x. When the moon its acceleration in the vertical direction is downward, its horizontal velocity is to the left (if $a_y=-$ then $v_x=-$). If the moon has no acceleration in the vertical direction, there is no horizontal velocity (if $a_y=0$ then $v_x=0$). When the moon its acceleration in the vertical direction is upward, its horizontal velocity is to the right ($a_y=+$ then $v_x=+$).

Table 2. Correspondences between quantity spaces in circular motion. The correspondences establish the co-occurrence of values of quantities of the horizontal and vertical direction of circular motion.

	value	x	y	a_x	a_y	v_x	v_y
x	−			$+^*$			$-^*$
	0			0^*			0^*
	+			$-^*$			$+^*$
y	−				$+^*$	$+^*$	
	0				0^*	0^*	
	+				$-^*$	$-^*$	
a_x	−	$+^*$					$+^*$
	0	0^*					0^*
	+	$-^*$					$-^*$
a_y	−		$+^*$			$-^*$	
	0		0^*			0^*	
	+		$-^*$			$+^*$	
v_x	−		$+^*$		$-^*$		
	0		0^*		0^*		
	+		$-^*$		$+^*$		
v_y	−	$-^*$		$+^*$			
	0	0^*		0^*			
	+	$+^*$		$-^*$			

* bi-directional correspondence

We can now add correspondences between quantity spaces of the horizontal and vertical motion. We only add four bi-directional correspondences (indicated by an arrow point on both sides) to the qualitative representation (Fig. 3) because by adding the correspondence between x and v_y and v_y and a_x, the correspondence between x and a_x becomes redundant. The same logic applies to the correspondence between y and y_x after adding the correspondences between y and v_x and v_x. and a_y. Note that we could have discarded other correspondences (or added them all). We made the decision to include correspondences between quantities of both directions, as they explicitly communicate the interdependence of the pendulum movements.

2.5 Simulation of the complete model

The representation is now ready and can be simulated. The starting condition for simulating the full representation is x=<+,?> and y=<0,?> which corresponds to state 1 in Figure 1 and Table 1. The state graph (Fig. 3, RHS) shows that the system has eight states.

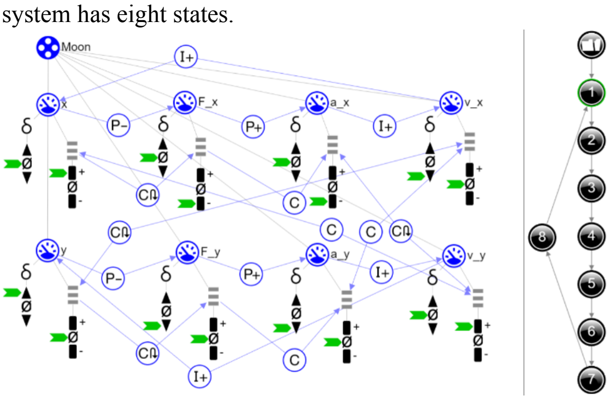


Fig. 3. Qualitative representation of circular motion. The vertical motion (with quantities y, F_y, a_y and v_y) is comparable to the horizontal motion (with quantities x, F_x, a_x and v_x). The simulation generates 8 states, as show in Fig. 3 on the RHS. The simulation result of state 1 is shown. An important insight concerns the four correspondences between the two motions.

Fig 4. Shows the value history of quantities x, a_x, v_x, y, a_y and v_y in the eight states. The value history shows the quantities, their possible values, their actual value, and their direction of change in each state. For instance, the quantity x in state 1 is positive and its change of direction is zero. By adding the correspondences, the motion in the vertical direction is now half a period out of phase with the horizontal motion. The sinusoidal patterns define the typical behavior observed in simple harmonic motion.

The relationship between position, velocity, and acceleration in simple harmonic motion can be summarized as follows: when an object is at its equilibrium position, the velocity is maximum and the acceleration is zero. For example, in state 3, x is at its equilibrium point on the x-axis and its direction of change is negative <0, −> and acceleration in the horizontal direction (a_x) is zero and its direction of change is positive <0, +>, i.e., the moon is in its equilibrium point on the x-axis and there is only gravitational pull in the vertical direction. The velocity in the horizontal direction is maximum in the negative direction <−, 0> , i.e., the moon is moving towards the left.

As the object moves away from the equilibrium position, the velocity decreases, and the acceleration increases in the opposite direction. When the object reaches its maximum displacement, the velocity becomes zero, and the acceleration is at its maximum (in the opposite direction). The cycle repeats as the object returns to the equilibrium position and continues oscillating.

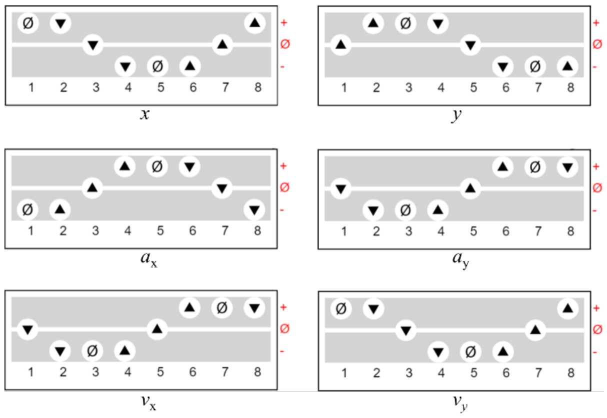


Fig. 4. Values history of x, a_x, v_x, y, a_y and v_y with regard to the eight states of circular motion.

3 Elliptical motion

Elliptical motion can be described by twelve distinct qualitative states (Fig. 5).

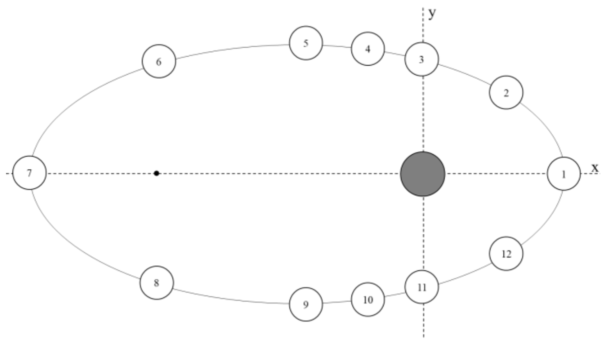


Fig. 5. Elliptical motion of a star orbiting a black hole. A system manifesting elliptical motion has twelve qualitatively distinct states.

A concrete example is a star orbiting a black hole, where the black hole is in one of the focal points of the ellipse. Within an elliptical orbit, as the distance from the black hole changes, the gravitational force exerted on the star varies, leading to corresponding alterations in acceleration. The equilibrium between gravitational force and the star's

Table 3. Twelve qualitative states of elliptical motion. Quantities are position: x-axis (x) and y-axis (y), acceleration: horizontal (a_x) and vertical (a_y), and velocity: horizontal (v_x) and vertical (v_y). Each quantity has a value and a direction of change, shown as <v,∂>. Force corresponds to acceleration. Force is not shown in this table.

	State											
Quantity	1	2	3	4	5	6	7	8	9	10	11	12
x	<+, 0>	<+, –>	<0, –>	<–, –>	<–, –>	<–, –>	<–, 0>	<–, +>	<–, +>	<–, +>	<0, +>	<+, +>
y	<0, +>	<+, +>	<+, +>	<+, +>	<+, 0>	<+, –>	<0, –>	<–, –>	<–, 0>	<–, +>	<–, +>	<–, +>
a_x	<–, 0>	<–, +>	<0, +>	<+, +>	<+, +>	<+, +>	<+, 0>	<+, –>	<+, –>	<+, –>	<0, –>	<–, –>
a_y	<0, –>	<–, –>	<–, –>	<–, –>	<–, 0>	<–, +>	<0, +>	<+, +>	<+, 0>	<+, –>	<+, –>	<+, –>
v_x	<0, –>	<–, –>	<–, 0>	<–, +>	<–, +>	<–, +>	<0, +>	<+, +>	<+, +>	<+, +>	<0, +>	<+, –>
v_y	<+, 0>	<+, –>	<+, –>	<+, –>	<0, –>	<–, –>	<–, 0>	<–, +>	<0, +>	<+, +>	<+, +>	<+, +>

momentum gives rise to the elliptical shape of the orbit. The specific shape of the ellipse depends on the starting situation of the object's motion, such as its distance, velocity, and direction relative to the central body. However, regardless of the specific shape, the presence of twelve states remains constant.

3.1 Expected qualitative states

Table 3 shows the values and directions of change for each of the quantities in the twelve states. States 1, 2, 6, 7, 8 and 12 in elliptical motion are similar to states 1, 2, 4, 5, 6, 8 in circular motion, respectively. In elliptical motion, there are six distinct states (3, 4, 5, 9, 10, and 11) that do not exist in circular motion, whereas states 3 and 7 in circular motion do not exist in elliptical motion. Although the relationships between position, force, acceleration, and velocity still govern the movements in both the horizontal and vertical directions, they are interdependent in a distinct manner compared to circular motion.

3.2 Completing the model

Table 4 shows the correspondences of elliptical motion and marks the differences with circular motion. Four bi-directional correspondences are the same as in circular motion: between x and a_x, a_x and v_x, y and a_x, and y and v_x. The other two correspondences (between x and v_y, and a_x and v_y) are different compared to circular motion: the values that correspond may differ, the correspondence can change from bi-directional to directed, or there may be no correspondence at all. Because in circular motion all correspondences are bi-directional, we will describe the specific changes for each pair of corresponding values in the context of elliptical motion below:

- x and v_y:
  - *(i)* In circular motion: if x=− then v_y=−. In elliptical motion there is no correspondence between x=− and values of v_y. That is, when the star is positioned on the left side of the y-axis (x=−), its vertical velocity is either downward (v_y=− in states 6, 7 and 8), it has no vertical velocity (v_y=0 in states 4 and 10), or vertical velocity is upward (v_y=+ in states 3 and 11). In elliptical motion the correspondence in the other direction (if v_y=− then x=−) is directed. When the star its vertical velocity is downward v_y=−, its position is on the left side of the y-axis (x=− in states 6, 7 and 8). This correspondence is directed because when the star is on the left side of the y-axis (x=−), it can also have no velocity in the vertical direction (v_y=0 in states 5 and 9) or its vertical velocity is upward (v_y=+ in states 4 and 10).
  - *(ii)* In circular motion: if x=0 then v_y=0. In elliptical motion, when the star crosses the y-axis its vertical velocity is upward (if x=0 then v_y=+ in state 3 and 11). So the value of this correspondence changed and it is now directed because the star its vertical velocity is also upward (v_y=+) when it is on the on the left (x=− in states 4 and 10) or on the right side of the y-axis (x=+ in states 1, 2, and 12). The correspondence in the other direction (if v_y=0 then x=0) changed its value and is now directed. When the star has no

Table 4. Correspondences in elliptical motion. The correspondences establish the co-occurrence of values of quantities of the horizontal and vertical direction of elliptical motion.

	value	x	y	a_x	a_y	v_x	v_y
x	−			$+^*$			^{3}
	0			0^*			$+^{1,2}$
	+			$-^*$			$+^2$
y	−				$+^*$	$+^*$	
	0				0^*	0^*	
	+				$-^*$	$-^*$	
a_x	−	$+^*$					$+^2$
	0	0^*					$+^{1,2}$
	+	$-^*$					^{3}
a_y	−		$+^*$			$-^*$	
	0		0^*			0^*	
	+		$-^*$			$+^*$	
v_x	−		$+^*$		$-^*$		
	0		0^*		0^*		
	+		$-^*$		$+^*$		
v_y	−	$-^2$		$+^2$			
	0	$-^{1,2}$		$+^{1,2}$			
	+	^{3}		^{3}			

* bi-directional correspondence; differences compared to circular motion: ^{1}value differs, ^{2} correspondence changed from bi-directional to directed, ^{3} no correspondence anymore.

vertical velocity, its position is on the left side of y-axis (if v_y=0 then x=− in states 5 and 9). This correspondence is directed because when the star its position is on the left side of the y-axis (x=−), its vertical velocity can be upward (v_y=+ in states 4 and 10) or downward (v_y=− in states 6, 7 and 8).

(iii) In circular motion: if x=+ then v_y=+. In elliptical motion, when the position of the star is on the right side of the y-axis, its vertical velocity is also upward (if x=+ then v_y=+ in states 1, 2, and 12), but the correspondence is now directed. It is directed because the star its vertical velocity is also upward (v_y=+) in states 3 and 11 (x=0) and states 4 and 10 (x=−). The correspondence in the other direction (if v_y=+ then x=+) does not exist in elliptical motion because the star its vertical velocity is upward (v_y=+) through the full quantity space of x {−, 0, +}.

- a_x and v_y:

(i) In circular motion: if a_x=− then v_y=+. In elliptical motion, when the star its acceleration in the horizontal direction is towards the left, its vertical velocity is also upward (if a_x=− then v_y=+ in states 1, 2, and 12). However, this correspondence is directed in elliptical motion because the star its vertical velocity is also upward (v_y=+) when horizontal acceleration is to the left (a_x=− states 4 and 10) or when there is no horizontal acceleration (a_x=0 in states 3 and 11). Therefore, the correspondence in the other direction (if v_y=+ then a_x=−) does not exist in elliptical motion.

(ii) In circular motion: if a_x=0 then v_y=0. In elliptical motion, when the star has no acceleration in the horizontal direction, its vertical velocity is upward (if a_x=0 then v_y=+ in states 3 and 11). This correspondence is directed because the star its vertical velocity is also upward (v_y=+) when acceleration in the horizontal direction is to the left (a_x=− in states 2 and 12) and to the right (a_x=+ in states 4 and 10). The value of the correspondence in the other direction (if v_y=0 then a_x=0) has changed. When the star is has no velocity in the vertical direction, the gravitational pull and thereby the acceleration in the horizontal direction is toward the right (if v_y=0 then a_x=+ in states 5 and 9). The correspondence is directed, because when the star its acceleration in the horizontal direction is to the right (a_x=+), velocity in the vertical direction can be downward (v_y=− in states 6, 7 and 8) or upward (v_y=+ in states 4 and 10).

(iii) In circular motion: if a_x=+ then v_y=−. In elliptical motion there is no correspondence between a_x=+ and values of v_y. That is, when the star its horizontal acceleration is to the right (a_x=+), its vertical velocity is downward (v_y=− in states 6, 7 and 8),), it has no vertical velocity (v_y=0 in state 3 and 11), or its vertical motion is velocity (v_y=+ in state 4 and 10). Therefore, the correspondence in the other direction (if v_y=− then a_x=+) is directed.

We can now add the correspondences between quantity spaces of both directions (vertical and horizontal) to describe elliptical motion (Fig. 6). As mentioned before, we do not need to add all correspondences from Table 4 because adding one correspondence can make another redundant. We add the bi-directional correspondences that are the same as in circular motion. We also add the directed correspondences that define states 3 and 11 (if x=0 then v_y=+) and states 5 and 9 (if v_y=0 then x=−).

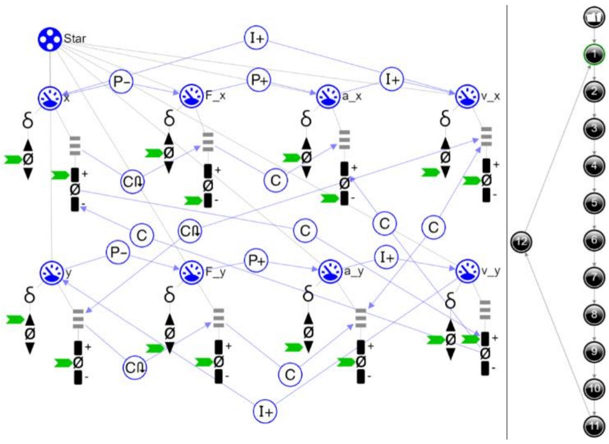


Fig. 6. Qualitative representation of elliptical motion. The simulation generates 12 states. The simulation result of state 1 is shown.

3.3 Simulation of the complete model

The representation can be simulated with initial conditions that correspond to state 1 in Fig. 5: x=<+,?> , y=<0,?> and v_x = <+,?>. The latter initial condition is needed because there is no correspondence that automatically sets the value of v_x in state 1.

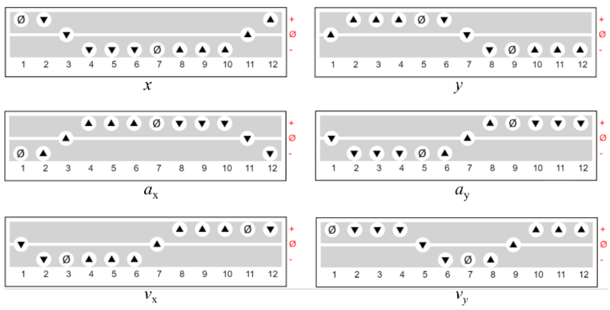


Fig. 7. Values history of x, a_x, v_x, y, a_y and v_y with regard to the twelve states of elliptical motion.

Fig 7. Shows the value history of quantities x, a_x, v_x, y, a_y and v_y in the twelve states of elliptical motion. While motions in both directions still exhibit sinusoidal patterns, it is important to note that in the case of elliptical motion, the system no longer strictly adheres to simple harmonic motion. The varying changes in gravitational force introduce complexities that deviate from the characteristics of circular motion in both directions.

4 Conclusion and discussion

In this paper, we present qualitative representations of circular and elliptical motion. The motions are depicted on a x- and y-coordinate plane. This allows for the decomposition of motion into a horizontal and vertical direction. To describe the dynamics of circular and elliptical motion, the representations include the quantities: position (x, y), force (F_x, F_y), acceleration (a_x, a_y), and velocity (v_x, v_y). The quantities have a quantity space that encompasses negative, zero, and positive values, hence {−, 0, +}. Note that force, acceleration and velocity are vectors and their qualitative value indicate both value *and* direction.

We describe the dependencies between quantities and the correspondences that exist in both circular and elliptical motion. Specifically, we focus on the correspondences between horizontal and vertical motion and highlight the differences between circular and elliptical motion.

Circular motion can be described by eight qualitatively distinct states, featuring six bi-directional correspondences between the quantities in the horizontal and vertical direction. When these correspondences are added to the representation, the system's behavior follows a pattern of two simple harmonic motions that are half a period out of phase.

Elliptical motion consists of twelve distinct qualitative phases. The dependencies between the quantities in both directions are similar to circular motion. However, compared to circular motion, there are changes in two correspondences: (*i*) between the horizontal position (x) and velocity in the vertical direction (v_y), and (*ii*) between acceleration in the vertical direction (a_x) and velocity in the vertical direction (v_y). These changes manifest in different ways: the values that correspond may differ, the correspondence itself may transition from being bi-directional to directed, or in some cases, there is no correspondence at all between certain values. These variations in correspondences highlight the distinct nature of elliptical motion compared to circular motion.

In conclusion, qualitative representations, such as the ones presented in this paper, offer an alternative approach to describing and understanding circular and elliptical motion, bypassing the traditional mathematical methods. By constructing qualitative representations, learners can gain valuable insights into the behavior of these systems, fostering a deeper comprehension of the concepts involved [Kragten & Bredeweg, 2023]. Future research aimed at continuous improvement of the pedagogical approach should examine how students learn optimally by constructing such representations and identify the essential support they need during the learning process.

5 References

[Alonzo & Steedle, 2009] Alicia Alonzo & Jeffrey Steedle. Developing and assessing a force and motion learning progression. *Science Education, 93*(3), 389–421, 2009.

[Barniol & Zavala, 2014] Pablo Barniol & Genaro Zavala. Test of understanding of vectors: A reliable multiple-choice vector concept test. *Physical Review Special Topics-Physics Education Research, 10*(1), 010121, 2014.

[Bredeweg et al., 2009] Bert Bredeweg, Floris Linnebank, Anders Bouwer, & Jochem Liem (2009). Garp3 — Workbench for qualitative modelling and simulation. *Ecological Informatics*, *4*(5-6), 263-281.

[Bredeweg et al., 2013] Bert Bredeweg, Jochem Liem, Wouter Beek, Floris Linnebank, Jorge Gracia, Esther Lozano,… David Mioduser. DynaLearn - An Intelligent Learning Environment for Learning Conceptual Knowledge. *AI Magazine*, *34*(4), 46-65, 2013.

[Bredeweg et al., 2023a] Bert Bredeweg, Marco Kragten, Joanna Holt, Patricia Kruit, Tom van Eijck, Monique Pijls,… Muriel de Boer. Learning with Interactive Knowledge Representations. *Applied Sciences, 13*(9), 5256, 2023.

[Bredeweg et al., 2023b] Bert Bredeweg, Marco Kragten, Joanna Holt, Dennis Vaendel, Joris Hanse, & Stef Bloemen. Stargazing Live! Inspiring with Real Data in a Mobile Planetarium and Learning Through Conceptual Modelling. In *ITS: 19th International Conference,* LNCS 13891, 257-269, 2023.

[Canlas, 2016] Ian Canlas. University students' alternative conceptions on circular motion. *International Journal of Scientific & Technology Research, 5*(3), 2277-8816, 2016.

[Kragten et al., 2022] Marco Kragten, Emile Jaspar, & Bert Bredeweg. Learning Domain Knowledge and Systems Thinking using Qualitative Representations in Secondary Education (grade 10-12). Paper presented at 35th *International Workshop on Qualitative Reasoning*, Vienna, Austria, 2022.

[Kragten & Bredeweg, 2023] Marco Kragten & Bert Bredeweg. Learning about circular motion of celestial bodies with interactive qualitative representations. *Artificial Intelligence in Education*: 24nd International Conference, AIED, 2023.

[Liu & Fang, 2016] Gang Liu & Ning Fang. Student misconceptions about force and acceleration in physics and engineering mechanics education. *International Journal of Engineering Education, 32*(1), 19–29, 2016.

An Evaluation of ChatGPT-4's Qualitative Spatial Reasoning Capabilities in RCC-8

A G Cohn

University of Leeds & The Alan Turing institute, UK

Abstract

Qualitative Spatial Reasoning (QSR) is well explored area of Commonsense Reasoning and has multiple applications ranging from Geographical Information Systems to Robotics and Computer Vision. Recently many claims have been made for the capabilities of Large Language Models (LLMs). In this paper we investigate the extent to which one particular LLM can perform classical qualitative spatial reasoning tasks on the mereotopological calculus, RCC-8.

Introduction

Qualitative Spatial Reasoning (QSR[1]) (Cohn and Renz 2008; Chen et al. 2015; Cohn and Hazarika 2001) is a well developed field which is concerned with the representation of qualitative spatial information and reasoning with it. In natural language, spatial information is usually represented qualitatively (using prepositions such as *on, in, left of, part of, under, touching, ...)* and many calculi have been developed to represent such information. There are calculi for mereological relations (such as RCC-5 (Jonsson and Drakengren 1997)), mereotopological relations (such as RCC-8 (Randell, Cui, and Cohn 1992; Cohn et al. 1997)), directions (such as OPRA (Moratz 2006)), size (Gerevini and Renz 2002) for example as well as calculi combining two different aspects of spatial information, such as the Rectangle Algebra (Guesgen 1989; Mukerjee and Joe 1990) which can represent both mereotopological information as well as directional. What is common to all these calculi is that they consist of a set of *jointly exhaustive and pairwise disjoint* (JEPD) *base* relations. For example, RCC-8 contains eight JEPD *base* relations, illustrated in 2D in Fig. 1.

Large Language Models (LLMs) (Devlin et al. 2019; Brown et al. 2020), such as ChatGPT-4 (Roumeliotis and Tselikas 2023) are a recent example of so called *Foundation Models* which have been trained on very large textual corpora in order to generate text in response to a prompt. This is not the place to survey this burgeoning field, but we note that many claims have been made for the power and apparent intelligent behaviour that these models can display. In particular their performance on some benchmarks may lead one

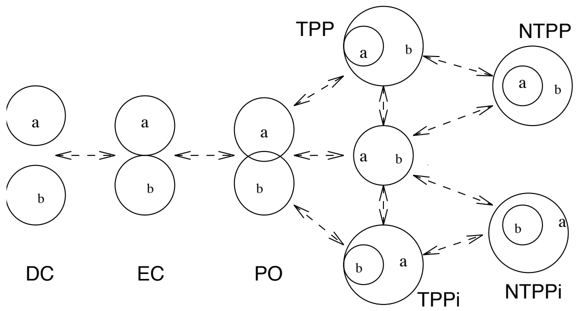


Figure 1: The eight relations of the RCC-8 calculus illustrated in 2D.

to believe that they possess, at least to some degree, the ability to perform commonsense reasoning. Spatial reasoning is usually regarded as one core aspect of common sense so it is natural to ask whether LLMs can reason about qualitative spatial information. This is the question that we address here.

In earlier work (Cohn and Hernandez-Orallo 2023) we use extended dialogues with an LLM to try to map the boundaries of spatial commonsense in some LLMs, addressing a variety spatial challenges, and examining not only the response given but also the explanation/justification of the response, but did not specifically focus on existing QSRs, though some questions were asked which do correspond to particular reasoning steps in an existing QSR. Here we focus on one specific QSR and ask the question as to what extent an LLM can perform reasoning in that calculus, and conduct a more exhaustive evaluation, but looking at the ability to perform compositions between relations and also to reason about the conceptual neighbourhood diagram of the calculus. Weaknesses in the reasoning powers of LLMs have previously been noted (e.g. (Cai, Chang, and Han 2023)) so one might not expect LLMs to perform well in this regard. But on the other hand, there are a large number of papers about QSR in the literature and these are likely to have formed part of the training corpus of an LLM, and thus might facilitate correctly responding to prompts – though the information concerning the actual reasoning steps are often given in tables (in particular *composition tables* – see below) and thus

[1]We may use QSR as shorthand for both Qualitative Spatial Reasoning and Qualitative Spatial Representation; context should usually make clear which is intended.

might be hard for LLM training procedures to process well.

There are now many LLMs in the literature. Some of these are open source and are explicit about the training corpus; others are closed and give no specific information about the training, or the precise corpus, such as the GPT family of LLMs. Nevertheless since we observed previously (Cohn and Hernandez-Orallo 2023) that ChatGPT-4 and GPT4 were the most performant for spatial reasoning, we use ChatGPT-4 as the LLM with which we perform our experiments. In each of the experiments below, an initial prompt gave the problem setting and the task to be performed. Subsequent prompts in the conversation probed about one specific inference (e.g. one cell in a composition table). Each experiment was a separate conversation and the version number of ChatGPT-4 was given as "May 24" (presumably, the version of May 24 2023). The conversations can be found at the following location: https://tinyurl.com/qr23sup.

Compositional Reasoning

The most researched form of reasoning with Qualitative Spatial Relations is that of composition: i.e. given a two facts R1(x, y), and R2(y, z), then what relations are possible between x and z? In general more than one base relation is possible, so the answer may take the form of a disjunction of relations, as can be seen in the composition table for RCC-8 in Fig. 2. Given the ubiquity of RCC-8 in the QSR literature, in this paper we focus on the abilities of an LLM to reason with RCC-8.

Experiment 1: Compositional Reasoning in RCC-8

In this experiment the standard names of the eight relations were given as well as a sentence stating that the relations were part of the RCC-8 calculus:

> *Consider the following set of eight pairwise disjoint and mutually exhaustive binary spatial relations. These relations form part of the well known RCC-8 qualitative spatial reasoning calculus. DC(x,y) means that x and y are disconnected and share no spatial parts. EC(x,y) means that x and y touch at a boundary but do not share any interior parts. PO(x,y) means that x and y share a spatial part, but neither is part of the other. TPP(x,y) means that x is part of y and touches y's boundary. NTPP(x,y) means that x is part of y but does not touch y's boundary. TPPi(x,y) is the same as TPP(y,x). NTPPi(x,y) is the same as NTPP(y,x). Finally, EQ(x,y) means that x and y are coincident. I will now ask you a series of questions about these relations. There may be more than one possible relation, in which case name all of the possible answers. If all eight relations are possible, just say ALL.*

As an example of a prompt and ChatGPT-4's response, consider the first one:

> **Prompt:** *If DC(x,y) and DC(y,z) then what are the possible relationships between x and z.* **Response:** *If DC(x,y) and DC(y,z), then x and z can either be disconnected (DC) or externally connected (EC). This is because the disconnection between x and y, and y and z, only guarantees that neither x nor z is connected to y. It does not preclude the possibility that x and z themselves might be disconnected or touching at a boundary. However, the other relations like part-of (TPP, NTPP) and their inverses (TPPi, NTPPi), partial overlap (PO), and equality (EQ) are not possible because they would imply some level of connection between x and z through y, which contradicts our initial conditions. So, the possible relationships between x and z are: DC(x,z): x and z are disconnected. EC(x,z): x and z are externally connected.*

Notice that in fact the composition of DC with DC is the union of all eight RCC-8 relations, so the reasoning is incorrect here.

The results of the 7*7=49 prompts issued (compositions involving EQ were omitted since they should be trivial) are summarised in Fig. 3. There is a total of 85 relations correctly predicted (those in black), 197 relations which are correctly not predicted (those not in the table). Thus there are a total of 282 correctly predicted (both present and missing) relations, i.e. 71.94% of the total possible (7*7*8 = 392). 11 of the cells (out of 49) are completely correct. This is clearly much better than chance but far from a correct composition table. Fig. 4 shows the absolute number and relative percentages of the 8 relations in the proportion that they are correctly or incorrectly predicted. As can be seen, DC is never missed as a correct prediction, and PO is never incorrectly predicted as present. DC has the highest accuracy which is perhaps not surprising since DC might be regarded as intuitively easier to reason about (and this fits with the predictions about human preferred models in the section below). Including both correctly predicted and correctly missing relationships, all eight relations have broadly similar accuracies.

In order to test whether the result was influenced by prior knowledge of RCC-8 gained as part of its training, we also performed the same experiment, but with all the relation names prefixed by an X to disguise the connection to RCC-8. The prompt was the same as above except for the change of relation names and the omission of the second sentence. The results are given in Fig. 5 while Fig. 6 shows the absolute number and relative percentages of the 8 relations in the proportion that they are correctly or incorrectly predicted. As can be seen, DC again is never missed as a correct prediction, and EC is only missed twice; again PO is never incorrectly predicted as present. As before, DC, EC, and POhave the highest accuracies, along with EQ, but EQ is never predicted as present correctly, only incorrectly. The overall average of correctly predicted relations (present and missing) drops from 71.94% in the non-anonymous case above to 67.09% so there is some loss of performance though whether is due to the anonymisation of the relations or the stochasticity of ChatGPT-4 is not clear.

Experiment 2: Preferred Compositions in RCC-8

As noted above, in general a composition of two relations will yield more than one possible base relations, but it turns

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	=
DC	*	DC, EC, PO, TPP, NTPP	DC, EC, PO, TPP, NTPP	DC, EC, PO, TPP, NTPP	DC, EC, PO, TPP, NTPP	DC	DC	DC
EC	DC, EC, PO, TPPi, NTPPi	DC, EC, PO, TPP, TPPi, =	DC, EC, PO, TPP, NTPP	EC, PO, TPP, NTPP	PO, TPP, NTPP	DC, EC	DC	EC
PO	DC, EC, PO, TPPi, NTPPi	DC, EC, PO, TPPi, NTPPi	*	PO, TPP, NTPP	PO, TPP, NTPP	DC, EC, PO, TPPi, NTPPi	DC, EC, PO, TPPi, NTPPi	PO
TPP	DC	DC, EC	DC, EC, PO, TPP, NTPP	TPP, NTPP	NTPP	DC, EC, PO, TPP, TPPi, =	DC, EC, PO, TPPi, NTPPi	TPP
NTPP	DC	DC	DC, EC, PO, TPP, NTPP	NTPP	NTPP	DC, EC, PO, TPP, NTPP	*	NTPP
TPPi	DC, EC, PO, TPPi, NTPPi	EC, PO, TPPi, NTPPi	PO,TPPi,NTPPi	PO, TPP, TPPi, =	PO, TPP, NTPP	TPPi, NTPPi	NTPPi	TPPi
NTPPi	DC, EC, PO, TPPi, NTPPi	PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO, TPP, NTPP, TPPi, NTPPi, =	NTPPi	NTPPi	NTPPi
=	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	=

Figure 2: The RCC-8 Composition Table (Cohn et al. 1997)

.

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi
DC	DEPTNQtn	DEPTN	DEPTN	DEPTN	DEPTN	DE	DE
EC	DEPtn	DE	DEPTtQ	DETN	DETN	DE	DE
PO	DEPtn	DEPTNQtn	DEPTNQtn	DEPTNQtn	DEPTNQtn	DEPTNQtn	DEPTNQtn
TPP	D	DE	DEPTN	TN	N	DEPTN Qt	DEPTNtn
NTPP	D	DE	DEPTN	TN	N	DEPTN	DEPTNQtn
TPPi	DEPtn	DEPtn	DEPTQtn	PTQtn	PTNQtn	TQtn	TQtn
NTPPi	D	DEPTN	PQtn	TNQn	NQn	Qtn	Qn

Figure 3: The Composition Table for RCC-8 produced by ChatGPT-4. The entry in each cell uses the following coding: D (DC), E(EC), P(PO), T(TPP), N(NTPP), t(TPPi), n(NTPPi), Q(EQ). Black means that relation is correctly predicted (85 times), red means that relation is incorrectly predicted (61 times), blue means that the relation was incorrectly not predicted (49 times).

.

out that humans tend to have a "preferred" relation. For example, Ragni et al (2007) report on experiments performed on native German speakers and native Mongolian speakers for RCC-8. In their experiments the relations were described, but the human subjects were not allowed to draw possible configurations, so the setting is essentially equivalent to an LLM setting.

Given that humans may struggle to see all the possible relations[2], determining whether there is agreement about the most preferred is good question to ask. It turns out that there is good agreement in general across and within the two cultures, with the the percentage of people agreeing with the same preferred relation ranging from 30% to 87.5% (a random choice would yield 12.5% on average since there are eight relations to choose from). (They did not query cases where the composition yields a unique relation, nor did they consider EQ as one of the two relations as this should be a trivial task.) This agreement is perhaps surprising since the two languages are linguistically very different. Ragni et al (2007) do report some differences though – for example although both language speakers preferred DC whenever it was consistent, Mongolians preferred PO over NTPPi

[2]The fact that some humans may struggle to compute the composition table does not stop it being a valid question to see if an LLM can determine the correct entries.

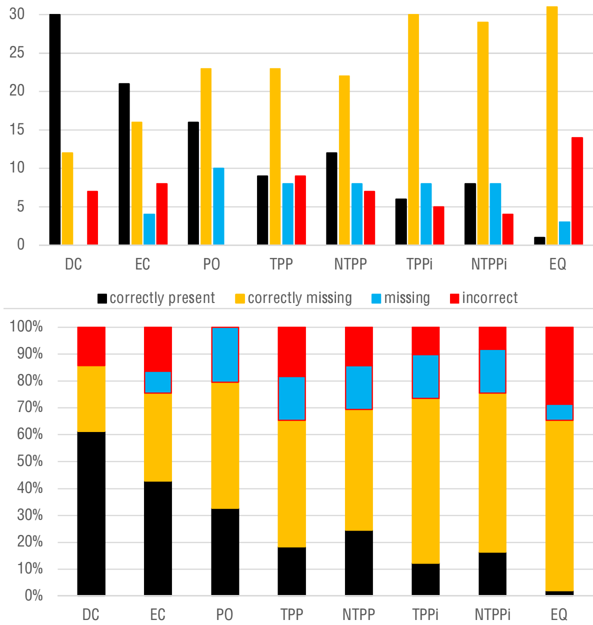


Figure 4: Relation statistics for the Composition Table for RCC-8 produced by ChatGPT-4. The upper chart shows the absolute number of relations, and the lower the relative percentage for each relation.

.

whereas for Germans the converse was true. Both cultures only chose EQ when composing a relation with its inverse (e.g. TPP with TPPi).

The theory of *preferred mental models* (Knauff, Rauh, and Schlieder 1995) states that people construct the simplest (computationally cheapest) model consistent with the premises. Their experiments showed that humans prefer models with the smallest overlapping complexity which explains the preference for DC noted above.

Given the difficulty reported in Experiment 1 in ChatGPT-4 correctly inferring all possible relations in a composition,

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi
DC	DE PTNQtn	DE PTN	DE PTN	DE PTN	DE PTN	DE	DE
EC	DE Ptn	DE PTQt	DEP TN	DEPTN	DEPTN	DEP	DE
PO	D Etn	DEP tn	DEP TNQt	DEPTN	DEPTN	DEPTtn	DEPTtn
TPP	D	DE	DEPTN	TN	N	DEPTQt	DEPNtn
NTPP	D	DE	DEPTN	N	N	DEPTNn	DEPTNQtn
TPPi	DEPtn	DEPtn	DEPTNtn	DEPTNQtn	DEPTNtn	TQtn	DEPTNtn
NTPPi	DEPtn	DEPtn	DP	DPTNtn	DPTNtn	DPTNtn	DPTNtn

Figure 5: The Composition Table for RCC-8 produced by ChatGPT-4 using anonymised relations. The entry in each cell uses the following coding: D (DC), E (EC), P (PO), T (TPP), N (NTPP), t (TPPi), n (NTPPi), Q (EQ). Black means that relation is correctly predicted (95 times), red means that relation is incorrectly predicted (57 times), blue means that the relation was incorrectly not predicted (72 times).

.

asking this more specific query seems a natural task to try; this gives some measure of whether the cognitive preferences of humans correspond at all to the choices made by ChatGPT-4.

The initial prompt given to ChatGPT-4 was identical to the initial prompt above when querying about composition. In its responses, sometimes ChatGPT-4 gave a clear preference for one relation, for example:

> **Prompt:** *If DC(x,y) and DC(y,z) then what is your preferred relationship between x and z?* **Response:** *The relations DC(x,y) and DC(y,z) mean that region x is disconnected from region y and region y is disconnected from region z. These relationships don't provide any information about the spatial relationship between x and z. x and z could be disconnected, they could touch at a boundary, or they could partially overlap, among other possibilities. However, in absence of any other information and considering spatial continuity, I would say the most likely relation is DC(x, z), assuming x and z are also disconnected. Please note that this is just a statistical assumption; it's possible for x and z to have any of the other relations defined in RCC-8.*

In other cases ChatGPT-4 justified its choice by saying it was the "cautious" choice, or the "safest choice". The humans in the (Ragni, Tseden, and Knauff 2007) experiment were not asked to justify their choices, but the claim made by the authors of that paper noted above is that choices were based on computational/simplicity considerations – which differs from the reasons claimed here by ChatGPT-4.

The initial prompt to ChatGPT-4 given was:

> *Consider the following set of eight pairwise disjoint and mutually exhaustive binary spatial relations, between two one piece spatial 2D regions. These relations form part of the well known RCC-8 qualitative spatial reasoning calculus. DC(x,y) means that x and y are disconnected and share no spatial parts. EC(x,y) means that x and y touch at a boundary but do not share any interior parts. PO(x,y) means that x and y share a spatial part, but neither is part of the other. TPP(x,y) means that x is part of y and touches y's boundary. NTPP(x,y) means that x is part of y but does not touch y's boundary. TPPi (x,y) is the same as TPP(y,x). NTPPi(x,y) is the same as NTPP(y,x). Finally, EQ(x,y) means that x and y are coincident. I will now ask you a series of questions about these relations. It is possible that in some cases there could be more than one relation that might hold; in these cases just give me your single most preferred relationship.*

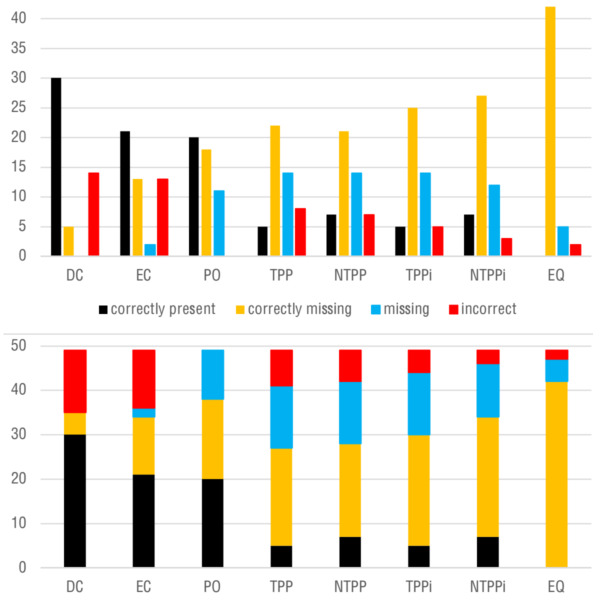


Figure 6: Relation statistics for the Composition Table for RCC-8 with anonymised relations produced by ChatGPT-4. The upper chart shows the absolute number of relations, and the lower the relative percentage for each relation.

.

As can be seen in Fig. 7, ChatGPT-4 only agreed with the average human on 20/49 (40.82%) though in a further three cases it agreed with one of the language groups (twice Mongolian, once German) giving a total of 23/49 (46.93%). In seven cases it actually predicted an impossible relation as its preferred composition[3], and in all cases where the composition is unique it failed to note that. In all the other cases, 18/49 (36.63%) its preferred relation was possible but not preferred by either language group or overall.

As can be seen, sometimes ChatGPT-4 made a logically inconsistent prediction, for example with EC/NTPPi; looking at the ChatGPT-4's response in this case we can see that this is because it seems to misunderstand the meaning of NTPPi:

> *If EC(x,y) and NTPPi(y,z), this means that region x touches the boundary of region y but does not overlap with it, and region y is a part of region z but does*

[3]Sometimes the humans in the (Ragni, Tseden, and Knauff 2007) experiment also predicted an impossible relation as can be seen in Table 2 of their publication.

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi
DC	DC	DC	DC	DC	DC	DC	DC
EC	DC	DC	EC	EC	EC	EC	EC
PO	DC	EC	PO	PO	PO	PO	PO
TPP	DC	EC	PO	TPP	NTPP	DC	DC
NTPP	DC	DC	NTPP	NTPP	NTPP	DC	DC
TPPi	DC	EC	PO	PO	PO	TPP	NTPP
NTPPi	DC	DC	PO	NTPP	NTPP	NTPP	NTPP

Figure 7: The table of preferred compositions for RCC-8 produced by ChatGPT-4. Green text means it agreed with the human most likely model. Orange text means it agreed with the human most likely model but failed to notify that it was the only possible model. Red means it chose an impossible relation. Blue means it chose a possible but not preferred model. A green background means that it agreed with preferred relation of one of the language groups but not the overall average.

.

not touch z's boundary, with the perspective of y being contained in z. In this scenario, it's likely that x and z are either disconnected (DC) or touch at the boundary (EC). Since y, which is part of z but does not touch z's boundary, is touching x at the boundary, there's a possibility for indirect spatial contact between x and z. Without additional context, it's safer to assume that x and z touch at the boundary (EC). However, this is just an assumption, and the actual relationship between x and z could be different depending on the specific spatial arrangement.

In a different case, TPPi/TPPi, the initial understanding of the relations is correct inferring that z is part of x, but then ChatGPT-4 makes a wrong step in the final part of its reasoning, and claims that TPP(x, y):

> *If TPPi(x,y) and TPPi(y,z), this means that region y is a part of region x and touches x's boundary, and region z is a part of region y and touches y's boundary. Given these conditions, it's likely that z is a part of x and touches x's boundary (TPP). This is because z, which is a part of y and touches y's boundary, is also indirectly a part of x through y, which is a part of x and touches x's boundary. Therefore, my preferred relationship between x and z would be TPP(x,z), assuming that z is a part of x and touches x's boundary. Please note that this is an assumption and the actual relationship between x and z could be different depending on the specific spatial arrangement.*

ChatGPT-4 makes exactly the same error with TPPi/NTPPi. For the wrong NTPPi/EC case, the the description of its reasoning contains a more fundamental error:

> *...This is because y, which is a part of x and does not touch x's boundary, is also only touching z's boundary, so x and z would not have any overlap...*

A similar more fundamental error is made in the cases of NTPPi / TPP, whilst in the final two wrong cases, ChatGPT-4 misunderstands the second relation as its inverse.

In order to test whether the result was influenced by prior knowledge of RCC-8 gained as part of its training, we also performed the same experiment, but with all the relation names prefixed by an X to disguise the connection to RCC-8. The prompt was the same as above except for the change of relation names and the omission of the second sentence. The results are given in Fig. 8. ChatGPT-4 only agreed with the average human on 24/49 (48.99%) though in a further five cases it agreed with one of the language groups (twice times Mongolian, twice German), and in one case the Mongolian preference was equal between the overall choice and the choice ChatGPT-4 made giving a an agreement with the German speakers of 26/49 (53.06%) and 27/49 (55.1%) with the Mongolian speakers. In three cases (6.12%) it actually predicted an impossible relation as its preferred composition (this also happened sometimes in the human experiment), and in two cases (out of 11) where t he composition is unique it failed to note that. In all the other cases, 20/49 (60.82%) its preferred relation was possible but not preferred by either language group or overall. The overall agreement with humans in this anonymised case is actually significantly higher than the non-anonymised experiment above; it is not clear if there is a reason for this and if so what.

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi
DC	DC	DC	DC	DC	DC	DC	DC
EC	DC	EC	PO	EC	EC	EC	EC
PO	DC	EC	PO	PO	PO	PO	PO
TPP	DC	EC	PO	TPP	NTPP	DC	DC
NTPP	DC	DC	PO	NTPP	NTPP	DC	DC
TPPi	DC	EC	PO	PO	PO	TPPi	NTTPi
NTPPi	DC	DC	PO	NTTPi	NTPP	NTPPi	NTPPi

Figure 8: The table of preferred compositions for RCC-8 produced by ChatGPT-4 with anonymised relations. The relation names are coloured in the same way as in Table 7.

.

Experiment 3: Spatial Continuity

Continuity networks were introduced by Randell and Cohn(1989) to represent the set of possible 'next' relations that might obtain, assuming that motion is continuous and any transformations in the shape and/or size of an object are also continuous. Subsequently these have been termed "conceptual neighbourhoods" (Freksa 1992). Fortunately, the abbreviation for both of these terms is CN, which we shall use henceforth. The CN for RCC-8 is depicted in Fig. 1. The initial prompt was the same as for the composition table prompt above, but with two additional sentences at the end: *In all these questions, assume that all motion is continuous, and any deformations of the spatial regions are also continuous. Also please describe the motion or changes of size or shape that would be required for the change of relation to take place*. Each subsequent prompt also emphasises the need for the relation to be an immediate next relation. As an example prompt/response pair consider the EC case:

> **Prompt:** *If EC(a,b) holds, then what are the next possible relations that might hold as the two regions translate or change their shape or size? I am interested in the immediate next relation, so no intermediate relation must hold.* **Response:** *If EC(a,b) holds,*

which means that the two regions a and b are just touching at a boundary but do not share any interior parts, then the immediate next possible relations as the two regions translate or change their shape or size could be either DC(a,b) or PO(a,b). DC(a,b): This would occur if the regions move away from each other and are no longer in contact at all, becoming fully disconnected. PO(a,b): This would occur if one region begins to overlap with the other, but neither region is completely contained within the other. They would then be partially overlapping, sharing some but not all of their spatial parts. The transitions to these relations would involve either moving the regions apart or moving them further into each other. In the case of transition to PO(a,b), there could also be some deformation of the regions if they change shape to allow for the partial overlap.

An example of a faulty prediction is from PO:

If PO(a,b) holds, indicating that regions a and b partially overlap but neither is fully contained within the other, the immediate next relations could be EC(a,b), DC(a,b), TPP(a,b), TPPi(a,b), or EQ(a,b). EC(a,b): This would occur if the overlapping parts of a and b are reduced to just boundary parts, meaning the regions are touching but no longer overlapping. DC(a,b): This would occur if regions a and b are moved or deformed such that they no longer overlap or touch at all. TPP(a,b) or TPPi(a,b): This would occur if one region becomes fully contained within the other, either by moving into it or shrinking. TPP(a,b) means a is part of b and touches b's boundary. TPPi(a,b) means b is part of a and touches a's boundary. EQ(a,b): This could occur if the regions are reshaped or resized such that they coincide exactly, becoming the same region in space. The transitions to these relations would involve either moving the regions, resizing them, or reshaping them in some way.

Here, ChatGPT-4 has mistakenly said that DC is a next relation, failing to notice that it has to pass through EC on the way.

As can be seen, the predictions are mostly correct with just five errors: three where a link is falsely predicted and two where a link is missing. The two missing links are both from EQ (to NTPP and to NTPPi). Interestingly the links in the reverse direction are correctly predicted, so ChatGPT-4 is not able to reason that a link in one direction implies the link in the other direction should also be present. Similarly whilst the absence of a link from DC to PO is correctly predicted, the inverse case is not, and is also the case for PO to/from NTPP and NTPPi.

In order to test whether the result was influenced by prior knowledge of RCC-8 gained as part of its training, we also performed the same experiment, but with all the relation names prefixed by an X to disguise the connection to RCC-8. The prompt was the same as above except for the change of relation names and the omission of the second sentence. The results are given in Fig. 10. There are 3 incorrectly predicted

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ
DC		x						
EC	x		x					
PO	x	x		x		x		x
TPP			x		x			x
NTPP			x	x				x
TPPi			x				x	x
NTPPi			x			x		x
EQ			x	x		x		

Figure 9: The Continuity Table for RCC8 produced by ChatGPT-4. An 'x' means that the relation in that column is predicted as an immediate neighbour of the relation in that row. An empty box means that the relation is not predicted as an immediate neighbour. Green means that the prediction was correct and red that it was incorrect. The leading diagonal is white since a relation is not a next relation of itself.

.

links, 3 missing links, 19 correctly predicted links and 31 correct missing links, giving an accuracy of 50/56 (89.2%). This is slightly worse than the case above. There is one more missing link but the missing links are all different in the two cases. Although there are the same number of wrong links, only one of these is in common (PO to DC). Overall the results are broadly similar and may be due to the stochastic nature of ChatGPT-4's responses, suggesting that either the disguise was not very effective, or that prior training did not really affect the response and it was able to reason from 'first principles' (if not always correctly) in response to each prompt.

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ
DC		x						
EC	x		x					
PO	x	x		x		x		
TPP		x	x		x			
NTPP				x				x
TPPi		x	x				x	
NTPPi						x		x
EQ			x	x	x	x	x	

Figure 10: The Continuity Table for RCC-8 produced by ChatGPT-4 using disguised relation names. The meaning of the colouring is the same as in Fig. 9.

.

Concluding Remarks and Future Work

This investigation has supported the widely-held view that LLMs can struggle to do reasoning tasks[4]. In the case of Experiment 1, in which ChatGPT-4 was asked to compute the entire composition table for RCC-8, this is a non trivial task even for humans, so it is perhaps not surprising that ChatGPT-4 did not achieve 100% accuracy – the scores of 71.94% (and 67.09% for the anonymised relatins) are clearly much better than chance and do suggest a reasonable facility to perform such computations. A detailed analysis of the actual conversations in the supplementary material shows that

[4]Bender et al (2021) have observed that LLMs might be regarded just as "stochastic parrots" and thus it is not suprising that precise, logically correct deductive reasoning is challenging for an LLM.

sometimes ChatGPT-4 does appear able to do some interesting (qualitative) spatial reasoning, but often fails, sometimes making elementary mistakes. It also shows inconsistency in being able to reason correctly about a relation but not its inverse. It also sometimes confuses a relation with its inverse. It is possible that fine tuning, explicit chain-of-thought prompting, or more carefully engineered prompts might improve performance; however, given the stochastic nature of LLMs it seems unlikely that the results would be as good as logical reasoning (the experiment on preferred relations is of course not strictly a logical reasoning exercise, except for the requirement not to predict spatially impossible relations).

There are a variety of avenues for further work which present themselves. Other calculi could be experimented with – for example the coarser calculus RCC-5, or calculi for reasoning about direction or size (Cohn and Renz 2008). Other LLMs could be evaluated – though since new LLMs and new LLM versions are continually being released, this is a challenge with no definite stopping point. Tracking the change in performance of a particular LLM across releases would also be of interest – though in the case of closed LLMs such as ChatGPT-4 where the owners have the right to harvest user conversations and use them for future training, it will not be clear if any improvement is the result of leakage from the previous conversation or more general performance improvement[5]. It has already been observed (Cohn and Hernandez-Orallo 2023) that different LLMs have different strengths – determining which LLMs are better at which spatial reasoning tasks would also be worth of future investigation. The overall conclusion that LLMs in general struggle with more complex spatial reasoning tasks is likely to remain the case, at least for the foreseeable future. In the API version of GPT, different temperatures could be tried, and multiple runs with averages computed. Different prompts and prompting strategies could be tried, though arguably since QSR has always been viewed as a form of commonsense reasoning, it should not be necessary to devise specific prompts to elicit commonsense behaviour.

It is not clear how successful the anonymisation was – in one case I mistyped an X relation and it was able to suggest the intended relation name, suggesting that it has the ability to dissect relation names; thus more sophisticated anonymisation might be tried. In earlier work (Cohn and Hernandez-Orallo 2023) we had already done some limited experimentation asking an LLM to reason about spatial relations in a real world context rather than the purely abstract setting used in the experiments in this paper – it would be interesting to conduct more extensive tests LLMs doing compositional reasoning in a more realistic setting, and similarly for the continuity experiment.

Experiment 2 above already investigated how LLM performance compared to human performance to a limited extent but further investigation would be worthwhile, including a head-to-head comparison rather than simply taking a result from the literature originally intended to investigate a different question. Another interesting avenue for further work will be to explore the use of multimodal FMs – when humans perform spatial reasoning tasks including the challenge of building a composition table, it is natural to use pencil and paper to sketch diagrams and possible scenarios – investigating whether a multi-modal FM with such abilities (including the ability to analyse its own drawings) would be of great interest to the spatial reasoning community.

As mentioned above, another possible avenue of research is to investigate different prompting strategies, including k-shot (Dang et al. 2022), chain-of-thought(Wei et al. 2022) and tree-of-thought(Yao et al. 2023) strategies. Not doing so was deliberate in this paper as I was interested in exploring in how the "vanilla" LLM would perform. Whilst for specific downstream tasks, fine-tuning or employing specific prompting strategies may reasonable, there is an argument to be made that for commonsense reasoning, this is not a reasonable strategy since the task is a general one rather than a specific downstream task.

[5]However, note that no feedback was given to ChatGPT-4 as to whether the proffered response was correct or not.

Data statement

All the conversations with ChatGPT-4 that support the summary tables in this paper can be found at http://tinyurl.com/qr23sup.

Acknowledgments

This work was supported by: The Alan Turing Institute; the Economic and Social Research Council (ESRC) under grant ES/W003473/1; the Turing's Defence and Security programme through a partnership with the UK government in accordance with the framework agreement between GCHQ and The Alan Turing Institute.

References

Bender, E. M.; Gebru, T.; McMillan-Major, A.; and Shmitchell, S. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 610–623.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Cai, Z.; Chang, B.; and Han, W. 2023. Human-in-the-Loop through Chain-of-Thought. *arXiv preprint arXiv:2306.07932*.

Chen, J.; Cohn, A. G.; Liu, D.; Wang, S.; Ouyang, J.; and Yu, Q. 2015. A survey of qualitative spatial representations. *The Knowledge Engineering Review*, 30(1): 106–136.

Cohn, A. G.; Bennett, B.; Gooday, J.; and Gotts, N. M. 1997. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *Geoinformatica*, 1(3): 275–316.

Cohn, A. G.; and Hazarika, S. M. 2001. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2): 1–29.

Cohn, A. G.; and Hernandez-Orallo, J. 2023. Dialectical language model evaluation: An initial appraisal of the commonsense spatial reasoning abilities of LLMs. *arXiv preprint arXiv:2304.11164*.

Cohn, A. G.; and Renz, J. 2008. *Handbook of Knowledge Representation*, chapter Qualitative Spatial Representation and Reasoning. Elsevier.

Dang, H.; Mecke, L.; Lehmann, F.; Goller, S.; and Buschek, D. 2022. How to Prompt? Opportunities and Challenges of Zero- and Few-Shot Learning for Human-AI Interaction in Creative Applications of Generative Models. arXiv:2209.01390.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Freksa, C. 1992. Temporal reasoning based on semi-intervals. *Artificial intelligence*, 54(1-2): 199–227.

Gerevini, A.; and Renz, J. 2002. Combining topological and size information for spatial reasoning. *Artificial Intelligence*, 137(1-2): 1–42.

Guesgen, H. W. 1989. Spatial reasoning based on Allen's temporal logic, Technical Report TR-89-049. Technical report, International Computer Science Institute Berkeley.

Jonsson, P.; and Drakengren, T. 1997. A Complete Classification of Tractability in RCC-5. *J. Artif. Intell. Res.*, 6: 211–221.

Knauff, M.; Rauh, R.; and Schlieder, C. 1995. Preferred mental models in qualitative spatial reasoning: A cognitive assessment of Allen's calculus. In *Proceedings of the seventeenth annual conference of the Cognitive Science Society*, 200–205.

Moratz, R. 2006. Representing relative direction as a binary relation of oriented points. In *ECAI*, volume 6, 407–411.

Mukerjee, A.; and Joe, G. 1990. A Qualitative Model for Space. In Shrobe, H. E.; Dietterich, T. G.; and Swartout, W. R., eds., *Proceedings of the 8th National Conference on Artificial Intelligence. Boston, Massachusetts, USA, July 29 - August 3, 1990, 2 Volumes*, 721–727. AAAI Press / The MIT Press.

Ragni, M.; Tseden, B.; and Knauff, M. 2007. Cross-cultural similarities in topological reasoning. In *Spatial Information Theory: 8th International Conference, COSIT 2007*, 32–46. Springer.

Randell, D.; and Cohn, A. 1989. Modelling topological and metrical properties in physical processes. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, 357–368.

Randell, D.; Cui, Z.; and Cohn, A. G. 1992. A Spatial Logic Based on Regions and Connection. In *3rd International Conference on Knowledge Representation and Reasoning, 1992*, 165–176.

Roumeliotis, K. I.; and Tselikas, N. D. 2023. ChatGPT and Open-AI Models: A Preliminary Review. *Future Internet*, 15(6): 192.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Chi, E.; Le, Q.; and Zhou, D. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Grounding Causality in Bayesian Networks Using Qualitative Reasoning

Mélanie Munch[1], Kamal Kansou[2], Bert Bredeweg[3,4], Cédric Baudrit[1], Pierre-Henri Wuillemin[5]

[1] INRAE, Bordeaux University-I2M, Talence, France
[2] INRAE, BIA, Nantes, France
[3] Faculty of Education, Amsterdam University of Applied Sciences, Amsterdam, The Netherlands
[4] Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
[5] Sorbonne University, CNRS, LIP6, Paris, France

Abstract

The complexity of analysing dynamical systems often lies in the difficulty to monitor each of their dynamic properties. In this article, we use qualitative models to present an exhaustive way of representing every possible state of a given system, and combine it with Bayesian networks to integrate quantitative information and reasoning under uncertainty. The result is a combined model able to give explanations relying on expert knowledge to predict the behaviour of a system. We illustrate our approach with a deterministic model to show how the combination is done, then extend this model to integrate uncertainty and demonstrate its benefits.

Introduction

Reasoning about a specific system's behaviour requires a good understanding of the involved **entities**, their **quantities** (*i.e.* their relevant numerical properties), how these are related and the value they can take on. Establishing a **model** able to explain those relations and the general behaviour of the studied system is a complex task, hindered further by the introduction of uncertainty: quantities are not always observed and values tainted with errors can frustrate the interpretations.

Bayesian Networks (Pearl 1985) (BNs), thanks to their graphical aspect, allow to understand the underlying probabilistic dependencies between the quantities (denoted as **variables** in this context). However, they can be impaired by the lack of physical understanding. While the models learned with BNs offer a good quantitative description of the studied system, they might lack explainability (*i.e.* their results do not always match human logical reasoning). This is due to the fact that BNs build correlations, and not causation: in practice, a model could learn **Rain→Grass** ("The rain soaked the lawn") as well as **Grass→Rain** ("The soaked lawn provoked the rain"). To prevent such unwanted results, (Pearl 2009) defined interventions (*i.e.* modifying one quantity without touching the others) to construct causal models. This however is not always doable for practical, ethical or economic reasons: for instance, studying the impact of smoking on health would require to intervene on people to force them to smoke.

Integrating external sources of knowledge can be useful to guide the learning and prune impossible models. The most common way of doing so for BNs is to impose a complete (Baudrit et al. 2022) or partial (Munch et al. 2022) structure, built with experts. This structure is denoted as **theory**, as it reflects the experts' (often causal) knowledge over the considered system. This approach helps to select the relevant variables, and reduces the learning to the parameters (the probabilities). However, this raises the question of the correctness and/or completeness of the fed causal theory: depending on the experts, their number, their area of expertise, ... several can be proposed, each with possible distinct impact over the learning.

On the other hand, qualitative reasoning (QR) builds sound models with solid grounding on causality. By reasoning over quantities and defined relations, they can generate all possible states of a system without relying on data (Forbus 2011). Instead, they allow to define **quantity spaces**, in order to consider only relevant values (*e.g.* {Ø, Low, Medium, High}) and to reason on a symbolic level. As such, they give a complete description of the system which can be used to assess the validity of the expert knowledge integrated in the BN's learning.

In this article, we combine BNs with knowledge of the system physics represented as qualitative models (QMs) to learn models able to apprehend uncertain systems with explainable answers. Below, the first section presents the necessary notions and state of the art on QM, BNs and the use of QM for quantitative modeling. The second section presents the principle of our approach illustrated by an example. Finally, the third section compares the results of our approach compared to naive BN learning in order to demonstrate the gain in explainability.

Modeling and simulation have been done using the Dynalearn environment (Bredeweg et al. 2013), which is based on the Garp3 software (https://dynalearn.nl/). BN learning and computing have been done using the PyAgrum library (Ducamp, Gonzales, and Wuillemin 2020).

Background

Qualitative Modeling with Garp3

Garp3 (Bredeweg et al. 2009) defines a qualitative system through (1) the use of entities and their associated quantities and (2) their relations. Quantities are described by their value (magnitude, *e.g.* +) and direction of change (derivative, *e.g.* 0). Values are picked from associated quantity spaces, which holds every possible values they can take. While magnitudes' quantity spaces can be defined as desired by experts, derivatives' are fixed: negative, null or positive. Following Garp3's notation, they are denoted as $\{\blacktriangledown, \bullet, \blacktriangle\}$, or $\{$-, 0, +$\}$. A combination of magnitude and derivative for each quantity (*e.g.* $<0, +>$) defines a **state**, *i.e.* the behaviour of the system at a certain time. Each state is a unique qualitative behaviour of the system, characterized by a unique set of quantity values and derivatives. Passing from one state to the other represents the evolution of the system: a **graph of state** is defined by a graphical representation of all possible transitions between the different states, where each node is a state and the edges the possible transitions.

In order to compute this graph, Garp3's inference engine reasons over two types of qualitative relations which defines causal relations between each other (Forbus 1984): *proportionalities* (changes caused by processes, denoted as P-/+), and *direct influences* (causal propagation of changes, denoted as I-/+). Additional constraints can be added: **correspondences** and **inequalities** allow the user to describe the relations between certain quantity's values and quantities (*e.g.* force the zero value, or force a value to be always higher than another). Finally, reasoning is done over **scenarios**, which define (some) values for the initial state.

Bayesian Networks

Bayesian Networks (BNs) (Pearl 1985) are acyclic graphs G=(V,E), with V and E respectively the sets of all their nodes (representing random variables) and arcs (representing the conditional dependencies). To each variable, a conditional probability table (CPT) is associated, giving the probability distribution for each possible value it can take and how the values of its parents (i.e. variables that have an oriented path toward that variable) influence it (as shown in Fig.1). A joint probability over all nodes V is defined as the product of local probabilities given as:

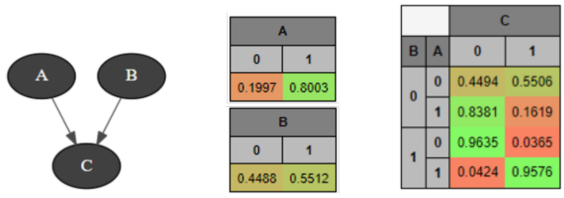
$$P(X_1, ...X_n)=\prod_{i=1}^{n} P(X_i|Pa(X_i))$$

with $P(X_i|Pa(X_i))$ being the conditional probability function associated with random variable X_i, conditioned on its parents $Pa(X_i)$. A probability of 0 describes an impossible event, while a probability of 1 is associated to a certain event.

While models have been proposed to take into account continuity within BN's structure, this article focuses on the discrete part. BN's learning is usually done is two steps: considering a discretized database, the structure G is first learned, then the probabilities. In this study, this last part is tackled, as structure is provided by the addition of expert knowledge from the QM.

Combining Quantitative and Qualitative

Explainable Artificial Intelligence has gained a tremendous attention over the past years (Guidotti et al. 2018), as the need of justifications for supporting a model's predictions is a key-question. More generally, there is an increase in the need of understanding things correctly (*e.g.* science).



(a) BN (b) Marginal Distribution (c) Conditional Distribution

Figure 1: (a) Example of a BN composed of three variables **A**, **B** and **C**. (b) Marginal distributions associated to **A** and **B**. (c) CPT associated to **C**: in this example the probabilities of **C**'s values (columns) depend on **A** and **B**'s values (rows).

Thanks to their graphical component BNs offer explainability for their prediction. However, the lack of causality in their approach leads to inaccurate models, unable to describe real systems. Algorithms such as PC (Spirtes, Glymour, and Scheines 2000) or more recently MIIC (Verny et al. 2017) have been developed to tackle this issue and learn causal structures from data alone. These approaches are however costly in data. On another hand, integrating expert knowledge (*e.g.* as partial node ordering (Parviainen and Koivisto 2013)) during the learning helps reducing the data cost by reducing the search space (Munch et al. 2017). Yet, different causal models lead to different BNs, whose correctness can be difficult to evaluate.

In this article, QM is proposed to define a stable structure able to frame the quantitative reasoning and integrate it into quantitative learning. Such combination has been proposed, for instance to improve simulations based on dynamic equations (Pang, Bruce, and Coghill 2018). In this case, QM allows to define constraints that reduce intervals of simulation for already known equations. It is often proposed to model systems in order to bypass equations and simplify the simulations (Soberl and Bratko 2022; Struss, Reiser, and Kreuzpointner 2018). (Klenk, Nabi, and Arvay 2016) proposes a methodology to compare different explanatory models for co-morbidities, using QM to develop mechanistic explanations. While they do not rely on data, they raise the question of inferences: given a patient and a validated causal model, is it possible to derive conclusions? In the frame of this article, the combination of BNs et QM would allow to answer quantitatively to these questions with probabilities, *i.e.* proposing different possible answers with probabilities of their happenstance. More generally, it aims at answering the three advantages defined by (Forbus and Falkenhainer 1990) for the combination of quantitative simulation with qualitative knowledge: (1) increased automation (*i.e.* no need for manually defining each relevant equation), (2) improved self-monitoring (*i.e.* consistence checking with reality) and (3) better explanations (*i.e.* justifications of predictions based on causal reasoning).

Combining BNs and QMs

This section presents the combination of BNs and QMs as showed in Fig.2, illustrated with a system.

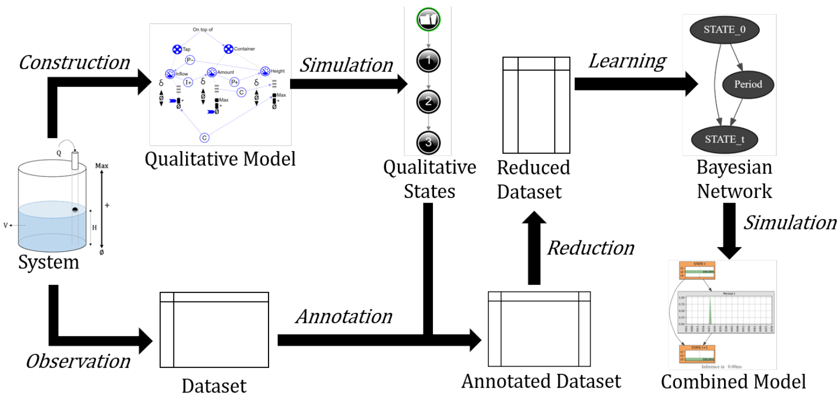


Figure 2: Summary of the approach. From a given system, a QM is constructed then used in a simulation to extract the different possible qualitative states. From a dataset observing the different values of the systems quantities, an annotated dataset is constructed, using the qualitative states to describe the dataset. This dataset is then transformed to allow the learning of a BN.

Example: the Container System

Consider a container with a floating cap being filled with water described by three quantities (illustrated in Fig.3 (a)):

Q The inflow of water going in the container through a tap. Initial flow is denoted Q_{in}.

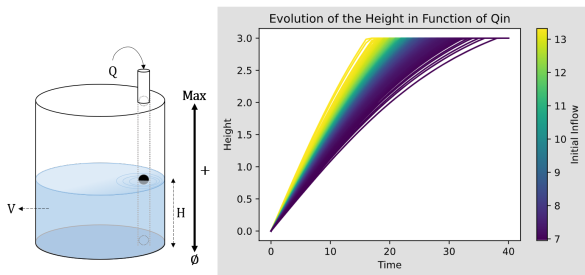
V The current volume of water in the container. Maximum volume is denoted V_{max}.

H The current height of the floating cap. Maximum height is denoted H_{max}.

Starting with an empty container, water is introduced at a given flow, which arbitrarily decreases while height increases. A floating cap is present such that, once the container is filled, it interrupts the flow. For the following, a dataset describing the values of the different quantities through different simulation is considered. Each simulation is initialized using:

- $H_{max} = 3$
- $V_{max} = 3\pi$
- $Q_{in} \hookrightarrow \mathcal{N}(10, 1)$

While this approach is able to address deterministic systems, randomness is introduced to demonstrate its robustness when facing uncertainty. Fig.3 (b) shows the influence of Q_{in} on the filling rate speed.



(a) Container system (b) Impact of Qin

Figure 3: Qualitative modeling of the container system. (a) System. (b) Impact of the initial inflow $\mathbf{Q}_{in}$ value over the time required to reach the maximal height.

Qualitative Model and States

The first step propose for the dynamic of this system a QM. Following the system's description, we consider two objects and three quantities:

- The ***tap***, associated to the **inflow** quantity **Q**. The quantity space is $\{\emptyset, +\}$, referring respectively to the absence and presence of flow.
- The ***container***, associated to the **amount V** of water and **height H** of the cap. Their quantity spaces is $\{\emptyset, +, Max\}$, with $<Max>$ respectively the maximum volume and height. For both, $<\emptyset>$ refers to the null value (*i.e.* no water), while $<+>$ refers to the amount and height in between.

Fig.4 (a) presents the relations existing between the defined objects and variables as well as the initial values used for the simulation: **Inflow** is $<+>$, while the **Amount** is $<\emptyset>$. This creates three possibles states for the system, denoted in the rest of this article as $s1$, $s2$ and $s3$:

$s1$ The tank is empty: water starts flowing through the tap. The volume and height are null, but increasing.

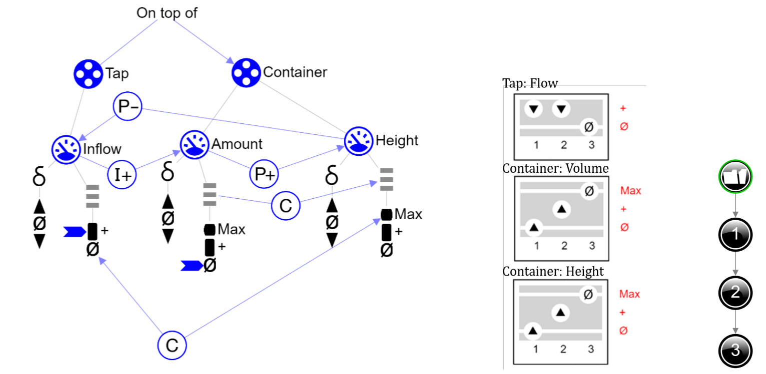
$s2$ The tank is being filled: the volume and height are not null and increasing, while the flow decreases.

$s3$ The tank is filled: water stops flowing. All quantities' derivative are null, the system is at equilibrium.

Table 1 recaps the states different values, while Fig.4 (b) and (c) presents the simulation's results.

Annotated Dataset

While QM reasons over states and transitions between those, quantitative models such as BNs are dedicated to the study



(a) Qualitative modeling of the container system.

(b) Evolution of quantities' values

(c) States

Figure 4: Qualitative modeling of the container system. (a) Model and initial conditions. (b) Evolution of the quantities' values during simulation. (c) State graph of the simulation.

State	**Q**	**V**	**H**
$s1$	$<+, ▼>$	$<\emptyset, ▲>$	$<\emptyset, ▲>$
$s2$	$<+, ▼>$	$<+, ▲>$	$<+, ▲>$
$s3$	$<\emptyset, ●>$	$<+, ●>$	$<Max, ●>$

Table 1: Description of the quantities <Magnitude, Derivative> for each state.

Time	H	V	Q	dH	dV	dQ	State	Period
1	0	0	10	▲	▲	▼	$s1$	1
2	2.9	9.1	7.3	▲	▲	▼	$s2$	1
3	4.6	14.4	3.7	▲	▲	▼	$s2$	2
4	5	15.7	0	●	●	●	$s3$	1
5	5	15.7	0	?	?	?	$s3$	2

Table 2: Example of a discretization using the QM, considering Q_{in}=10, H_{max}=5 and 5 time steps. Since Step 3 is an equilibrium state, we assume that the observation at time 5 still matches state 3, although the derivatives are unknown.

State$_t$	**State**$_{t+1}$	**Period**$_t$	**State**$_t$	**State**$_{t+1}$	**Period**$_t$
$s1$	$s2$	1	$s3$	$s3$	1
$s2$	$s2$	1	$s3$	$s3$	2
$s2$	$s3$	2			

Table 3: Transformation of the database of Table 2 into a database suitable for the BN learning.

variable's evolution across fixed intervals. This difference of focus requires the definition of a novel quantity in order to bridge between the two representations: the **Time Step**. In practice, learning a quantitative model requires values to reflect on; when learning a dynamic model, it helps to have data describing the system at regular intervals (the time steps). This is particularly important as the aim of the final model is to be able to describe precisely the evolution (*i.e.*, the passage or not from one state to the other) of the system: irregular time steps would scramble the predictions. For the following, the time step quantity refers to the time at which the system is described.

Using Table 1 states, each time step is associated to one:

1. By discretizing the quantity's value with its quantity space. For instance, if **Height** = 0, then its discretized value is $<\emptyset>$; H_{max} becomes $<Max>$; otherwise, it is discretized to $<+>$.
2. By looking at the derivative for each quantity: if the difference between the quantity's value at time t and $t+1$ is negative, then the derivative is also *negative* (▼); if the values are equal, then the derivative is *null* (●); if it is positive, then the derivative is *positive* (▲).

In the end, using the combination of the discretized value and the derivative for each quantity, each time step can be associated to a QM state. A new quantity is also introduced for interval of Time Steps: the **Period**. While **Time Step** marks the passing of time, the **Period** indicates how long the system has been in the current state. For the rest of this article, given a quantity $\mathbf{X}$, the variable $\mathbf{X}_t$ denotes its value at time step t. Table 2 shows an example of the whole discretization process.

Reduced Dataset

Since the model aims to learn the evolution of the system, *i.e.* the transitions between steps, a new dataset is composed from the **State**$_t$, **State**$_{t+1}$ and **Period**$_t$. This way, each lines brings information of the system's state, how long it has been this way, and whether it will remain the same (or transition) in the next time step. Table 3 shows the transformation applied to Table 2 in order to be able to learn a BN.

Bayesian Network

Structure Definition Once the database is prepared, a structure is manually defined to guide the BN learning, based on two assumptions:

		STATE t+1		
Period t	STATE t	s1	s2	s3
t022	s1	0.5000	0.5000	0.0000
	s2	0.0000	0.6897	0.3103
	s3	0.0000	0.0000	1.0000

Table 4: Excerpt of the CPT showing the probabilities of passing from $\textbf{State}_t$ to $\textbf{State}_{t+1}$ if $\textbf{Period}_t$ = 22.

- The $\textbf{Period}_t$ value depends only on the value of $\textbf{State}_t$;
- The probability of passage from $\textbf{State}_t$ to $\textbf{State}_{t+1}$ depends on $\textbf{State}_t$ and the $\textbf{Period}_t$;

This defines the following structure: $\textbf{State}_t \rightarrow \textbf{State}_{t+1} \leftarrow \textbf{Period}_t \rightarrow \textbf{State}_t$.

Parameters Learning Once the structure defined, parameters are learned through a statistical learning whose goal is to maximize the likelihood by estimating the probability of an event according to its frequency in the considered database. In case an event is never observed (*e.g.* if the system never stays more than one time step in $s1$, then the combination $\{\textbf{State}_t{=}s1, \textbf{Period}{=}10\}$ is never observed), the probabilities are by default equiprobable: all possible outcomes are considered as likely. The learned BN thus encompasses the QM model in its structure, and heavily depends on the data only for its parameters.

Combined Model

In this article, two applications are presented in order to demonstrate the reasoning offered by the learned model:

1. **State Prediction.** Reading the CPT, the probability of passing from one state to the other knowing the period can be deduced. Table 4 presents an excerpt, focusing on the passage from one step to the other after a period of 22 ($\textbf{Period}_t$ = *22*). It shows that depending of $\textbf{State}_t$, the most probable value of $\textbf{State}_{t+1}$ depends: if $\textbf{State}_t = s2$, then it has a probability of 0.69 of staying $s2$; on another hand, if $\textbf{State}_t = s3$, then it will stay $s3$ (which is logical, since it is a equilibrium state). To be noted, if $\textbf{State}_t = s1$, then the probability of transitioning is equiprobable between $s1$ and $s2$ ($s3$ is not considered as $s1{\rightarrow}s3$ is not possible according the state graph): this is due to the fact that the system has no information about cases where a system has stayed 22 time steps in $s1$.

2. **Period Prediction.** Another way of exploiting the probabilistic relations is to make inferences: knowing the value of some variables, it is possible to compute the most probable values of the others. Fig.5 presents such an example: knowing that $\textbf{State}_t = s2$ and $\textbf{State}_{t+1} = s3$ (in orange to indicate it is *observed*), the most probable period (in grey to indicate it is *computed*) is 22.

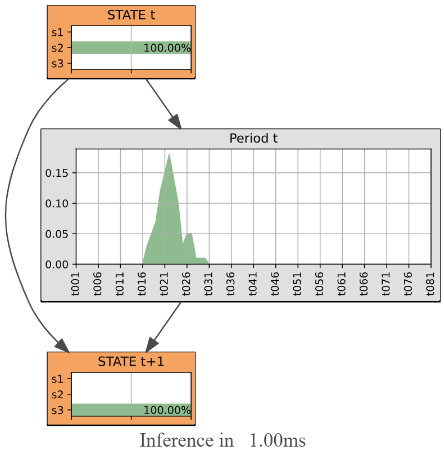


Figure 5: Example of an inference using the BN.

Comparison to Naive BN[1]

In order to evaluate its performances, the combined model is compared to naive BNs. To do so, two naive BNs integrating different expert constraints are presented. Evaluation is done for both (1) the graph and (2) predictions, which are compared against a ground truth.

Naive BNs Learning

"Naive" refers to the fact that the QM's model is not known during the learning, the main difference being that state knowledge is not taken into account. This section presents two versions, with different degree of the model's understanding:

1. Learning is approached with no information from the system at all. Discretization is made using quantiles (instead of the system space's values), and the structure is constrained only so that variables from the past ($\mathbf{Q}_t$, $\mathbf{V}_t$, $\mathbf{H}_t$) can be used to predict the future ($\mathbf{Q}_{t+1}$, $\mathbf{V}_{t+1}$, $\mathbf{H}_{t+1}$), but not the contrary. This approach represents the most naive learning, and gives an idea on how BNs handle this kind of data without prior knowledge of the system. It is denoted as the unguided approach.
2. A second learning is made to include more system's knowledge. The QM's space's values is used for the discretization, and the structure is forced in order to transcribe the expert knowledge used in the QM. This approach is denoted as the guided approach.

Unguided Approach Seven variables are considered: six to capture the values of the quantities **Q**, **V** and **H** at times t and $t+1$, and one to capture the **Time Steps**. To be noted, the **Time Steps** variable in this context is different from the $\textbf{Period}_t$ one presented until now: since states are not known, time refers here to the beginning of the simulation, and not to the time passed in a certain state. The structure

[1] All code used in this article are available at https://gitlab.com/melanie.munch/qr23-submission

is learned through a classical greedy algorithm (Chickering 2003), with the only constraint that variables at time t can explain variables at time $t+1$, but not the contrary (temporal constraints). Discretization is done such that (1) **Q**, **V** and **H** are discretized in 5 quantiles; (2) V_{max} and H_{max} are a 6th category in order to capture when the tank is filled; **Time Steps** is not categorized to keep track of the time as precisely as possible.

Guided Approach This approach still considers seven variables, but handles them differently. First, discretization is done following the QM's space values; secondly, structure is oriented so that additionally to the temporal constraints forced in the naive version, it also takes into account (1) the expert knowledge integrated in the QM (**Q**→**V**→**H**), (2) the influence of the **Time Steps** variable over the values measured at time t and (3) for each variable its value at time t to predict its value at $t+1$.

Simulations

Given a database of 1000 simulations, three models are learned using the same sample of 100 experiments:

1. **Combined Model:** A model learned using the method presented in the previous section.
2. **Unguided Model:** A model learned using the unguided approach presented in this section.
3. **Guided Model:** A model learned using the guided approach presented in this section.

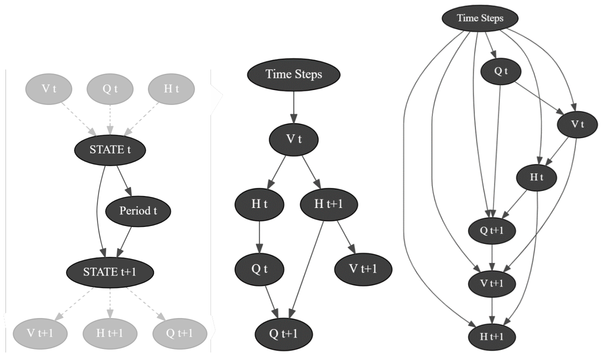
The database of 1000 experiments represents the ground truth that the learned models aim to reach.

Graph Evaluation

Result of the learning are presented in Fig.6. For the sake of explainability, variables **Q**, **V** and **H** have been represented in the combined model (a), so that it can be compared to the other structures. Since it was learned without knowledge, unguided structure (b) differs the most from the QM structure of Fig.4 (a), leading to non causal relations (*e.g.*, $\mathbf{V}_t$→$\mathbf{H}_t$→$\mathbf{Q}_t$). As such, the learned relation are not able to explain the system in a causal way: it only displays correlations, and cannot generate sound explanations to justify the model's prediction. On another hand, the guided approach (c) presents a structure coherent with the QM. Moreover, on the contrary of the combined structure, it directly displays the relations between the variables, instead of having them hidden between the states transitions. While this is an advantage in term of readability for systems with only a few variables, this can become a hassle when considering bigger systems.

Predictions Evaluation

Generation For each model, 1000 simulations are done using the principle illustrated in Fig.7: starting from the same initialization ($\mathbf{Q}_t$>0, $\mathbf{V}_t$=0, $\mathbf{H}_t$=0, **Time Period/Period**$_t$=1), marginal laws for the next step are computed (*i.e.*, probabilistic distribution for the possible values). Using these laws, new values for the variables $\mathbf{Q}_{t+1}$, $\mathbf{V}_{t+1}$ and $\mathbf{H}_{t+1}$ are drawn. If either $\mathbf{H}_{t+1}$ takes the maximal value



(a) Combined Structure (b) Unguided Structure (c) Guided Approach Structure

Figure 6: Models' structures comparison.

or the number of time steps exceed 100, then the simulation is finished. In the first case, the number of time steps is kept; in the second, it means that the model could not reach the end of the simulation and thus the run is incorrect. Frequency of the number of time steps required to conclude the simulation (*i.e.* to fill the container) are compared to the frequency measured in the initial dataset.

Results Results are presented in Fig.8. The first notable thing is that despite the fact that only 10% of the original dataset was used to learn the models, all models have an average time of filling close to the ground truth's. A Kolmogorov-Smirnov goodness of fit test is performed in order to compare each distribution to the baseline: H_0 means that both distributions are identical, while H_1 means they are distinct. H_1 is rejected for both combined and guided models (with p-values respectively of 0.6 and 0.3), while it is validated by the unguided approach (p-value$<10^{-5}$). This means that the unguided approach did not manage to capture the underlying distribution of the dataset. On another hand, both combined and guided are statically indistinguishable, both having an average expectancy of time steps (*i.e.* the average time taken to fill the container) close to the truth's (respectively 24.3 and 23.2 against 24.7).

The main difference between combined and guided models lies in the evolution of the different values. Fig.9 shows three independent simulation results for each model. Guided and unguided are characterized by (1) a decorrelation between the three variables (*e.g.* **V** reaches *Max* value before **H**); and (2) impossible evolution of the values (*e.g.* **Q** increasing). This shows that even if the guided model is close in structure and (for this particular problem) of the ground truth's predictions, it fails at providing an explanation grounded into the causal model. Combined model, on the contrary, is able to provide a description of the system which is consistent with the QM.

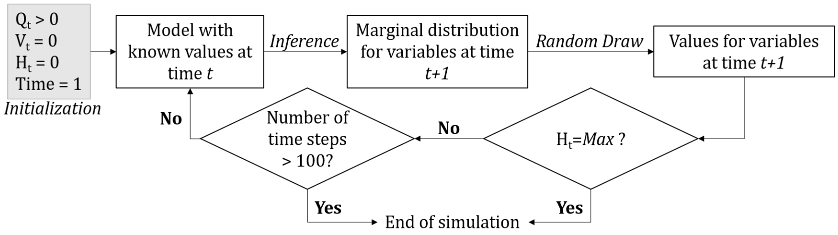
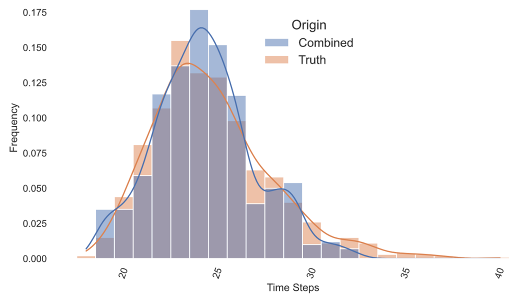
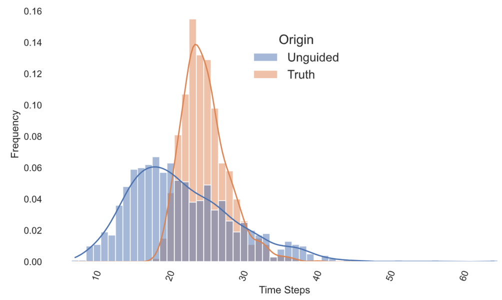


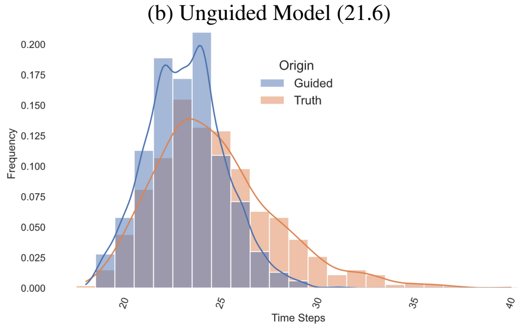
Figure 7: General flowchart for the simulations. To be noted, in the case of the combined model, variables are encompassed in the **State** variable and **Time** is replaced by the **Period** variable.



(a) Combined Model (24.3)



(b) Unguided Model (21.6)



(c) Guided Model (23.2)

Figure 8: Frequency of the times taken to finish the simulation (average number of time step). Truth has an average number of time step of 24.7.

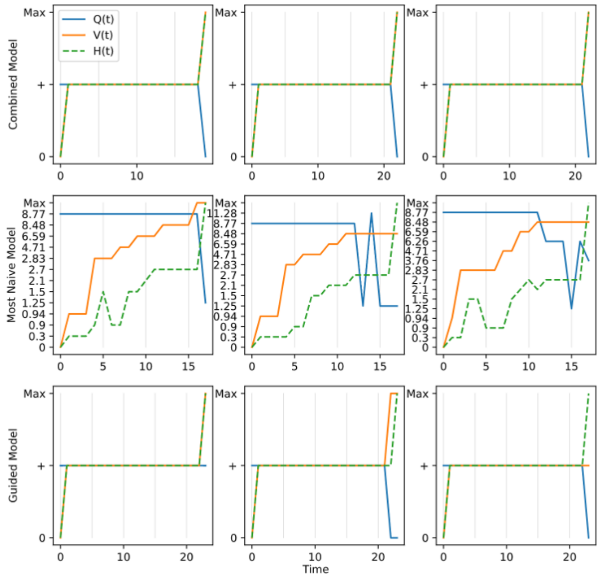


Figure 9: Example of three independent simulations for each model.

Discussion

Comparison to naive BNs has shown that grounding causal knowledge from QM to BN's learning results in a model able to provide simulation close to the reality of the studied system. This is particularly due to the distinction between **Period** and **Time Steps** quantities: while the combined model is able to reason only on state transitions, naive models can only consider total times. As such, they cannot reason about the state they are in, but only how long the simulation has been running. In this simulation, the simplest case was considered, as only $s2$ had a non-constant time: $s1$, for instance, always lasts one time step. Further experiments should be done on systems with more complex state graphs (*e.g.* with cycles and branching paths), in order to assess whether the combined model can adapt. While the approach presented in this article only requires a dataset and a QM as inputs, more work should be done on its automation. More broadly, it should be interesting to see how the combined model can scale on systems with more quantities. Especially, it is important to also take into account the challenges brought by introducing more quantities, as some could be uncertain or missing from the dataset: states could then be uncertain as well, if not all quantities are known.

Another lead to explore would be to use the combine model to assess the adequacy between a theory and a dataset. (Kansou et al. 2017) proposes two tests to define whether a model can be well described by a QM or not: the encompassment (the adequacy between the QM and the dataset) and the sufficiency (the adequacy between the QM and the model's behaviour). To pass this verification, it is important to consider technical aspects:

- The choice of time steps has an influence: if too great, it is possible to skip some combinations of value (and thus states) when annotating the dataset. This would result in a model not respecting the state graph. For instance, if passing from states takes 2 time steps ($s1 \rightarrow s1 \rightarrow s2 \rightarrow s2 \rightarrow s3$), then having a time step of 3 would lead to a model predicting a passage from 1 directly to 3 ($s1 \rightarrow s3$).
- Another critical point is the computation of derivatives. The same way the choice of time steps influences the model's learning, data's sensibility can influence the derivatives' precision. Indeed, depending of the precision of measurement, zero derivative can be hard to catch, as it usually concerns one data point.

Finally, it is important to consider that the data depends on multiple parameters not represented as quantities in the QM. For instance, the required time to fill the container depends on V_{max} (maximal height and radius): if a model is learned only on high and/or large containers, its predictions will not be relevant for smaller containers.

Conclusion

In this article, a new approach of combining BNs with QM has been presented, with the goal of improving BN's modeling by integrating expert knowledge. Comparison with naive BNs has displayed better results for the combined model in term of prediction and explainability. In conclusion, the resulting model is able to provide explainable answers and simulations over an uncertain system. The learning is based only on a dataset and the expert knowledge encompassed in the QM, which dispenses the modeller with the prior definition of system equations.

References

Baudrit, C.; Buche, P.; Leconte, N.; Fernandez, C.; Belna, M.; and Gésan-Guiziou, G. 2022. Decision support tool for the agri-food sector using data annotated by ontology and Bayesian network: a proof of concept applied to milk microfiltration. *International Journal of Agricultural and Environmental Information Systems*, 13(1).

Bredeweg, B.; Liem, J.; Beek, W.; Linnebank, F.; Gracia, J.; Lozano, E.; Wissner, M.; Bühling, R.; Salles, P.; Noble, R.; Zitek, A.; Borisova, P.; and Mioduser, D. 2013. DynaLearn - An Intelligent Learning Environment for Learning Conceptual Knowledge. *Ai Magazine*, 34: 46–65.

Bredeweg, B.; Linnebank, F.; Bouwer, A.; and Liem, J. 2009. Garp3 — Workbench for qualitative modelling and simulation. *Ecological Informatics*, 4(5): 263–281.

Chickering, D. M. 2003. Optimal Structure Identification with Greedy Search. *J. Mach. Learn. Res.*, 3: 507–554.

Ducamp, G.; Gonzales, C.; and Wuillemin, P.-H. 2020. aGrUM/pyAgrum : a Toolbox to Build Models and Algorithms for Probabilistic Graphical Models in Python. In *10th International Conference on Probabilistic Graphical Models*, volume 138 of *Proceedings of Machine Learning Research*, 609–612. Skørping, Denmark.

Forbus, K. D. 1984. Qualitative process theory. *Artificial Intelligence*, 24(1): 85–168.

Forbus, K. D. 2011. Qualitative modeling. *WIREs Cognitive Science*, 2(4): 374–391.

Forbus, K. D.; and Falkenhainer, B. 1990. Self-Explanatory Simulations: An Integration of Qualitative and Quantitative Knowledge. 380–387.

Guidotti, R.; Monreale, A.; Ruggieri, S.; Turini, F.; Giannotti, F.; and Pedreschi, D. 2018. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.*, 51(5).

Kansou, K.; Rémond, C.; Paës, G.; Bonnin, E.; Tayeb, J.; and Bredeweg, B. 2017. Testing scientific models using qualitative reasoning: Application to cellulose hydrolysis. *Scientific Reports*, 7.

Klenk, M.; Nabi, M.; and Arvay, A. 2016. Challenges in Formulating Explanatory Models for Co-morbidities.

Munch, M.; Buche, P.; Dervaux, S.; Dibie, J.; Ibanescu, L. L.; Manfredotti, C.; Wuillemin, P.-H.; and Angellier-Coussy, H. 2022. Combining ontology and probabilistic models for the design of bio-based product transformation processes. *Expert Systems with Applications*, 203: 117406.

Munch, M.; Wuillemin, P.-H.; Manfredotti, C.; Dibie, J.; and Dervaux, S. 2017. Learning Probabilistic Relational Models Using an Ontology of Transformation Processes. In Panetto, H.; Debruyne, C.; Gaaloul, W.; Papazoglou, M.; Paschke, A.; Ardagna, C. A.; and Meersman, R., eds., *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*, 198–215. Cham: Springer International Publishing. ISBN 978-3-319-69459-7.

Pang, W.; Bruce, A.; and Coghill, G. 2018. Non-constructive interval simulation of dynamic systems.

Parviainen, P.; and Koivisto, M. 2013. Finding Optimal Bayesian Networks Using Precedence Constraints. *Journal of Machine Learning Research*, 14: 1387–1415.

Pearl, J. 1985. Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning. In *Proc. of Cognitive Science Society (CSS-7)*.

Pearl, J. 2009. *Causality: Models, Reasoning and Inference*. New York, USA: Cambridge University Press, 2nd edition. ISBN 052189560X, 9780521895606.

Soberl, D.; and Bratko, I. 2022. Qualitative control of cart and pole on uneven terrain.

Spirtes, P.; Glymour, C.; and Scheines, R. 2000. *Causation, Prediction, and Search*. MIT press, 2nd edition.

Struss, P.; Reiser, R.; and Kreuzpointner, F. 2018. A Qualitative Gait Model for Diagnosis of Juvenile Idiopathic Arthritis.

Verny, L.; Sella, N.; Affeldt, S.; Singh, P. P.; and Isambert, H. 2017. Learning causal networks with latent variables from multivariate information in genomic data. *PLOS Computational Biology*, 13(10): e1005662.

KNACK v2: Using Analogical Generalization over Qualitative Representations for Quantitative Estimation

Walker Demel, Will Hancock, & Kenneth D. Forbus
Qualitative Reasoning Group, Northwestern University
2233 Tech Drive, Evanston, IL, USA 20208
walker.demel@northwestern.edu, wwhancock@u.northwestern.edu, forbus@northwestern.edu

Abstract

One of the roles of qualitative representations is to provide context for numerical information, making explicit how it is grounded in the world. This supports tasks like quantity estimation, e.g. estimating the cost of a used bicycle by comparing it with similar items. The KNACK model (Paritosh & Klenk, 2006) used analogical retrieval of a fixed number of cases to perform such estimates. This paper describes a new algorithm, KNACK v2, which uses analogical generalization to provide a more robust notion of context for quantitative estimation. We describe how KNACK v2 works and test its performance on a dataset of country information from Wikidata, showing it is competitive with linear regression while providing explanations.

1 Introduction

Quantity estimation is integral to our everyday lives. We may estimate how long it would take to commute home if we stop at the grocery store on the way, whether we have enough fuel to drive to our destination, or how much we should charge for our used bicycle after upgrading to a new one. Solving these estimation problems typically requires some experience with similar examples as well as domain-knowledge about the world.

Before setting the asking price for our old bicycle, we need a contextual sense of what bicycles cost in our environment. We might browse online listings of used bicycles or stop by a used bike shop in our town to get a general idea of the distribution. These serve as reference points for generating our own estimate, or in this scenario, asking price.

During quantity estimation we also regularly use our domain-specific qualitative and quantitative world knowledge. For example, we know that bicycles with a sophisticated multi-gear system are more costly than those without one. The weight of the frame or the thickness of the tires may also be factors that influence our estimate.

The dominant computational model for estimating quantities is multiple linear regression, but this approach has two drawbacks. First, linear regression does not handle qualitative information adeptly. The classic workaround solution is to create one-hot dummy variables that are active when a case has a given feature and inactive when it doesn't. In our bicycle example, the presence or absence of a gear-shifting system would be represented by a 1 or a 0 in a dedicated dimension. This approach can lead to sparsity in feature vectors and subsequent overfitting. The second drawback of pure regression is its lack of explainability. A regression output is simply an intercept and a series of coefficients for associated dimensions. There is no dependency, no higher-level cognitive mechanism that guarantees a reasonable estimate, and no clear explanation for why a given estimate makes sense. Returning to our used bicycle example, negotiations over price often hinge on specific factors (e.g. fancier gear-shifting system versus more wear), so an explainable model would likely give customers more peace of mind that they are getting a fair price.

This paper describes KNACK v2, a model for quantity estimation based on qualitative representations and analogical generalization. We start by discussing relevant background, including the anchoring and adjustment psychological model of quantitative estimation, our analogical processing models, and the construction of qualitative representations of quantities via CARVE (Paritosh 2004). Then we describe the KNACK v2 algorithm, and an experiment using a dataset extracted from Wikidata (Vrandecic & Krotzsch 2014). The experiment provides evidence that KNACK v2 is competitive with linear regression, but with the ability to provide explanations. We close with conclusions and future work.

2 Background

2.1 Anchoring and Adjustment

There has been significant psychological evidence for the heuristic of *anchoring and adjustment* (Tversky and Kahneman, 1974). This method for quantity estimation involves two steps. The first step anchors an estimate by retrieving a relevant example from memory and using its value for that quantity. This retrieval can be a prototypical class instance (subject to the availability heuristic (Tversky and Kahneman 1974)) or a similar example. For instance, when estimating the rent for an apartment, we may start with the rent for apartments of the same configuration (e.g. one bedroom) in the same neighborhood. Using that sample as our estimate would be a type of nearest neighbor sampling, but we can often be more accurate by utilizing *adjustment*. This second step incorporates our intuitive heuristic knowledge of the world to scale up or down our estimate.

We use two ideas in developing computational models based on anchoring and adjustment. The first is the structure-mapping theory of analogy and similarity (Gentner, 1983) to both find similar examples and compute how they are aligned with the current situation. In structure-mapping, similarity is based on structured representations, including relationships between entities as well as attributes (aka features). There is ample evidence that this model is more psychologically plausible than purely feature-based approaches (e.g. Markman & Gentner, 1993). Returning to the rental example, when trying to estimate the rent for one apartment, we may retrieve another apartment—whose rent we do know—and map the two cases together with their relative parts, comparing configuration with configuration, location with location, price with price, etc. These alignable properties help provide the grist for adjustment: If one apartment is larger than the other, then that suggests its rent might be higher. Qualitative representations provide this kind of causal information needed to drive adjustment. We use *qualitative proportionalities* (Chapter 7, Forbus 2019), which describe how quantities are causally connected with one another. If rent is qualitatively proportional to square footage, then an apartment with more square footage will have a higher rent, all else being equal. Of course, what makes these problems difficult is that all else typically is not equal: A small apartment in a great neighborhood may be more expensive than a huge apartment in an unsafe neighborhood.

This approach is broadly compatible with psychological evidence about component processes. Previous studies have found that relational retrieval improves with domain expertise (Blanchette & Dunbar 2001; Novick 1988; Gentner, Loewenstein, & Thompson 2004). Similarly, the adjustment phase of quantity estimation gets better with expertly tuned heuristics and knowledge of qualitative proportionalities and other quantity relationships (Paritosh & Klenk 2006).

2.2 Analogical Processing

We draw on computational models for three processes involved in analogical learning and reasoning, matching, retrieval, and generalization, discussing each in turn.

Matching is performed by the Structure-Mapping Engine (SME; Forbus et.al. 2017). It takes two cases as input, both structured representations that include both statements about object attributes (e.g. being a bicycle) and relationships (e.g. that the basket is connected to the rear wheel of the bicycle). It constructs one or more *mappings*, each of which consists of three parts. (1) A numerical score indicates the overall quality of the match. This depends on properties such as the nested overlap in relationships, thereby capturing human preferences for arguments and explanations, (2) a set of correspondences, indicating what objects and statements align with each other. Correspondences can be used in supporting how an example is relevant to a situation, among other things. (3) A set of candidate inferences, indicating how non-aligned information in the base or target might be mapped onto the other description, based on the correspondences. These provide conjectures and highlight salient differences between the two descriptions.

Retrieval of cases is modeled by MAC/FAC (Forbus, Gentner & Law 1995). The probe is the case for which a reminding is sought from a case library consisting of structured representations. For scalability, MAC/FAC consists of a two-stage process, both of which use map/reduce. The MAC stage computes a coarse estimate of the probe with every case in the library, in parallel, based on *content vectors*. Content vectors are automatically constructed from structured representations, with the strength of a dimension related to the number of occurrences of each kind of predicate, attribute, or logical function. The dot product of two content vectors is an estimate of SME's structurally grounded similarity score. The best M matches from the MAC stage are passed into FAC, which uses SME for its comparisons, producing the best N matches as outputs.

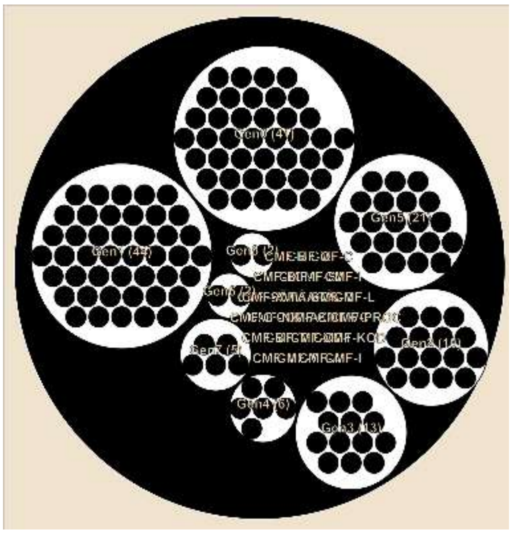


Figure 1:
A Sage Generalization Pool. Generalizations (groups of analogically similar cases) are shown in white circles, with individual constituent cases shown as black dots.

Generalization, the process by which we naturally group similar cases together, is modeled by Sequential Analogical Generalization Engine (SAGE) (Kandaswamy & Forbus 2012). SAGE builds analogical models of concepts incrementally, using structure-mapping as a clustering metric. Each model consists of a *generalization pool*, which can contain both generalizations and outliers (Figure 1). Given a new example of a concept, MAC/FAC is used to retrieve the most similar item, treating the pool as a case library. If the similarity score produced by SME is higher than the *assimilation threshold* for that generalization pool, then the case and the item are assimilated. If the item was an outlier, then a new generalization is formed by merging the corresponding statements.

Generalizations also record summative statistics about constituent cases. For example, in a generalization composed of two countries, facts they share (high population, medium GDP, etc.) will have probabilities of 1, while facts that exist in only one constituent case have probability 0.5. Non-identical entities are replaced by skolems constants called *generalized entities*. If the item was a generalization, the merge process updates the probabilities for the statements based on overlap, and introduces new generalized entities as needed. At any time, the generalizations and outliers in the pool constitute a disjunctive model of that concept given the data so far. It is analogous to k-means clustering with outliers, except that the clustering metric is structure-mapping and the number of clusters is determined automatically based on the data. This ability to handle disjunctive concepts provides a finer-grained notion of context for reasoning, e.g. racing bicycles will likely end up in different clusters from cargo bicycles.

Currently, our analogy stack (MAC/FAC, SME, and SAGE) is not sensitive to quantity; that is, the analogy stack was built primarily for cognitively plausible, qualitative reasoning over relational cases, rather than numerical analyses. In order to make analogy sensitive to attributes on quantitative dimensions, we employ a model called CARVE.

2.3 Qualitative Representation of Quantities

Structure-mapping operations are not sensitive to numerical values. For example, the difference between apartments with 700 and 705 square feet is the same to SME as the difference between apartments with 700 and 1000 square feet. We take this as a job for qualitative representations: In apartments, 5 square feet is a negligible difference. In an engineering analysis of materials needed for an aircraft, five extra square feet can be a considerable difference. Thus we argue that translation to appropriate qualitative values, in a task-specific manner, is a sensible and psychologically plausible way to incorporate such information.

In Qualitative Process (QP) theory (Forbus, 1984), limit points are used to distinguish ranges in numerical values based on when the underlying causal laws change. But what about situations where either it isn't known yet which causal laws are relevant yet, or even what they are? Paritosh (2004) proposed using *distributional limit points*, dividing numerical ranges into a discrete set of values via classic k-means clustering. For example, population might be initially divided into three bins, High, Medium, and Low. Once distributional limit points have been computed, numerical facts can automatically be converted to qualitative statements. For example,

```
(populationOfRegion unitedStatesOfAmerica
  (UnitOfCountFn Person) 331000000)
```

becomes

```
(isa UnitedStatesOfAmerica (CountryTypeFn
  (MediumAmountFn CountryPopulation)))
```

Where the literal value is replaced by the qualitative label (medium) within the broader case library context (all countries). Thus countries that are qualitatively similar in relevant dimensions are more likely to be retrieved. Significant differences in quantities are highlighted via candidate inferences generated during the mapping process.

CARVE (Paritosh 2004), uses k-means clustering to introduce distributional limit points and then used a precursor to SAGE to look for useful partitionings. At the time, the paucity of available data limited experimentation. With modern Semantic Web data sources, that has changed. The experiments described here use the CARVE algorithm with three qualitative values to symbolize quantity.

3 The KNACK v2 Algorithm

KNACK v2 is an algorithm for quantitative estimation using analogical generalization over qualitative representations. It takes a stream of examples to learn analogical models via SAGE, as described in Section 2.2. Figure 2 describes the algorithm for ingestion of new examples, and Figure 3 describes how estimations are made, given the current state of the generalization pool. We discuss each in turn.

Algorithm: Ingest Example
Given example E and generalization pool GP,
1. Convert all quantitative values in E to qualitative values
2. Add E to GP via SAGE

Figure 2: KNACK v2 Ingestion Algorithm

The example ingestion process (Figure 2) is straightforward. All statements involving numerical parameters are replaced with qualitative statements, as per the example above. This has the effect of flattening the representation to some degree, since it is replacing relations (e.g. `populationOfRegion`) with attributes (e.g. `(CountryTypeFn (MediumAmountFn CountryPopulation))`), which has the effect of making analogical retrieval sensitive to differences in values, as desired.

Quantity estimation can be viewed as a form of anchor and adjustment. Step 1 in Figure 3 retrieves the anchor. As per Step 1(a), if nothing is retrieved, the average of Q across the examples in the pool is used as a fallback. If the closest anchor is an outlier, then there isn't enough information to build a linear regression model, so the value of Q in the outlier is used instead (Step 2). Step 3 is the interesting case. As noted above, qualitative proportionalities provide the kind of partial causal constraints that can be assembled to form a model for a quantity. We assume the retrieval of relevant qualitative proportionalities (Step 3(a)) is done respecting the constraints of a QP domain theory. Steps 3(b-d) does the adjustment, by constructing and using a linear regression model based on the examples in the retrieved generalization. One subtlety concerns missing data in examples: If an example is missing data, it is thrown out, and if none of the examples in I have relevant data, the marginal average across the pool is used instead as a fallback.

The use of generalization to provide a more focused context is the key innovation of KNACK v2. The original version of KNACK used MAC/FAC over a case library of examples, looking for a hard-coded number of examples—5— to use in model construction. By using analogical generalization instead, we are assured that the cases are all reasonably similar to each other, as opposed to being just the most similar that could be found. Thus this algorithm scales smoothly between low-data situations (e.g. two examples) and high-data situations (e.g. dozens of examples). This does raise the question of what should be done with generalizations that have thousands or even millions of examples. Such situations have never arisen, but if they do, one approach would be incrementally computing more summative statistics rather than keeping everything in the original cases.

Algorithm: Estimate
Given: New example E with quantity Q to be estimated, with respect to generalization pool GP

1. Retrieve closest item I from GP, using MAC/FAC
    a. If no retrieval, return marginal average of Q across all cases in GP
2. If I is an outlier, use the value of Q in I as the estimate.
3. If I is a generalization,
    a. Let qprops = {qualitative proportionalities constraining Q}
    b. Let $a_1,\ldots,a_n$ be the antecedent quantities from qprops.
    c. Construct linear regression model from values for $a_1,\ldots,a_n$ using the cases used to produce I
    d. Produce estimate from linear model, computing Q from data for $a_1,\ldots,a_n$ from E.

Figure 3: KNACK v2 Estimation algorithm

At the time of the original KNACK's publication in 2006, the landscape of open-source datasets was very different. Prior to the machine learning boom of the 2010s, datasets for learning were more often smaller and experiment-specific. The datasets used with the original KNACK algorithm, for example, contained 15 cases (each case representing one basketball player). Datasets have ballooned in size since this time, and access is often easy and free (Forbus & Demel, 2022). Thus to test the scalability of the KNACK v2 algorithm, we generated a new dataset using Wikidata, one of the largest open knowledge graphs available.

[1] NextKB is available at qrg.northwestern.edu, and we will make the country dataset available on the web as well to support replication.

4 Experiment

We generated a dataset describing information about 197 countries, and used KNACK v2 to build models for quantity estimation. We start by summarizing Wikidata and how we translated the data into our representations[1]. Then we describe our experimental method and the results.

4.1 Wikidata

Wikidata is a collaboratively edited knowledge graph hosted by the WikiMedia foundation (Wikipedia, Wiktionary, etc.) Utilizing an extensive distributed community of editors, Wikidata has grown to over 104 billion items at the time of writing.[2] The open-source nature of Wikidata allows it to serve as a downstream aggregate of otherwise siloed data from various sources. For example, Wikidata contains data from the Google Books initiative as well as the Vatican Library, linking common entities across domains. We briefly describe the structure of Wikidata items.

Wikidata items are entities with a unique identifier (QID) and a set of statements concerning them. Each statement is a key-value pair, with the key being a property (associated with a unique property ID, or PID) and the value being some value—a quantity, another item, or multimedia like a photo. This structure is effectively a series of triples of the form <subject, predicate, attribute>. This RDF structure makes all of Wikidata queryable from a SPARQL endpoint.[3] For example, say one wants to find the capital of the United States. The United States is an item in Wikidata with the QID Q30. There is a *capital* property with the PID P36. Then all we have to do is query for the statement <Q30, P36 ?X> in SPARQL, giving us another entity, Washington D.C. (Q61).

But statements can be more sophisticated than linking multiple items. Some predicates, like *area* (P2046), link an item to a quantity, margin of error, and a unit. (According to Wikidata, the United States (Q30) has an area (P2046) of 9,826,675±1 square kilometers.) Other facts have qualifiers attached—between 1785 and 1790, the capital of the United States was New York City. Similarly, the value for a population statement is constrained by the year when it holds true. Finally, most facts in Wikidata can be traced back to their source through citations or provenance information, increasing the trustworthy of the data available.

We queried Wikidata for 197 countries and their associated statements. We gathered both qualitative data, like:

- Bordering Countries
- Continent Membership
- Currency
- Bordering Bodies of Water
- International Organization Membership
- Language Spoken

Along with quantitative data, such as

- Area
- Population
- Human Development Index (HDI)

[2] For up-to-date statistics on items, edits, and users, visit https://www.wikidata.org/wiki/Special:Statistics

[3] query.wikidata.org

- Development Index
- Gross Domestic Product (GDP)
- GDP Per Capita
- Literacy Rate
- Fertility Rate
- Life Expectancy
- Median Income
- Democracy Index.

Due to the crowdsourced nature of Wikidata[4], not all cases are complete with every dimension. For example, no literacy rate was found for Mexico, and no median income found for Mauritius. Wikidata had only 4 of 11 possible quantitative facts for Monaco: area, population, GDP, and GDP per capita. This makes our estimation task more difficult but is inevitable in real-world situations.

To build our dataset, facts retrieved via SPARQL queries were automatically translated into the OpenCyc ontology used in NextKB, our knowledge base. For example, a population fact in Wikidata looks like

```
<U.S. (Q30), Population (P1082), ~331Million>
```

This is translated to this CycL sentence:

```
(populationOfRegion unitedStatesOfAmerica
  (UnitOfCountFn Person) 331000000)
```

Since this dataset will be used for analogical estimation, we need to have an understanding of what it means for two countries to be analogically similar to one another. There are qualitative similarities: if they are a part of the same continent, in the same international organizations, use the same currency, or share cultural similarities like the language spoken. There are also quantitative similarities. They may have similar populations or areas, or their Human Development Indices may both be between 0.8 and 0.9. Consequently, we used CARVE to generate qualitative representations of quantitative dimension facts using three qualitative distinctions to generate facts like

```
(isa Poland
  (CountryTypeFn (LowAmountFn Area)))
(isa Spain
  (CountryTypeFn
    (MediumAmountFn CountryGDP)))
(isa UnitedStatesOfAmerica
  (CountryTypeFn
    (HighAmountFn CountryGDP)))
```

4.4 Experimental Method

The dataset we built contains 197 country cases, each consisting of 2 to 91 facts, with a mean of 38. Each experimental fold consisted of holding out 19 or 20 cases for testing while the model learned (generalized) the remaining ones. The assimilation threshold for SAGE during the learning phase of KNACK v2 was set at 0.8, requiring strong match strength between a test case and a given generalization. The dimensions to be estimated were GDP, Human Development Index (HDI), and Democracy Index (DI). The qualitative proportionalities involving them are shown in Table 1.

DIMENSION	DEPENDS ON
GDP	Population
HDI	Life Expectancy
DI	HDI

Table 1. World knowledge is built into the model of dimensional dependence. Dimensions in the right column were used as independent variables when regressing on a generalization.

Measuring accuracy is subtle given the varying nature of these quantities. Gross Domestic Product is an unbounded quantity that ranged from 39,000 to 19 trillion US Dollars. Accuracy for GDP was measured by distance away from ground-truth values, scaled by the magnitude of the ground truth itself.

$$|truth - estimate| \,/\, truth$$

where *truth* is the ground-truth fact, and estimate is the output generated by KNACK v2. This was done according to Weber's law (Fechner 1966), which states that perceived similarity of quantities is measured by a ratio between them, i.e. although 1,000 and 1,001 are the same distance apart as 1 and 2, the former pair is judged to be closer together because the ratio of the two is closer to 1 than the ratio of the latter pair.

Accuracy for Human Development Index and Democracy Index were measured in mean squared error since they are bounded quantities. Since HDI is measured on a 0 to 1 scale, a 0.2 estimate for 0.3 would be considered less accurate than a 0.7 estimate for 0.8.

For all three test dimensions, 10 folds were generated that contained 19 or 20 held-out test cases. Accuracy was averaged across every predicted case in every fold.

We also generated a baseline linear regression model across all cases. The linear regression estimator is run using the implementation in Python's sklearn module, using default parameters. This requires vectorizing structured knowledge from the country cases by generating a set of features from the structured facts. This was accomplished by manually creating a mapping, where each quantity type is considered a feature, and each unique qualitative attribute (e.g. currency, international association membership) is represented by a one-hot vector. Missing quantities are imputed using Python's impute function in the scipy module. This results in 883 features across the 197 country cases.

4.5 Results

Table 2 shows the results from KNACK v2 against those generated by pure linear regression. The first run of our experiment recorded accuracy only for those cases suited especially well for analogy; they mapped to a generalization and used regression to generate an estimate. The second run of our

[4] Wikidata editors often have conflicting views of correct representations. The label for *Czech Republic* (Q213) has alternated between *Czechia* and *Czech Republic* multiple times in 2023.

experiment included accuracy for cases that mapped outside of generalizations—either to an outlier or to nothing—that fell back to a baseline of sampling within a generalization or using the marginal average. This was necessary for anywhere from 0 to 4 (with an average of 0.9 cases per fold) of the 20 test cases for a cross validation fold.

Pure KNACK v2 (with thrown away estimations) performed better than Pure Regression for 2 of the 3 testing dimensions, but not significantly ($P > 0.05$). P-values are shown in the right-most column of Table 2.

	Pure KNACKv2 Accuracy	KNACKv2 + sampling Accuracy	Pure Regression Accuracy	p-val
HDI	**0.004003988**	0.00459702	0.00708158	0.19
DI	2.367816143	2.63343881	**2.08532303**	0.64
GDP*	**15.47324807**	15.2617797	60.7963312	0.18

Table 2. KNACK results compared with KNACK and sampling for a complete set of estimations.
**GDP accuracy is normalized (|truth – estimate| / truth)*

4.6 Explainability

One of the advantages of our methods as opposed to pure quantitative regression is the explainability of our models. The primary mechanism that provides this capability is the summative statistics generated by SAGE. Recall that each generalization will yield a unique linear regression during our estimation procedure, so overarching information about the generalization can help explain unique trend lines. For example, when our system predicts the HDI of Belarus, we retrieve a generalization made up of Tajikstan, Kyrgyzstan, Armenia, Pakistan, Uzbekistan, and Kazakhstan. SAGE tells us these are all located on the Asian continent, have low (as labeled by CARVE) democracy indices, land area, and GDPs. Five of the six have low populations. Four of the six are members of the Central Asian Cooperation Organization. Being able to identify these trends and patterns is insight that other tools for quantitative estimation lack.

5 Discussion & Future Work

The results show that KNACK v2 is competitive with pure linear regression. It's interesting to note that falling back to sampling an anchor country or the marginal average made the results slightly less accurate for HDI and DI, but made the prediction for GDP *more* accurate. This could be explained by cases that do not get mapped to generalizations tending to have GDPs close to the marginal average of all countries.

The results show that under the right circumstances, KNACK v2 might be a more accurate model than pure linear regression. And unlike traditional linear regression, the cases that were used to form the estimate can be traced back to their source, increasing the explainability of, and potentially trust in, its results.

We see four directions for future work. First, we need to test KNACK v2 over more datasets. For example, Wikidata provides copious information about movies and their releases, with qualitative and quantitative information that appears promising for analogical estimation. Second, we plan to experiment with ways that systems using KNACK v2 can tune it to produce more relevant results. For example, agricultural models of a country might focus on different aspects than models of its overall economy or educational system. This could be handled with different case construction strategies and accumulating models in separate generalization pools. Third, we plan to investigate the effects of incrementality on estimation, e.g. how rapidly do estimates improve? Fourth, we plan to use KNACK v2 in a number of tasks using the Companion cognitive architecture (Forbus & Hinrichs 2017), such as back of the envelope reasoning (Paritosh & Forbus, 2007; Bundy et al. 2013) but also in metacognitive reasoning within the architecture itself, e.g. estimating effort and utility of tasks.

Acknowledgments

This research was sponsored by the US Air Force Office of Scientific Research under award number FA95550-20-1-0091.

References

Blanchette, Isabelle & Dunbar, Kevin. (2001). Analogy use in Naturalistic settings: The influence of audience, emotion and goals. Memory & cognition. 29. 730-5. 10.3758/BF03200475.

Bundy, A., Sasnauskas, G., Chan, M. (2013). Solving Guesstimation Problems using the Semantic Web: Four Lessons from an Application. *Semantic Web Journal*, doi:10.3233/SW-130127.

Fechner, Gustav Theodor (1966) [First published 1860]. Howes, D H; Boring, E G (eds.). *Elements of psychophysics* [*Elemente der Psychophysik*]. Vol. 1. Translated by Adler, H E. United States of America: Holt, Rinehart and Winston.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24, 85-168.

Forbus, K. D. (2019). *Qualitative Representations: How People Reason and Learn about the Continuous World.* MIT Press.

Forbus, K. D., & Demel, W. (2022). Integrating QR Quantity Representations with the Semantic Web: A Progress Report. *Proceedings of QR 2022*.

Forbus, K. D., Ferguson, R. W., Lovett, A., & Gentner, D. (2017). Extending SME to Handle Large-Scale Cognitive Modeling. *Cognitive Science*, 41(5), 1152–1201.

Forbus, K. D., Gentner, D., & Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19(2), 141–205.

Forbus, K., & Hinrichs, T. (2017) Analogy and Qualitative Representations in the Companion Cognitive Architecture. *AI Magazine*.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2), 155–170.

Gentner, Dedre & Loewenstein, Jeffrey & Thompson, Leigh. (2004). Learning and Transfer: A General Role for Analogical Encoding. J Educ Psychol. 95. 10.1037/0022-0663.95.2.393.

Kandaswamy, S. and Forbus, K. (2012). Modeling Learning of Relational Abstractions via Structural Alignment. *Proceedings of the 34th Annual Conference of the Cognitive Science Society (CogSci).* Sapporo, Japan

Markman, A., & Gentner, D. (1993). Structural alignment during similarity comparisons. *Cognitive Psychology*, 25, 431–467

Novick, L. R. (1988). Analogical transfer, problem similarity, and expertise. Journal of Experimental Psychology: Learning, Memory, and Cognition, 14(3), 510–520.

Paritosh, P. K. (2004). Symbolizing Quantity. *Proceedings of 2004 Annual Meeting of the Cognitive Science Society*

Paritosh, P.K. and Forbus, K.D., (2005). Analysis of Strategic Knowledge in Back of the Envelope Reasoning, In *Proceedings of the 20th National Conference on Artificial Intelligence.*

Paritosh, P. K., & Klenk, M. E. (2006). Cognitive Processes in Quantitative Estimation: Analogical Anchors and Causal Adjustment. *The Proceedings of the 28th Annual Conference of the Cognitive Science Society, Vancouver*.

Tversky, A., and Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases, *Science*, 185, pp 1124-1131.

Vrandecic, D. & Krotzsch, M. (2014). Wikidata: A Free *Collaborative Knowledgebase. Communications of the ACM*, 57(10):78-85, October, doi:10.1145/2629489

Learning Domain Knowledge and Systems Thinking using Qualitative Representations in Upper Secondary and Higher Education

Marco Kragten[1], **Tessa Hoogma**[1], **and Bert Bredeweg**[1,2]
[1]Faculty of Education, Amsterdam University of Applied Sciences, Amsterdam, The Netherlands
[2]Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
{m.kragten, t.e.hoogma b.bredeweg}@hva.nl

Abstract

This paper presents three lesson activities for upper secondary and higher education that focus on learning by constructing an interactive qualitative representation. By constructing the representation learners learn domain knowledge as well as general system thinking skills. The learning goals and the pedagogical approach are described.

1 Introduction

We investigate the pedagogical approach of learning by constructing interactive qualitative representations [Bredeweg *et al.*, 2023a]. By constructing qualitative representations, learners can develop a comprehensive understanding of subject-specific systems and improve their generic system thinking skills [Bredeweg *et al.*, 2023b; Kragten & Bredeweg, 2023].

In this paper, we describe the pedagogical approach of three lesson activities for upper secondary and higher education. The lesson activities are developed within the project Denker (https://denker.nu). The topics of the lesson activities are *photoelectric effect* (physics), *thermoregulation* (biology) and *global warming* (geography). Learners create qualitative representations using DynaLearn (https://www.dynalearn.eu). This learning space supports multiple levels at which qualitative representations can be constructed [Bredeweg *et al.*, 2013]. Each successive level adds new features to describe increasingly complex system behavior. The three lesson activities presented in this paper are at level 5 of the Dynalearn software. The lesson activities were designed for learners that are already familiar with features of level 2 [grade 7-8, see Spitz *et al.*, 2021], 3 [grade 8-9, see Kragten *et al.*, 2021] and 4 [grade 10-12, see Kragten *et al.*, 2022].

Below, we first describe the vocabulary of qualitative representation at level 5 (which includes the features of level 2, 3 and 4). This paper is dedicated to providing a comprehensive and detailed description of lesson activities that involve constructing qualitative representations at level 5. For each lesson activity we describe the learning goals of the subject-specific system and explain our pedagogical approach by showing how the representation is constructed step-by-step.

2 Qualitative Representations

2.1 Vocabulary

Entities can be either physical objects or abstract concepts, while quantities represent changeable features related to those entities in a specific system. Quantities have two characteristics: a current value and a direction of change. The latter is denoted as δ. In Figure 1 there are two entities, namely E1 and E2. E1 has a single quantity, Q1, while E2 has three quantities, namely Q2, Q3, and Q4. Possible values of a quantity are described using the notion of a quantity space, which represents the characteristic states of a quantity using a range of alternating point and interval values. Q1 has a quantity space that includes the values {0, +, Point, ++}, Q2 has a quantity space with the values {0, Interval}, while Q3 has no quantity space.

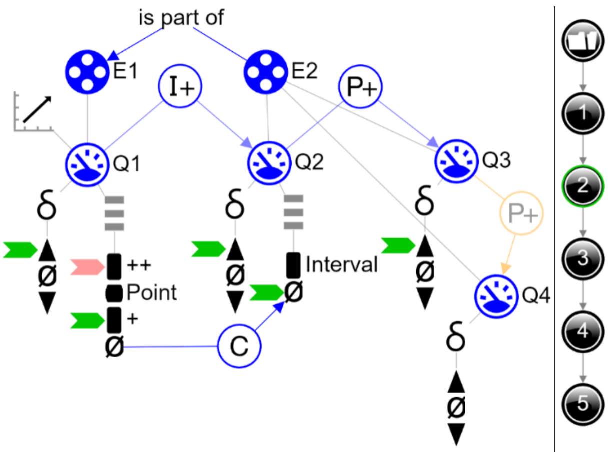


Figure 1. Qualitative representation – The simulation result of state 2 is shown.

Co-occurrence of values can be specified by adding a correspondence (C). There is a directed correspondence between values of the quantity spaces of Q1 and Q2 (if Q1 = 0 then Q2 = 0). Quantities can have causal relationships with other quantities. There are two types of causal relationships: influence (I) which is the primary cause of change, due to a process being active, and proportionality (P) which propagates change. Both relationships can be positive (I+, P+) or negative (I–, P–). In Figure 1, there is a causal relationship of the type positive influence (I+) between Q1 and Q2 (if Q1 = 0 then δQ2 = 0 and if Q1 = +, Point or ++ then δQ2 > 0). There is a positive proportional relationship (P+) between Q2 and Q3 (changes in Q2 causes changes in Q3). The notion of an exogenous influence can be used to specify a continues change for a quantity. Q1 is influenced by an exogenous influence of the type increasing. Inequalities (<, ≤, =, ≥, >) can be added to the representation to specify ordinal relations between quantities or values of a quantity space. Calculi allow qualitative calculations for operations such as multiplication or subtraction of values of quantities, resulting in the generation of a new value.

At level 5, conditional expressions can be added to the representation. Conditional expressions specify behaviors that only occur under specific conditions. Color coding is used to distinguish between the conditions and consequences of a model. In Figure 1, the positive proportional relationship (P+) between Q3 and Q4 is conditional indicated by a yellow color. The relationship is only valid if Q1 = ++ indicated by an arrow with a red color.

The qualitative representation of a system can be analyzed through a simulation that reveals the system's behavior and the direction of change of its quantities based on specified initial settings. To depict this behavior, a state graph (Figure 1, RHS) is employed, illustrating the possible states of the system. By studying the state graph, learners can gain insights into how a system evolves over time and how different factors influence its behavior. Figure 1 shows the simulation result of state 2. The representation is simulated with initial values: Q1 = 0 and an increasing exogenous influence acts on Q1. The state graph consists of five consecutive states. In state 2 Q1 = + and is increasing (δQ1 > 0), Q2 = 0 and is increasing (δQ2 > 0), and Q3 is increasing (δQ3 > 0). The change of Q4 is not determined because the condition for the positive proportional relationship to be valid (Q1 = ++) is not (yet) met.

2.2 Support

In Dynalearn, students construct representations with support based on a norm representation. This norm-based support detects differences between the student's representation and a predefined norm representation [Bredeweg *et al.*, 2023a]. An incorrect ingredient will be highlighted in red in the representation and a red question mark will appear on the right side of the canvas. A progress bar informs the students about the number of ingredients still to be added. The scenario advisor is a function that inspects the status of the model before starting a simulation and flags missing initial and/or inconsistent settings. The built-in video support functions informs students how to add ingredients to the representation (the clips are domain independent). Learners are guided using a workbook to support them with constructing the qualitative representations.

3 Photoelectric effect

The topic of this lesson activity is the photoelectric effect. It fits well into the physics curriculum of upper secondary education and higher education. Understanding the photoelectric effect in physics education is important as it provides fundamental insights into the behavior of light and electrons, serves as a cornerstone of quantum mechanics, explains experimental observations, and highlights the historical significance of scientific discoveries. The lesson was developed together with a physics teacher educator.

3.1 Subject Matter Learning Goals

The photoelectric effect is a phenomenon in physics that describes the emission of electrons from a material when it is exposed to light or other forms of electromagnetic radiation. When light interacts with a material, it transfers its energy to the electrons within the material. A key principle of the photoelectric effect is that light energy is quantized into discrete packets called photons. As the frequency of a photon increases, so does its energy. If the energy of the photons is sufficient, it can cause the electrons to be emitted from the material.

In the photoelectric effect, a crucial concept is the threshold frequency, which represents the minimum energy needed to overcome the binding forces that hold electrons within a material. This threshold frequency is unique to each specific material. When the frequency of the incident light (v) is lower than or equal to the threshold frequency (v_0) of the material, no electrons are emitted. If the frequency of the incident light (v) surpasses the threshold frequency (v_0), electrons are emitted.

The energy of a photon (E_{photon}) can be calculated by $h{\cdot}v$ where h is Planck's constant and ν is the frequency of the light. The kinetic energy (KE_{electron}) of the emitted electrons depends on the frequency of the incident light. The kinetic energy of the emitted electrons can be calculated as the energy of the photon minus the energy required to free the electron (also known as the work function Φ), so $\text{KE}_{\text{electron}} = h{\cdot}v - \Phi$. The relationship between the kinetic energy of an electron ($\text{KE}_{\text{electron}}$) and light frequency ($v$) is shown in Figure 2 (LHS).

Amplitude (A) determines the brightness or intensity of light. If the light amplitude is kept constant, the number of photons being absorbed by the material remain constant. Consequently, the rate at which electrons are emitted from the material, i.e., the electric current, remains constant as well (Figure 2, RHS).

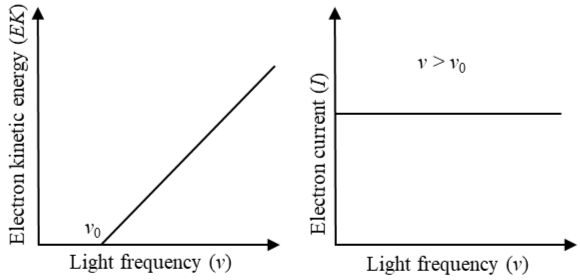


Figure 2. Photoelectric effect – Relationship between light frequency, (*i*) electron kinetic energy (LHS) and (*ii*) electron current (RHS).

Increasing the amplitude of the incident light has no effect on the energy of the incoming photon. If the frequency of the light is above the threshold and the light amplitude is increased than the kinetic energy of the electrons remains constant (Figure 3, LHS). Higher amplitude light means more photons. This results in more electrons emitted over a given time period. Hence, if the light frequency is greater than the threshold, increasing the light amplitude will cause the electron current to increase proportionally (Figure 3, RHS).

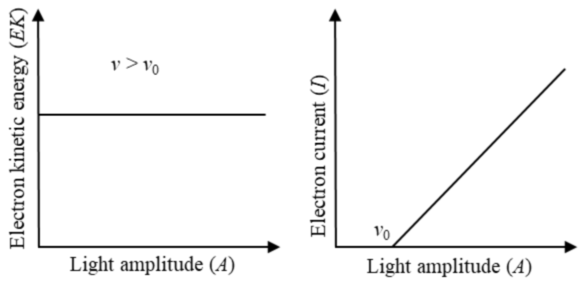


Figure 3. Photoelectric effect – Relationship between light amplitude, (*i*) electron kinetic energy (LHS) and (*ii*) electron current (RHS).

3.2 Photoelectric effect – The Representation

The final representation for this lesson activity is shown in Figure 4. The entities are *Light*, *Material* and *Electrons*. The entity *Light* has quantities *Frequency (v)* and *Amplitude (A)*, the entity *Material* has the quantity *Threshold frequency (v0)*, and the entity *Electrons* has quantities *Kinetic energy (EK)* and *Current (I)*.

3.3 Pedagogical Approach

The first part of the lesson activity focusses the relationship between the frequency of incident light and the kinetic energy of electrons.

First, learners create the entities *Light* and *Electrons*. The workbook provides an explanation regarding the nature of light, presenting it as a collection of photons that carry energy in discrete packets known as quanta. The amount of energy carried by these photons is determined by their frequency.

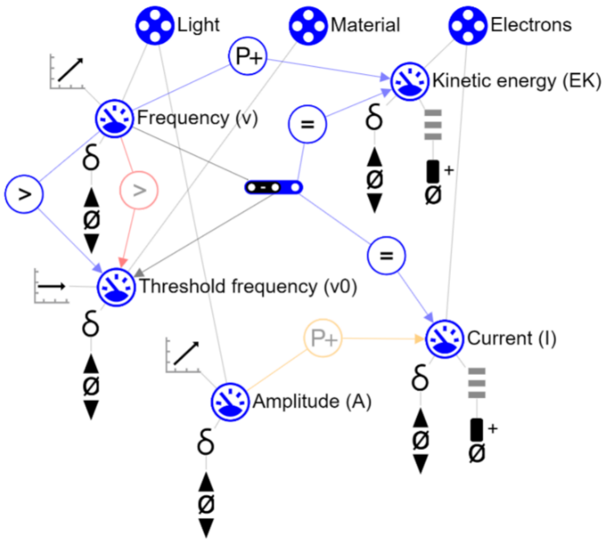


Figure 4. Photoelectric effect – Complete representation.

When light interacts with electrons, it can be absorbed by them, leading to a transfer of energy and causing the electrons to move. This movement of electrons results in the acquisition of kinetic energy by the electrons.

Learners add the quantity *Frequency (v)* to the entity *Light* and the quantity *Kinetic energy (KE)* to the entity *Electrons*. There is a positive proportional relationship (P+) between these quantities. The quantity *Kinetic energy (KE)* has a quantity space with the values {0, +}.

Learners are then instructed to set the following initial settings: an increasing exogenous influence acting on *Frequency (v)* and *Kinetic energy (KE)* is zero (0). The representation and simulation result of state 2 is shown in Figure 5.

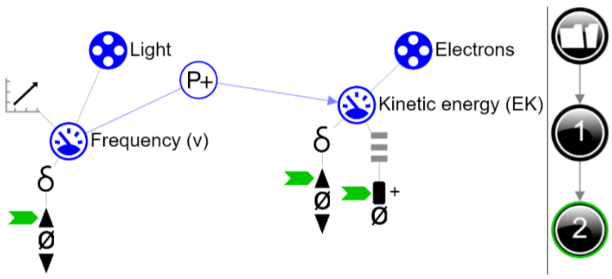


Figure 5. Photoelectric effect – Frequency of light and its effect on kinetic energy of electrons. The simulation result of state 2 is shown.

In state 2, *Frequency (v)* is increases and *Kinetic energy (KE)* is positive (+) and increasing. Learners are required to interpret the results by answering a cloze question: "In the first state the electrons *have/don't have* kinetic energy because its value is *0/+*. The frequency of light *decreases/is constant/increases* and as a result the kinetic energy of the electrons *decreases/is constant/increases*. In state 2, electrons *have/don't have* kinetic energy because its value is *0/+…*".

Next, learners are introduced to the concept that electrons within a material possess a threshold frequency. This threshold frequency indicates that incident light must surpass a certain minimum value in order to cause the emission of electrons from the material.

Learners create the entity *Material* and add the quantity *Threshold frequency (v0)*. A calculus is created that computes *Kinetic energy (KE) = Frequency (v) –Threshold frequency (v0)*.

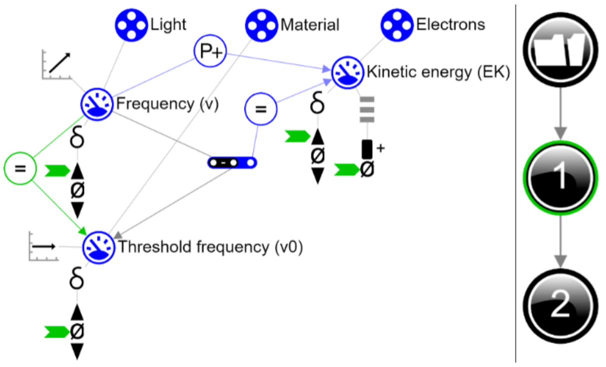


Figure 6. Photoelectric effect – Calculus of kinetic energy. The simulation result of state 1 is shown.

Figure 6 shows the representation thus far and state 1 of the simulation result with initial settings: an increasing exogenous influence acting on *Frequency (v),* a steady exogenous influence acting on *Threshold frequency (v0),* and an equality (=) between *Frequency (v)* and *Threshold frequency (v0)*. Note that, the value of *Kinetic energy (KE)* does not need to be specified as an initial setting because it is calculated. In state 1, *Kinetic energy (KE)* is zero because *Frequency (v0)* is equal to *Threshold frequency (v)* and increasing. *Frequency (v)* keeps increasing due to the exogenous influence. In state 2 (not shown), *Frequency (v)* > *Threshold frequency (v)* and thereby *Kinetic energy (KE)* is positive (+) and increasing.

Learners again answer a cloze questions that requires them to interpret the behavior of the system. Furthermore, learners are presented with the formula $KE_{electron} = h \cdot v - \Phi$ and Figure 2 (LHS). The qualitative representation constructed thus far encourages learners to gain insight into how this formula capture the relationships among these quantities.

In the next part of the lesson, the focus is on understanding how the amplitude of light affects the current of electrons. Learners learn that the amplitude of light only influences the current when the frequency of the light is higher than the threshold frequency. Moreover, it is explained that the amplitude of light has no impact on the kinetic energy of electrons.

First, learners add the quantity *Amplitude (A)* to the entity *Light* and the quantity *Current (I)* tot the entity *Electrons*. The quantity *Current (I)* has a quantity space with values {0, +}. Learners expand the existing calculus to include the notion that the value of *Current (I)* = value of *Frequency (v)* – value of *Threshold frequency (v0)*. So there is only a current when the frequency of the light exceeds the threshold frequency. The effect of *Amplitude (A)* on *Current (I)* is conditional. The amplitude of the light only has an effect on the current if the frequency of the incident light is greater than the threshold frequency. Learners add the conditional expression: if *Frequency (v) > Threshold frequency (v0)* (shown as an inequality with a red color) then there is a positive proportional relationship between *Amplitude (A)* and *Current (I)* (shown as P+ with a yellow color). Note that there is no causal relationship between *Amplitude (A)* and *Kinetic energy (KE)* and also no relationship between *Frequency (v)* and *Current (I)*.

Learners then investigate the behavior of the system under different initial settings. Figure 7 shows the representation and the simulation result with initial settings: a steady exogenous influence acting on *Frequency (v),* a steady exogenous influence acting on *Threshold frequency (v0),* an increasing exogenous influence acting on Amplitude (A), and an inequality relating the frequency to the threshold *Frequency (v) > Threshold frequency (v0)*.

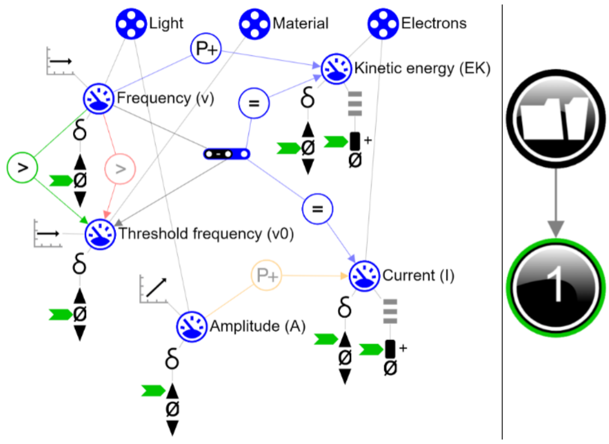


Figure 7. Photoelectric effect – Amplitude of light has an effect on the current of the electrons. State 1 of the simulation result is shown.

The simulation generates one state. The frequency of the incident light is greater than the threshold frequency so electrons are emitted and have kinetic energy (*Kinetic energy (KE)* = +). The frequency is steady so the kinetic energy of the electrons is constant (δ*Kinetic energy (KE)* = 0). There is a current (*Current (I)* = +) because the frequency of the incident light is greater than the threshold frequency. The amplitude of the light now has an effect on current. The amplitude is increasing so current is also increasing (Figure 3, RHS).

Figure 8 shows the representation of the simulation result with initial settings: a steady exogenous influence acting on *Frequency (v),* a steady exogenous influence acting on

Threshold frequency (v0), an increasing exogenous influence acting on *Amplitude (A), and the quality Frequency (v) = Threshold frequency (v0)*. Note how increasing the amplitude has no effect on the current.

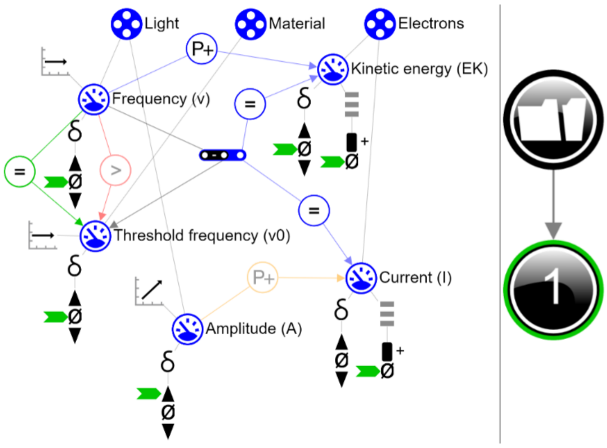


Figure 8. Photoelectric effect – Frequency of the light is below the threshold frequency therefor the amplitude of light has no effect on the current of the electrons. State 1 of the simulation result is shown.

4 Thermoregulation

This lesson activity revolves around the concept of thermoregulation. It is typically taught in upper secondary and higher education levels. Learning about thermoregulation in biology is important as it provides a foundation for understanding homeostasis, promoting human health, and exploring the diversity of life's adaptations to temperature variations.

4.1 Subject Matter Learning Goals

This lesson activity focuses on how the body responds to changes in temperature by activating heat loss or heat preservation mechanisms. Figure 9 shows a typical representation that is used in biology textbooks. When the body temperature exceeds a set point in the hypothalamus, heat loss mechanisms are activated. These include increased sweating, dilating blood vessels, and seeking shade as a behavioral response. On the other hand, if the body temperature drops below the set point, the body initiates heat preservation mechanisms. These involve constriction of skin blood vessels, stimulation of skeletal muscles to shiver, and seeking shelter as a behavioral response. Overall, this system operates as a negative feedback loop, with various mechanisms being activated above or below the set point. For instance, sweating becomes active above the set point, while shivering becomes active below the set point. However, the constriction and dilatation of blood vessels in the skin are mechanisms utilized by the body both above and below the set point.

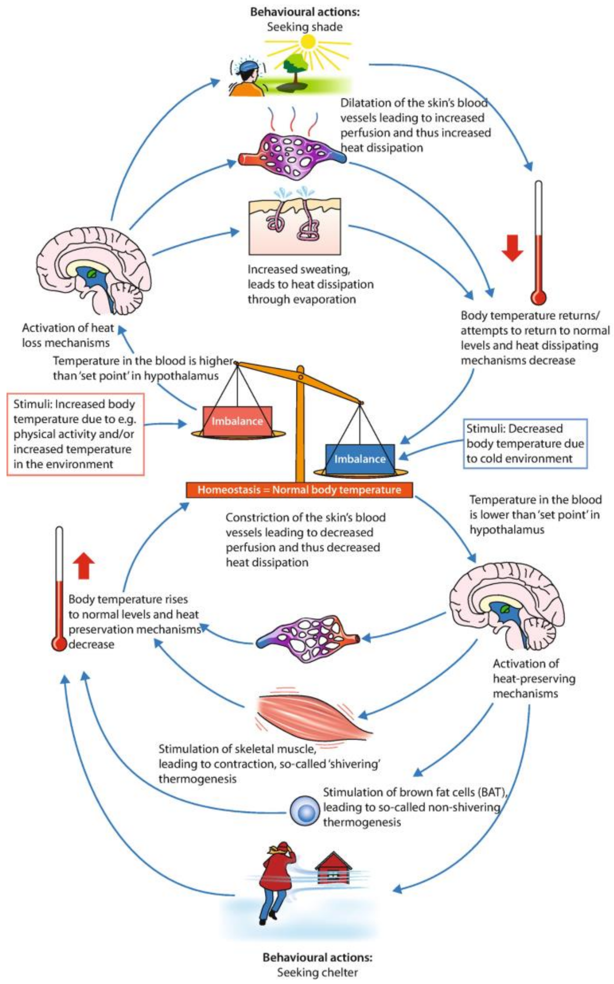


Figure 9. Thermoregulation – Image from biology textbook [Grodzinsky & Sund Levander, 2020].

4.2 Thermoregulation – The Representation

The final representation for this lesson activity is shown in Figure 10. There are five entities: *Blood*, *Hypothalamus*, *Skeletal muscles*, *Skin* and *Blood vessels*. The entity *Blood* has the quantity *Temperature*, the entity *Hypothalamus* has the quantities *Norm* and *Difference*, the entity *Skeletal muscles* has the quantity *Shivering*, the entity *Skin* has the quantity *Sweating* and the entity *Blood vessels* has the quantity *Blood flow*.

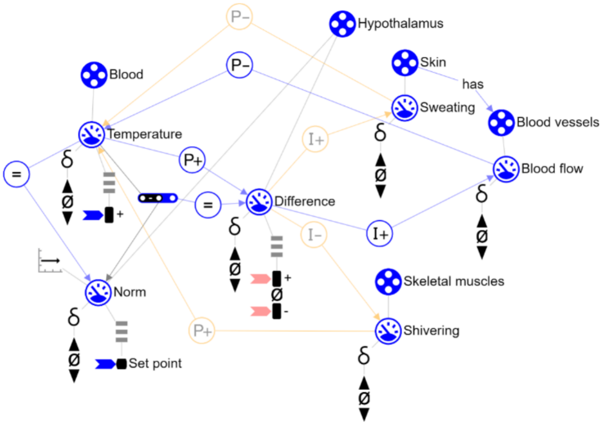


Figure 10. Thermoregulation – Complete representation.

4.3 Pedagogical approach

The first part of the lesson activity focusses on the relationship between the temperature of the blood and the hypothalamus that compares the difference with a set point.

First, learners create the entity *Blood* and add the quantity *Temperature*. They also create the entity *Hypothalamus* and add the quantity *Difference* with a quantity space with values {−, 0, +}. *Difference* is positive proportional to *Temperature.*

Figure 11 shows the representation and state 2 of the simulation result with initial settings: an increasing exogenous influence acting on *Temperature* and the starting value *Difference* = 0. In state 2, *Difference* is positive (+) and increasing (δ*Difference > 0*).

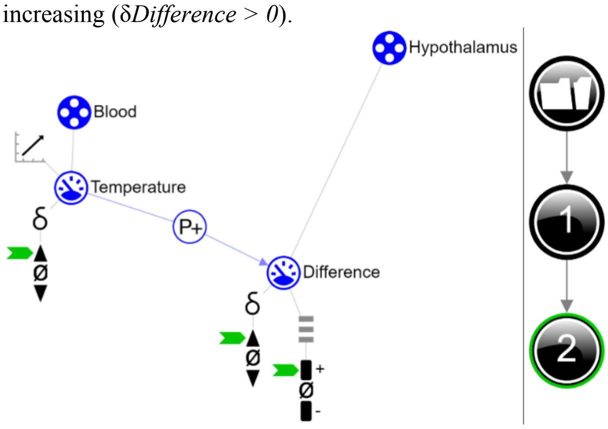


Figure 11. Thermoregulation – First simulation.

Next, the lesson activity focusses on a more precise understanding of how the difference between the temperature of the blood and the set point of the hypothalamus can be calculated.

Learners add the quantity *Norm* to the entity *Hypothalamus* and add a quantity space with point value {Set point} to it. They also add quantity space with an interval {+} to the quantity *Temperature*. Learners then create a calculus that computes *Difference* = value of *Temperature* – value of *Norm*. A constant exogenous influence indicates that *Norm* does not change.

Learners add an inequality as an initial setting between *Temperature* and *Norm* for the calculus to have an effect (without this information there is no way of knowing the outcome of the calculus).

Figure 12 shows the representation this far and the simulation result of state 1. The inequality shows that *Temperature* < *Norm* so *Difference* is negative (−). Learners are required to explain what happens by answering a cloze question: "In state 1, the temperature of the blood is *lower than/equal to/higher than* the set point of the hypothalamus so the difference is *negative/zero/positive*.".

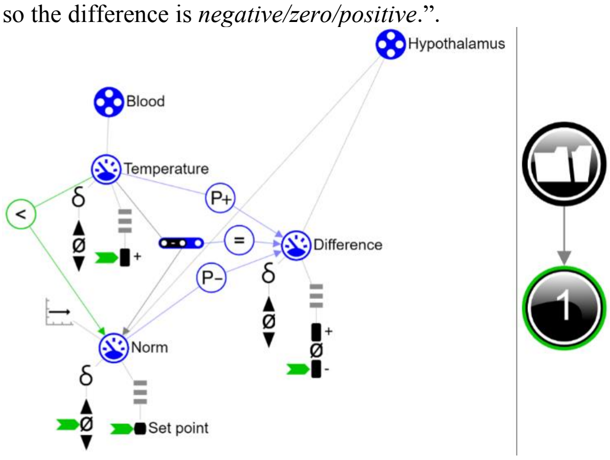


Figure 12. Thermoregulation – *Difference* is calculated by *Temperature* minus *Norm*. The inequality shows that *Temperature* is smaller than *Norm* so *Difference* is negative (−) in this state (#1).

The next part of the lesson activity focuses on the regulation of blood flow in the skin in response to deviation from the norm.

Learners are informed that the hypothalamus plays a vital role in regulating the blood flow in the skin, which directly influences the amount of heat loss by the body. Learners create the entity *Skin* and the entity *Blood vessels* and add a configuration (skin *has* blood vessel). The entity *Blood vessels* has the quantity *Blood flow*. It is explained that when the hypothalamus detects a *negative/positive* difference than the amount of impulses to the muscles that determine the diameter of the blood vessels *decreases/increases* and thereby the blood flow *decreases/increases*. Learners add a causal relationship of the type positive influence (I+) between *Difference* and *Blood flow* and a negative proportional relationship (P-) between *Blood flow* and *Temperature*.

Figure 13 shows the representation and state 1 of the simulation result with initial setting: *Temperature* < *Norm*. The state graph shows two consecutive states. In the first state *Difference* is negative (−) and increasing. The latter is due to decreasing *Blood flow* which has a positive effect on *Temperature* of *Blood*. In state 2 (not shown), *Difference* is zero (0) and *Blood flow* and *Temperature* are constant. So there is a negative feedback loop working to maintain homeostatic balance around the set point.

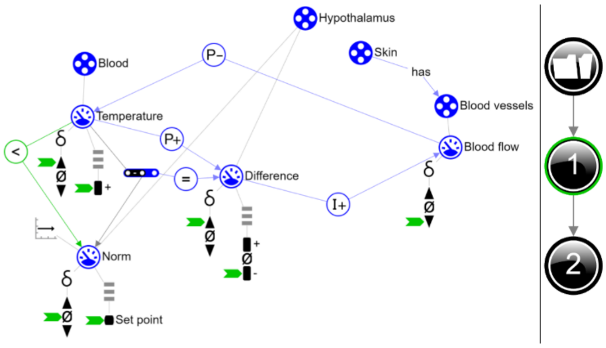


Figure 13. Thermoregulation – *Blood flow* has an effect on *Temperature* of the *Blood*.

The next part of the lesson activity emphasizes the additional measures taken by the hypothalamus in response to temperature deviations from the norm, extending beyond the regulation of blood flow.

Learners are explained that regulating blood flow is in most cases not enough to maintain a stable temperature and that additional measures are needed. Some of these measures are conditional, meaning that they are only implemented at specific temperature values. Learners first implement the shivering of the skeletal muscles as a conditional response to a negative difference between the temperature of the blood and the norm. Learners create the entity *Skeletal Muscles* and add the quantity *Shivering*. They are required to create a conditional response: if *Difference* = − then there is a causal relationship of the type negative influence (I-) between *Difference* and *Shivering* and a positive proportional relationship (P+) between *Shivering* and *Temperature*. Figure 14 shows the representation and state 1 of the simulation result with intital condition: *Temperature* < *Norm*. In state 1, *Difference* is negative (−) and increasing. The number of impulses to the skeletal muscles will increase and thereby *Shivering* will increase which has a positive effect on *Temperature*.

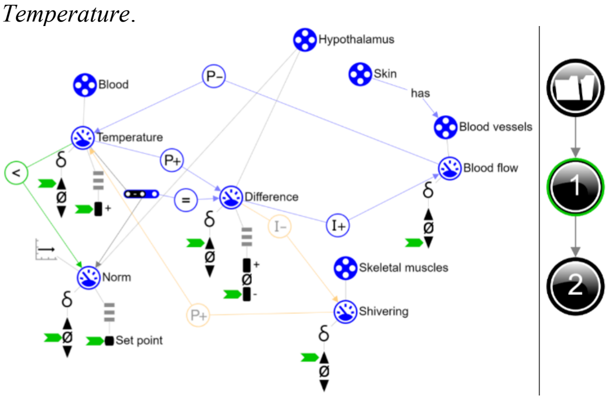


Figure 14. Thermoregulation – Shivering is a conditional response that is only applicable if the temperature of the blood is below the norm.

Learners finish the representation (Figure 15) by adding the quantity *Sweating* to the entity Skin and by adding the condition: if *Difference* = '+' then there is a causal relationship of type positive influence (I+) between *Difference* and *Sweat* and a negative proportional relationship (P-) between *Sweat* and *Temperature*.

Learners simulate the representation with initial setting: *Temperature* > *Norm*. The state graph shows two consecutive states. In the first state *Difference* is positive (+) and decreasing. The number of impulses to the skin will increase and thereby *Sweating* will increase which will increase heat loss by the skin and thereby has a negative effect on *Temperature*. Note that the other condition (*Difference* = −) is not met and that *Shivering* is not determined. Learners answer several cloze questions, e.g.: 'If the temperature of the blood is above the norm than sweating will *decrease/be steady/increase*...'.

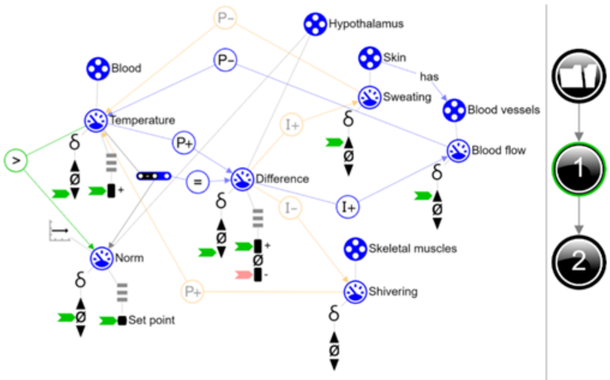


Figure 15. Thermoregulation – Sweating is a conditional response that only applies if the temperature of the blood is above the norm.

5 Global warming

The lesson centers around global warming and is well-suited for a geography class.

5.1 Subject Matter Learning Goals

This lesson has two main learning goals. First, research findings indicate that learners have a limited understanding of carbon dioxide accumulation [Qudrat-Ullah & Kayal, 2018]. The levels of carbon dioxide in the atmosphere are influenced by two primary processes: emissions by human activity (and natural processes like volcanic activity) and uptake by the biosphere, which includes processes like photosynthesis and oceanic absorption. Emissions contribute to the increase of carbon dioxide levels, while the biosphere acts as a natural regulator by taking in carbon dioxide from the atmosphere. Learners often encounter challenges in predicting carbon dioxide levels in the atmosphere based on emissions and uptake. A common misconception among learners is that a decline in carbon dioxide emissions will automatically lead to a decrease in atmospheric carbon dioxide levels. However, it is important to note that this is only true when the rate of emission is lower than the rate of uptake. The balance between emissions and uptake determines the overall impact on atmospheric carbon dioxide levels (Figure 16). This lesson aims to improve learners' comprehension in this area.

The second learning goal aims to develop learners' understanding that global warming affects different regions of the world unequally. In this lesson, we concentrate on the varying effects of temperature rise on economic production. Research findings demonstrate that as temperatures increase, there is a positive effect on economic production up to a certain threshold, beyond which the effect turns negative [Burke *et al.*, 2015].

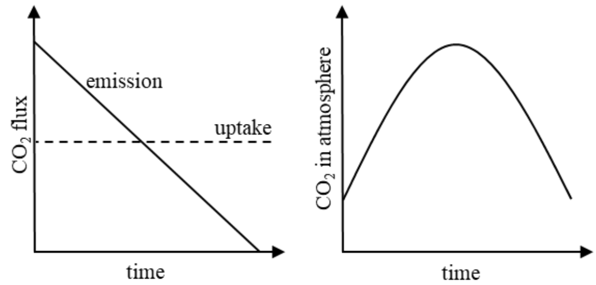


Figure 16. Carbon dioxide emissions and (*i*) uptake (LHS), and (*ii*) the impact on atmospheric carbon dioxide levels (RHS).

5.2 Global warming – The Representation

The final representation for this lesson activity is shown in Figure 17. The entity *Human activity* has quantities *Emission* and *Economic production*, the entity *Atmosphere* has quantities *CO2* and *Temperature*, the entity *Biosphere* has quantity *Uptake.*

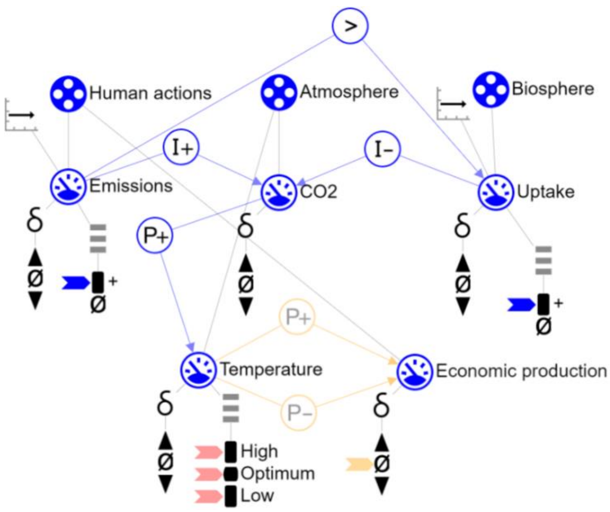


Figure 17. Global warming – Complete representation.

5.3 Pedagogical approach

The first part of the lesson activity focusses on understanding the effect of emissions and uptake on carbon dioxide levels in the atmosphere.

Learners start by creating the entity *Human actions* and add the quantity *Emissions*. Next, they create the entity *Atmosphere* and add the quantities *CO2* and *Temperature*. Learners add a causal relationship of the type positive influence (I+) between *Emissions* and *CO2* and a relationship of the type positive proportional (P+) between *CO2* and *Temperature*. *Emissions* has a quantity space with values {0, +}.

Learners simulate the representation with successively decreasing, steady and increasing exogenous influences on *Emissions*. Figure 18 shows the representation and the simulation result with the exogenous influence being steady. Given the constant emissions, learners observe that the concentration of carbon dioxide (CO2) will consistently increase. To interpret the results, learners are required to answer cloze questions, e.g., “If emission is positive and decreasing then CO2 concentration in the atmosphere *decreases/is steady/increases.*”.

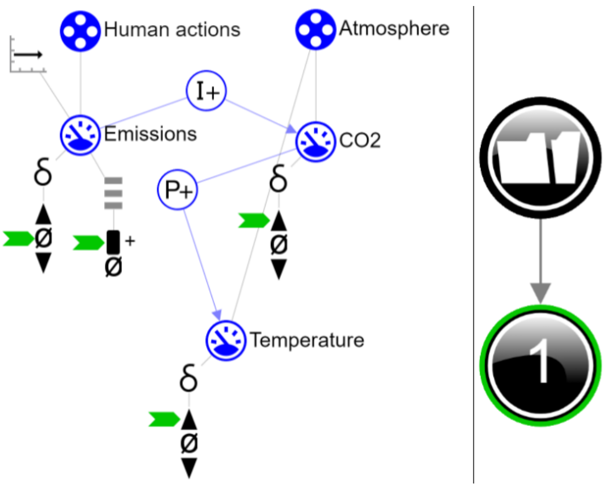


Figure 18. Global warming – The effect of emission on carbon dioxide levels and temperature.

In the next step learner create the entity *Biosphere* and add the quantity *Uptake*. The quantity has a quantity space with values {0, +} and a causal relationship of the type negative influence (I-) with *CO2*.

Figure 19 shows the simulation result with initial settings: *Emission* is decreasing, *Uptake* is steady and *Emission* > *Uptake*. The simulation result has four consecutive states.

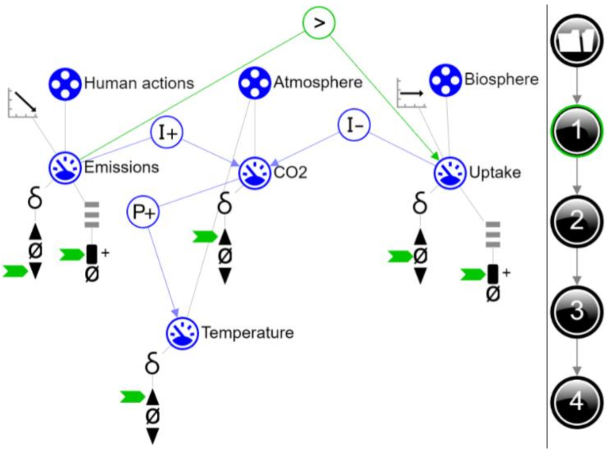


Figure 19. Global warming – The effect of emission and uptake on carbon dioxide levels.

Learners are instructed to select the four states of the simulation result and to display the value and inequality history (Figure 20). The value and inequality history displays the behavior of the system to learners in a convenient way as they do not have to click the states in the state graph one-by-one to inspect values, changes and inequalities. The value history shows that in states 1-3 *Emissions* is positive (+) and decreasing and is zero (0) in state 4 and that *Uptake* is positive (+) and steady in all states. In state 1 *Emissions* > *Uptake*, in state 2 *Emissions* = *Uptake* and in state 3 and 4 *Emissions* < *Uptake*. The value history also shows the second derivative of each quantity (if applicable). From this, learners can infer that *CO2* is decreasingly increasing in state 1. In state 2, *CO2* has reached its maximum value. In state 3, *CO2* is increasingly decreasing and in state 4 *Emission* = 0 and *CO2* is steady decreasing.

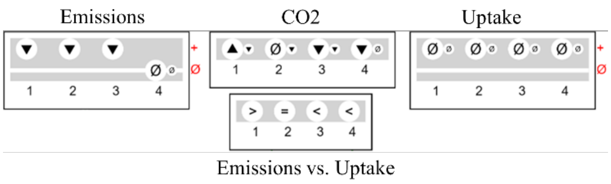


Figure 20. Global warming – Value and inequality history of the simulation result.

Learners are required to interpret the value and inequality histories and translate these into a line graph that corresponds to Figure 16. Also note that state 4 is not realistic. This provides opportunity for learners to the learn about the limitation of the representations and to think about ways to improve it.

The second part of the lesson focusses on the relationship between temperature and economic production. It is taught that the effects of global warming vary by region.

Learners add the quantity *Economic production* to the entity *Human actions* and add a quantity space with values {Low, Optimum, High}. Learners add three conditional expressions to the representation. The first expression details: if *Temperature* = Low (shown as an arrow with a red color in Figure 21) then there is a positive proportional relationship between *Temperature* and *Economic production* (shown as P+ with a yellow color). The second expression details: if *Temperature* = Optimum then *Economic production* is steady (shown as a yellow arrow on the value 0 of the derivative of *Economic production*). The third expression details: if *Temperature* = High then there is a negative proportional relationship between *Temperature* and *Economic production* (shown as P- with a yellow color).

Figure 21 shows the representation and state 3 of the simulation result with initial settings: *Emissions* is positive (+) with a steady exogenous influence acting on it, *Uptake* is positive with a steady exogenous influence acting on it, and *Emissions* > *Uptake*. *Temperature* increases in all states but in state 1 *Economic production* increases (not shown), in state 2 it is steady (not shown) and in state 3 it decreases (shown).

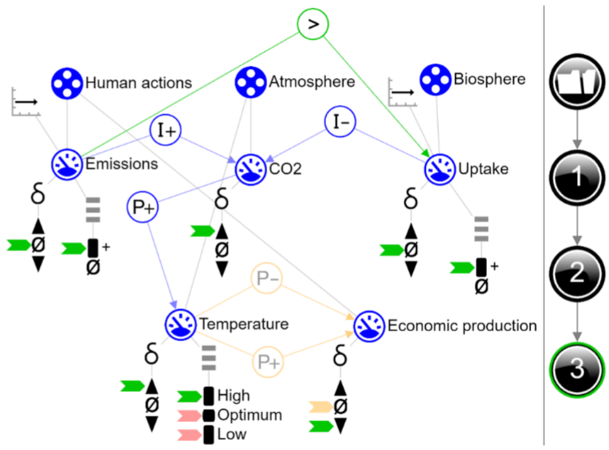


Figure 21. Global warming – The conditional effect of temperature on economic production.

7 Conclusion and Discussion

This study investigated the pedagogical approach of learning through the construction of qualitative representations. Three lesson activities were developed for upper secondary and higher education. These activities, conducted at level 5 of the DynaLearn software, aimed to enhance learners' understanding of subject-specific systems and improve their system thinking abilities. The importance and relevance of acquiring systems thinking skills are widely supported and emphasized in educational discourse [Jacobson & Wilensky, 2006; NRC, 2012]. The lesson activities focused on the *photoelectric effect* (physics), *thermoregulation* (biology), and *global warming* (geography). These topics were specifically selected because they require the use of conditional expressions, which align with the typical features of level 5 in the Dynalearn software. By focusing on these topics, we provided learners with opportunities to construct and explore qualitative representations that capture the complex and conditional nature of these systems.

With this paper we conclude a series of papers exploring the pedagogical value of interactive qualitative representations across different subjects in education. Our previous papers focused on lower secondary education (level 2 and 3; Spitz *et al.*, 2021; Kragten *et al.*, 2021) and upper secondary education (level 4; Kragten *et al.*, 2022). By incorporating qualitative representations in various subject areas, learners were able to develop a comprehensive understanding of different systems encountered in their classes. By engaging with qualitative representations, learners not only gained a deeper understanding of specific systems but also developed generic system thinking skills. This skill set is potentially transferable and can support learners in comprehending new systems by recognizing underlying principles that are often shared across domains [Goldstone & Wilensky, 2008].

The three lesson activities presented in this paper, developed as part of project Denker (https://denker.nu), exemplify the value of creating qualitative representations to enhance learning across diverse subjects. The project has successfully developed over 30 lesson activities for biology, physics, geography, and economics classes, ranging from level 2 to level 5 of the software. To facilitate easy access and immediate implementation, a collection of lesson activities ready for immediate use in the classroom can be found on the Dynalearn website (https://www.dynalearn.eu).

References

[Bredeweg et al., 2013] Bert Bredeweg, Jochem Liem, Wouter Beek, Floris Linnebank, Jorge Gracia, Esther Lozano, Michael Wißner, René Bühling, Paulo Salles, Richard A. Noble, Andreas Zitek, Petya Borisova and David Mioduser. DynaLearn - An Intelligent Learning Environment for Learning Conceptual Knowledge. *AI Magazine*, *34*(4), 46-65, 2013.

[Bredeweg et al., 2023a] Bert Bredeweg, Marco Kragten, Joanna Holt, Patricia Kruit, Tom van Eijck, Monique Pijls, Anders Bouwer, Malou Sprinkhuizen, Emile Jaspar, & Muriel de Boer. Learning with Interactive Knowledge Representations. *Applied Sciences, 13*(9), 5256, 2023.

[Bredeweg et al., 2023b] Bert Bredeweg, Marco Kragten, Joanna Holt, Dennis Vaendel, Joris Hanse, & Stef Bloemen. Stargazing Live! Inspiring with Real Data in a Mobile Planetarium and Learning Through Conceptual Modelling. In *Augmented Intelligence and Intelligent Tutoring Systems: 19th International Conference, ITS*, LNCS 13891, 257-269, 2023.

[Burke et al., 2015] Marshall Burke, Solomon Hsiang, & Edward Miguel. Global non-linear effect of temperature on economic production. *Nature*, *527*, 235-239, 2015.

[Goldstone & Wilensky, 2008)] Robert Goldstone & Uri Wilensky. Promoting transfer by grounding complex systems principles. *The Journal of the Learning Sciences, 17*(4), 465-516, 2008.

[Grodzinsky & Sund Levander, 2020] Ewa Grodzinsky & Märta Sund Levander. Thermoregulation of the human body. In *Understanding Fever and Body Temperature: A Cross-disciplinary Approach to Clinical Practice* (pp. 49-65). Palgrave Macmillan Cham, 2020.

[Jacobson & Wilensky, 2006] Michael Jacobson & Uri Wilensky. Complex systems in education: Scientific and educational importance and implications for the learning sciences. *The Journal of the Learning Sciences, 15*(1), 11-34, 2006.

[Kragten et al., 2021] Marco Kragten, Loek Spitz and Bert Bredeweg. Learning Domain Knowledge and Systems Thinking using Qualitative Representations in Secondary Education (grade 9-10). Paper presented at 34th *International Workshop on Qualitative Reasoning*, Montreal, Canada, 2021.

[Kragten et al., 2022] Marco Kragten, Emile Jaspar, & Bert Bredeweg. Learning Domain Knowledge and Systems Thinking using Qualitative Representations in Secondary Education (grade 10-12). Paper presented at 35th *International Workshop on Qualitative Reasoning*, Vienna, Austria, 2022.

[Kragten & Bredeweg, 2023] Marco Kragten & Bert Bredeweg. Learning about circular motion of celestial bodies with interactive qualitative representations. *Artificial Intelligence in Education*: 24nd International Conference, AIED, 2023.

[NRC, 2012] NRC (National Research Council). *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas.* Washington, DC: The National Academies Press, 2012.

[Spitz et al., 2021] Loek Spitz, Marco Kragten, & Bert Bredeweg. Learning Domain Knowledge and Systems Thinking using Qualitative Representations in Secondary Education (grade 8-9). Paper presented at 34th *International Workshop on Qualitative Reasoning*, Montreal, Canada, 2021.

[Qudrat-Ullah & Kayal, 2018] Hassan Qudrat-Ullah & Ayman Kayal. How to improve learners'(mis) understanding of CO2 accumulations through the use of human-facilitated interactive learning environments? *Journal of Cleaner Production*, *184*, 188-197, 2018.

Exploring Emotional Dimensions of Food Waste Perception

Konstantina Zacharaki[a;*], Jennifer Nguyen[a], Queralt Prat-i-Pubill[a] and Núria Agell[a]

[a]ESADE Business School, Ramon Llull University
ORCiD ID: Konstantina Zacharaki https://orcid.org/0000-0001-8264-8457,
Jennifer Nguyen https://orcid.org/0000-0002-7498-7536,
Queralt Prat-i-Pubill https://orcid.org/0000-0003-3505-4490, Núria Agell https://orcid.org/0000-0001-9264-2147

Abstract Approximately one third of the food produced globally is lost (14%) or wasted (17%) (FAO, 2019; UNEP, 2021). This phenomenon deserves full attention from governmental institutions and the academic community. The European Commission has just proposed to include food waste reduction targets for 2030 in its Waste Framework Directive. As two main areas for reduction are retail and household food waste, citizen panel discussions have already begun to understand their views on the topic. Given these concerns, in this paper we consider a qualitative reasoning model using hesitancy to explore people's emotions towards the environment with a specific focus on food waste. We collected data from 188 participants in an in-person taste experiment. We analyze consumers' preference towards fruit that comes from the supermarket or alternative sources such as the 'Too good to go' application in relation to their self-reported emotional reaction towards FW using linguistic terms. Data on emotion perception while throwing away food is used to calculate a consensus across the different fruit preference groups of participants. In our research, we additionally include gender and participants' prior knowledge of the application as variables during data analysis. This approach using hesitant linguistic terms was used to unveil the most pertinent emotions related to FW and was able to identify which are the emotions that are more relevant in different groups.

1 Introduction

Unsustainable consumption and production patterns are the root cause of the triple planetary crisis: climate change, biodiversity loss, and pollution. The United Nations' 2030 Agenda for Sustainable Development emphasizes 17 urgent goals (SDGs) to address these challenges. Food loss and waste have significant environmental, economic, and social consequences. Food systems alone contribute to 34% of total anthropogenic greenhouse gas emissions (GHG; CO2, CH4, N2O, fluorinated gases) (Crippa, 2021), with 50% of these emissions attributed to food loss and waste (Zhu et al, 2023). Additionally, precious resources crucial for food production, including labor, energy, land, and freshwater, are being lost or wasted. Astonishingly, it is estimated that 24% of global freshwater, 23% of cropland, and 23% of total fertilizers used worldwide are being squandered (Kummu et al., 2012). This wastage also leads to land degradation through soil erosion, desertification, deforestation, and nutrient depletion (Rockström et al, 2023). Disturbingly, simultaneously approximately 800 million people suffer from hunger, and around 30% of the global population face moderate to severe food insecurity in 2021(UN, 2022).

Previous literature on FW has focused on different aspects of the issue such as the variables that affect this phenomenon. It is pivotal to understand the underpinnings of why consumers are willing to throw away food in the household as it is the main source of FW. De Hooge et al. (2017) run a big sample experiment in five Northern European countries in which participants had to decide between typical and suboptimal products of different categories. The results show that there are many factors in play such as the context of buying the products (online or at the supermarket) but also how suboptimal the product was. They also found that demographics, personality, and individual values affected their choices. Related research has been conducted in relation to potatoes in Spain (Gracia & Gómez, 2020) and citrus fruit in Taiwan (Huang et al., 2021). In a similar vein, Ponis et al. (2017) conducted a household survey in Greece

*Corresponding Author. Email: konstantina.zacharaki@esade.edu

regarding the impact of different shopping habits, eating preferences and food management on FW. The difference here being that the focus was on the behavior of the consumer instead of the product'(s) characteristics.

The FW problem is multi-faceted as it entails social, environmental, and economic aspects. In the present manuscript we have taken a multidisciplinary approach that combines different techniques. We ran an in-person taste experiment where participants were offered to try two kinds of apple. One coming supposedly from the supermarket and the other one from the 'Too good to go' application. This application is a well-known platform across the European Union where users can buy surplus food at a cheaper price. This second type of apple could be considered the suboptimal one as it is potentially not as fresh. After participants tried the two supposedly different apples, they were asked which apple they preferred and to answer a short survey. Our contribution is that we have innovatively designed an experiment to capture how participants' pre-conceptions of FW affect their taste.

We are using the fruit preference experiment in combination with a series of questions regarding food waste and environmental tendencies as a proxy to understand FW behavior in a young population. The reason why we decided to focus our research on young people is twofold: firstly, it has been shown that younger people contribute more to food waste (de Hooge et al., 2017) and secondly, it was easier to recruit young participants as they were tested in the university.

In the present manuscript we have focused our analysis on emotional response to food waste. This was done because previous research highlights those non-cognitive variables of emotions and habits influence FW and there is very limited, mostly qualitative research on the topic (Rusell et al., 2017). People's perceptions about real problems may be imperfect and incomplete and several studies have considered the use of qualitative or approximate reasoning to model sensory perceptions (Castro-Lopez & Alonso, 2019). Additionally previous work has focused solely on negative emotions while we have 6 different emotions that have a negative or a positive connotation. We have also analyzed demographic information on gender as it has also been shown that females tend to contribute more to FW and previous knowledge of the application.

The methodology used in this paper moves in two directions: first, analyzing differences among emotions when in different groups of people using a statistical analysis, and then using qualitative reasoning models including hesitant terms to find a central opinion of different profiles and measuring the consensus in each one of these groups. The methodology is able to capture subtle differences in group responses that classical statistical methods were not able to detect.

The rest of this paper is structured as follows: Firstly, Sect. 2 introduces preliminary concepts on HFLTS presenting definitions of centroid and consensus, these basic concepts were already presented in a previous study (Montserrat-Adell et al., 2016). Section 3 introduces our experimental approach together with data analysis and results considering both approaches, the numerical statistical and the qualitative reasoning approach. Finally, Sect. 4 contains the main conclusions and lines of future research.

2 Preliminaries

A summary of the basic concepts related to hesitant linguistic term sets (HLTS) that will be referenced in the experimental part of the paper is presented in this section.

Let S denote a finite totally ordered set of linguistic terms, $S = \{s_1, \dots, s_n\}$, with $s_1 < \dots < s_n$, where the elements of S are considered as the basic terms, and n denotes the granularity of the model. Aligned with the concepts introduced by Rodriguez et al. (2011), *hesitant fuzzy linguistic term set* (HFLTS) over S is a subset of consecutive linguistic terms of S, i.e., $\{x \in S | s_i \le x \le s_j\}$, for some i, $j \in \{1, \dots, n\}$ with $i \le j$. We note $[s_i, s_j]$ to this HFLTS, or $\{s_i\} \equiv [s_i, s_i]$ if $i = j$.

Then, $\mathcal{H}_S$ is defined as the set of all possible HLTS over S excluding the empty set. In addition, we define the hesitancy of a linguistic term $[s_i, s_j]$ as $\mathcal{W}([s_i, s_j]) = j - i + 1$. In $\mathcal{H}_S \cup \{\emptyset\}$, the *intersection* $\cap$ and the *connected union* $\sqcup$ are defined as follows:

- $[s_i, s_j] \cap [s_k, s_l] = [s_{\max\{i,k\}}, s_{\min\{j,l\}}]$, if this HFLTS exists or $\emptyset$ otherwise.
- $[s_i, s_j] \sqcup [s_k, s_l] = [s_{\min\{i,k\}}, s_{\max\{j,l\}}]$.

Note that intersection and connected union are closed binary operations defined on $\mathcal{H}_S \cup \{\emptyset\}$. It is not difficult to prove that the set $\mathcal{H}_S \cup \{\emptyset\}$, jointly with the two-binary operation intersection and connected union, form a lattice (Amina and Azim, 2019).

In addition, a distance between two HLTS as defined in Porro et al. (2022). Given $H_1, H_2 \in \mathcal{H}_S$, the distance between H_1 and H_2 is defined as:

$$d(H_1, H_2) = 2 \cdot card(H_1 \sqcup H_2) - card(H_1) - card(H_2) \quad (1)$$

In addition, given a set of linguistic terms $G = \{H_1, \dots, H_k\}$, we define the centroid as:

$$H^C = \arg\ min \sum_{j=1}^{k} d(H, H_j) \quad (2)$$

with $H_j \in G$ that is to say the element in the lattice that minimizes the addition of the distances to all the elements of the given set G. When the set of linguistic terms G come from the opinions of a group of individuals this element is considered as the central opinion. The central opinion is the hesitant term that is most representative of all the opinions in the group. It is not necessarily one of the individual opinions, but it is able to capture global uncertainty in responses. Note that in some cases, the centroid is not a unique element Finally, the consensus among all G elements is computed by means of:

$$\delta(G) = 1 - \frac{\sum_{i=1}^{k} d(H^c, H_i)}{k(n-1)} \quad (3)$$

This consensus degree proposed by Montserrat-Adell et al. (2016) is used to quantify the opinion agreement among a set of individuals. The consensus complements the centroid as it shows the polarity of the opinions of the group. A small consensus implies low agreement among all the individuals in contrast to a large consensus where there is considerable agreement. This will allow us to compare the relevance or impact of two different aggregate opinions.

In this paper the individuals are the participants of the experiment, the variable of analysis will be the emotions, and the opinion are with respect to the emotions.

3 Experimental approach

3.1 Participants

A total of 181 participants were tested (Mean age = 19, SD age = 1, female = 97). An additional 5 participants were tested but discarded from the final sample due to technical error (n=5) and failure to complete the whole study (n=2). Participants were undergraduate students from the ESADE Business School and were given extra-credit scores as compensation for their participation. The present experiment was approved by the Research Ethics committee at ESADE (009/2023), and all data were treated confidentially. All participants signed a consent form before taking part in the experiment.

3.2 Materials

In the spirit of sustainability, we used a local Catalan variety of apples (Golden Empordà) for the experiment from a nearby market. The fruit was always freshly cut, no more than 10 minutes before the arrival of the participants. The apples were first peeled and then cut using an apple cutter to guarantee equal slices(see figure 1).

3.3 Paradigm/Experimental Procedure

We used an adaptation of the testing paradigm used in Sörqvist et al., (2013). This type of paradigm is typically used in the context of taste experiments. Deception is used given that the same product is used but labelled differently when the research question is on the drivers behind a phenomenon and not the products themselves (e.g. Liem et al., 2012 for soup). Participants were tested in a soundproof room at the Decision Lab located at the ESADE Sant Cugat Campus. They sat in a chair facing a table where two transparent bowls were placed containing the apple slices. Participants were asked to give their consent by signing a form and were asked whether they had any food allergies. Then, the experimenter asked if they had any previous knowledge of the 'Too good to go' app. Irrespective to their response, the experimenter gave the same brief description to all participants in order to make sure they all had a basic understanding of the source of the fruit. Participants were then asked to try the fruit. The experimenter labeled each bowl (supermarket/'too good to go') and offered the participants as many slices as they fancied to have a concrete idea of the taste of the apple. They were instructed to have some water between the two tastings. The order and side of presentation of the two kinds of apple was counterbalanced across participants. Following the tasting, they were guided to an adjoining room where a Microsoft Surface Tablet equipped with a keyboard was placed. Participants had to answer a series of questions privately using a survey on Qualtrics. This setting was chosen to reduce social biases based on which participants felt pressured to answer the desirable choices according to society. They had to choose which apple they preferred. They were given three choices: Supermarket, 'Too good to go' or both. Participants were asked to rate how they feel when throwing away food on a scale from 1 (Not at all) to 5 (Very much) allowing multiple answers per emotion, using the six basic emotions: angry, ashamed, happy, indifferent, guilty and sad and other questions on their behavior concerning FW and the environment. Lastly, we also asked them to fill in some basic demographic information on their gender (Female, Male, Non-binary/Third Gender, Prefer not to say), age and previous familiarity with the 'Too good to go' app. These variables are included in a bigger project that contains more questions on the profile of the participants.

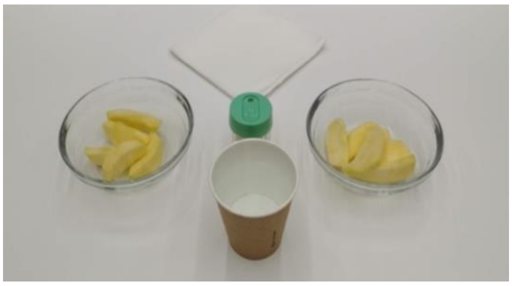


Figure 1. Experimental setting. The two alleged distinct apples were placed in two transparent bowls in front of the participants.

4 Data analysis

We analyzed the data on emotions related to food waste. Participants' evaluations were analyzed using two different approaches:

4.1 Statistical Approach

The evaluations of participants in relation to the six emotions (angry, ashamed, guilty, happy, indifferent, and sad) were treated as numerical values ranging from 1 to 5. Answers that included hesitancy, i.e., more than one value per emotion were averaged. For instance, if a participant answered that they felt ashamed 2-5, then these range was replaced by their mean which is 3.5. These values were submitted to a mixed-ANOVA as the dependent variable. The type of Emotion was introduced as a within participant factor. Three factors were introduced to the ANOVA as between: Fruit Preference (Both, Supermarket, Too good to go), Gender (Female, Male) and Previous Knowledge of 'Too good to go' (Yes, No). The ANOVA included the main effects of these factors but also their interaction.

4.2 *Qualitative Reasoning Approach*

The evaluations of participants in relation to the six emotions (angry, ashamed, guilty, happy, indifferent, and sad) were treated as linguistic labels considering the opinions from a lattice of HFLTS with granularity 5 where the basic terms were $S = \{s_1, \dots, s_5\}$, with $s_1 < \cdots < s_5$. Answers could include basic terms or hesitancy. In this approach, if a participant answered that they felt ashamed 2-5, we considered the HFLTS as $[s_2, s_5]$ to maintain the hesitancy given in the answer. Using different levels of precision in the linguistic terms allows us to capture the hesitancy that is inherent in peoples' emotions. Then to define groups among participants we consider two partitions. The first partition was constructed from the values of Fruit Preference (Both, Supermarket, Too good to go) and Gender (Female, Male). The second partition was constructed from the values of Fruit Preference (Both, Supermarket, Too good to go) and Previous Knowledge of 'Too good to go' (Yes, No). Twelve groups were defined and emotions among these groups were compared. To this end, the centroid and consensus were computed and differences among groups were considered following equations (2) and (3).

5 Results

Out of the 181 valid participants, 70 (38%) had no apple preference as they chose both apples, 63 participants preferred the 'Too good to go' apple (34%) and 48(27%) preferred the Supermarket apple. Out of the 181 participants, 97 identified as Female and 125 participants had previous knowledge of 'Too Good to go'.

5.1 Statistical Approach

A Mixed design ANOVA has several assumptions that should be met. In the presence of multiple factors, we checked for approximate normality of the residuals of the model using a qqplot. The data did not appear to be skewed after visual inspection. Levene's test was used to check for homogeneity of variance because of the between-subjects design. Only one violation was found between males and females for the emotion 'happy' ($F(1,178) = 4.67$, $p = .032$).

A significant effect of Emotion was found ($F(3.35, 566.13) = 119.20$, $p < .001$, $\eta_p^2 = .41$) (see table 1 for descriptives). Meaning that the evaluations to the six emotions regarding food waste were different. This is an expected finding as one would expect low values for happy and indifference while higher values for angry, ashamed, guilty and sad. The interaction between Emotion and Previous Knowledge (see Figure 2) was also significant $F(3.35, 566.13) = 3.51$, $p = .012$, $\eta_p^2 = .02$. Lastly, the interaction between Emotion and Gender (see Figure 3.) was also found statistically different $F(3.35, 566.13) = 3.75$, $p = .008$, $\eta_p^2 = .02$. We performed post-hoc analysis on the statistically significant interactions. Due to the very high number of comparisons (sixty-six in each case), we decided to not add them in the main paper as they are not directly relevant to the hypothesis. Overall, most of the comparisons were significant (please see figure 2 and figure 3 for visual comparisons).The main effect of Fruit Preference, Gender, or Previous Familiarity in addition to the rest of the interactions not mentioned above did not reach statistical significance.

Table 1. Descriptive summary of the main effect of Emotion

	Emotion					
	Angry	Ashamed	Guilty	Happy	Indifferent	Sad
Mean	2,42	3,17	3,79	1,13	1,88	3,15
SD	1,22	1,21	1,10	0,41	1,10	1,26

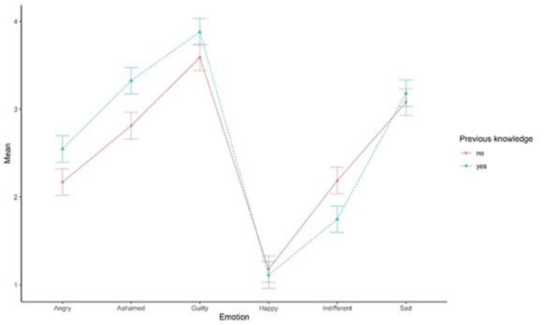


Figure 2. Mean evaluations of the six emotions while throwing away food and their error bars (Fisher's Least Significant Difference was used to enable within-Ss comparisons) are shown. The red line shows the values for the people who had no previous knowledge of 'Too good to go', while the blue one stands for the people who have previous knowledge.

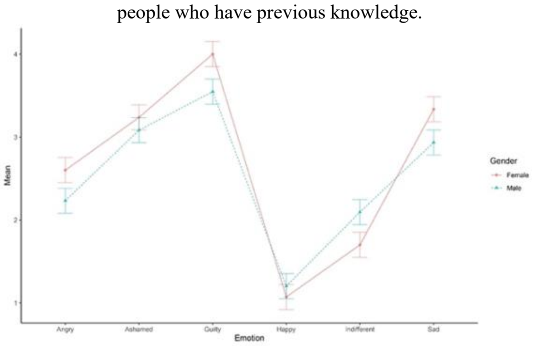


Figure 3. Mean evaluations of the six emotions while throwing away food and their error bars (Fisher's Least Significant Difference was used to enable within-Ss comparisons) are shown. The red line shows the values for the people who self-identified themselves as Female, while the blue one stands for the people who self-identify as Male.

5.2 Qualitative Reasoning Approach

Differences among emotions were found between the centroids with respect to the groups. In the first partition, differences between male and female were detected for all emotions except happy. For example, with respect to the emotion angry, males who preferred the Supermarket product are represented by a centroid of [2,2]. This indicates that their central opinion is a 2 when considering the scale from 1 (Not at all) to 5 (Very much) and it does not reflect any hesitancy when considering the opinion of the group as a whole. In contrast, females who preferred the Supermarket product are represented by a centroid of [2,3], indicating that their central opinion ranges from equal to slightly higher than that of males with the same produce preference. As their central opinion is formed by a range, it captures the hesitancy in the opinion of the group.

Table 2. Comparison of centroid emotions (partition 1)

Fruit Preference	Emotion: angry		ashamed		guilty		happy		indifferent		sad	
	M	F	M	F	M	F	M	F	M	F	M	F
Supermarket	[2, 2]	[2, 3]	[2, 2]	[4, 4]	[3, 3]	[4, 4]	[1, 1]	[1, 1]	[2, 2]	[1, 1]	[2, 2]	[4, 4]
Supermarket,Too good to go	[2, 2]	[2, 2]	[3, 3]	[3, 3]	[4, 4]	[4, 4]	[1, 1]	[1, 1]	[1, 2]	[1, 1]	[3, 3]	[3, 3]
Too good to go	[2, 2]	[3, 3]	[3, 3]	[4, 4]	[4, 4]	[5, 5]	[1, 1]	[1, 1]	[2, 2]	[1, 1]	[3, 3]	[4, 4]

In the second partition, differences between Too Good To Go familiarity were detected for all emotions except happy and guilty.

Table 3. Comparison of centroid emotions (partition 2)

Fruit Preference	Emotion: angry		ashamed		guilty		happy		indifferent		sad	
	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No
Supermarket	[2, 3]	[2, 2]	[4, 4]	[2, 2]	[4, 4]	[4, 4]	[1, 1]	[1, 1]	[1, 1]	[2, 2]	[4, 4]	[2, 2]
Supermarket,Too good to go	[3, 3]	[2, 2]	[3, 3]	[3, 3]	[4, 4]	[4, 4]	[1, 1]	[1, 1]	[1, 1]	[2, 2]	[3, 3]	[3, 3]
Too good to go	[2, 2]	[2, 2]	[4, 4]	[3, 3]	[4, 4]	[4, 4]	[1, 1]	[1, 1]	[1, 1]	[2, 2]	[3, 3]	[4, 4]

Finally, we computed the consensus corresponding to each group and emotion. In both partitions, we obtained values greater than 0.7 which is considered to be a high consensus given the granularity. Note that the consensus is considered from [0,1], therefore, this value indicates that there is little polarity in the opinions.

6 Conclusions and Future Research

There are several crucial outcomes shown in this paper. First of all, comparing the different results obtained when using classical statistics, we detected the tests were not able to find differences across the distinct preference groups (supermarket, supermarket/too good to go, to go to go) based on their emotional evaluations, whereas with our analysis based on qualitative hesitant terms it was possible. Secondly, both types of analyses were able to replicate previous findings on FW, since we found differences based on gender in relation to emotional responses. We have also extended previous work on FW since we investigated the interplay between previous knowledge of 'Too good to go' or gender identity on emotional valence. This approach has not been taken before as far as we know. We found that participants who have had previous knowledge of the 'Too good to go' app or are Female are more likely to rate higher in the negative emotions and lower in indifference. Thirdly, the combination of experimental methods with qualitative research is an approach that allowed us to gain a more nuanced understanding of the emotional issues connected to FW by separating perception from biased thought and connecting the resulting preference groups to salient emotions. Finally, we applied the interdisciplinary approach to the field of FW. This work represents the initial results of our efforts to comprehensively understand the drivers, emotional aspects, behavioral patterns, and cognitive factors connected to food waste. We believe these findings pave the way for further exploration and might have practical implications for policy.

Acknowledgments

This research has been partially supported by the PERCEPTIONS R. P. (PID2020-114247GB-I00), funded by the Spanish Ministry of Science and Information Technology.

References

Amina, B., & Azim, T. (2019). SCANCPECLENS: A Framework for Automatic Lexicon Generation and Sentiment Analysis of Micro Blogging Data on China Pakistan Economic Corridor. *IEEE Access*, *7*, 133876–133887.

Castro-Lopez, A., & Alonso, J. M. (2019). Modeling Human Perceptions in e-Commerce Applications: A Case Study on Business-to-Consumers Websites in the Textile and Fashion Sector. In Applying Fuzzy Logic for the Digital Economy and Society (pp. 115–134). Springer Nature Switzerland.

Crippa, M., Solazzo, E., Guizzardi, D., Monforti-Ferrario, F., Tubiello, F. N., & Leip, A. (2021). Food systems are responsible for a third of global anthropogenic GHG emissions. *Nature Food*, *2*(3), 198–209.

de Hooge, I. E., Oostindjer, M., Aschemann-Witzel, J., Normann, A., Loose, S. M., & Almli, V. L. (2017). This apple is too ugly for me!: Consumer preferences for suboptimal food products in the supermarket and at home. *Food Quality and Preference*, *56*, 80–92.

FAO - Food and Agriculture Organization of the United Nations. (2019). *The state of Food and agriculture*.

Food and Agriculture Organization of the United Nations. (2022). The future of food and agriculture: Drivers and triggers for transformation. In *The future of food and agriculture: Drivers and triggers for transformation*.

Gracia, A., & Gómez, M. I. (2020). Food sustainability and waste reduction in Spain: Consumer preferences for local, suboptimal, and/or unwashed fresh food products. *Sustainability (Switzerland)*, *12*(10).

Huang, J., & Koroteev, D. D. (2021). Artificial intelligence for planning of energy and waste management. *Sustainable Energy Technologies and Assessments*, *47*(January), 101426.

Kummu, M., de Moel, H., Porkka, M., Siebert, S., Varis, O., & Ward, P. J. (2012). Lost food, wasted resources: Global food supply chain losses and their impacts on freshwater, cropland, and fertiliser use. *Science of the Total Environment*, *438*, 477–489.

Liem, D. G., Toraman Aydin, N., & Zandstra, E. H. (2012). Effects of health labels on expected and actual taste perception of soup. *Food Quality and Preference*, *25*(2), 192–197.

Montserrat-Adell, J., Agell, N., Sánchez, M., Prats, F., & Ruiz, F. J. (2017). Modeling group assessments by means of hesitant fuzzy linguistic term sets. *Journal of Applied Logic*, *23*, 40–50.

Ponis, S. T., Papanikolaou, P. A., Katimertzoglou, P., Ntalla, A. C., & Xenos, K. I. (2017). Household food waste in Greece: A questionnaire survey. *Journal of Cleaner Production*, *149*, 1268–1277.

Porro, O., Agell, N., Sánchez, M., & Ruiz, F. J. (2022). A Multi-perceptual-Based Approach for Group Decision Aiding. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13408 LNAI*, 15–25.

Rockström, J., Gupta, J., Qin, D., Lade, S. J., Abrams, J. F., Andersen, L. S., Armstrong McKay, D. I., Bai, X., Bala, G., Bunn, S. E., Ciobanu, D., DeClerck, F., Ebi, K., Gifford, L., Gordon, C., Hasan, S., Kanie, N., Lenton, T. M., Loriani, S., … Zhang, X. (2023). Safe and just Earth system boundaries. *Nature*.

Rodriguez, R. M., Martinez, L., & Herrera, F. (2012). Hesitant fuzzy linguistic term sets for decision making. *IEEE Transactions on Fuzzy Systems*, *20*(1), 109–119.

Russell, S. V., Young, C. W., Unsworth, K. L., & Robinson, C. (2017). Bringing habits and emotions into food waste behaviour. *Resources, Conservation and Recycling*, *125*(March), 107–114.

Sörqvist, P., Hedblom, D., Holmgren, M., Haga, A., Langeborg, L., Nöstl, A., & Kågström, J. (2013). Who needs cream and sugar when there is eco-labeling? Taste and willingness to pay for “eco-friendly” coffee. *PLoS ONE*, *8*(12).

United Nations Environment Programme. (2021). Food Waste Index Report 2021. In *UNEP*.

United Nations: Department of Economic and Social Affairs. (2022). *The Sustainable Development Goals Report 2022*.

Zhu, J., Luo, Z., Sun, T., Li, W., Zhou, W., Wang, X., Fei, X., Tong, H., & Yin, K. (2023). Cradle-to-grave emissions from food loss and waste represent half of total greenhouse gas emissions from food systems. *Nature Food*, *4*(March).

Qualitative Monitoring of the Consequences of AI Solutions in Safety-Critical Systems

Mark Tappe,[1] **Benjamin Kelm,** [2] **Oliver Niggemann,** [1] **Stephan Myschik** [2]

[1] Institute of Automation Technology, Helmut Schmidt University, Hamburg, Germany
[2] Institute for Aeronautical Engineering, University of the Bundeswehr, Munich, Germany
mark.tappe@hsu-hh.de,

Abstract

Today, Cyber-Physical Systems (CPS) are often used in safety-critical situations. More and more, Artificial Intelligence (AI) and especially data-based methods, i.e. Machine Learning (ML), are used to increase the adaptability of systems. This immediately leads to a security risk, since data-based methods usually learn a black-box model (e.g. neural network or reinforcement learning). To still use these AI methods for safety-critical systems, like anomaly detection, optimisation or reconfiguration tasks, a supervision tool is needed.

In order to enable safe operation of data-based ML algorithms and to make statements about the stability of the system we present an implementation of qualitative monitoring of the system behaviour in the context of reconfiguration. This leads to the next problem, as a qualitative state prediction tends to branch infinitely for complex systems. Our approach limits the state prediction to the states with immediate impact. To achieve this goal and to visualise the effects for a supervision task a virtual structure similar to decision trees is implemented to generate an overview of the upcoming predicted system states. In addition, the behaviour of the system variables is extracted from the qualitative states in order to determine the risk of a predicted state.

In summary, this algorithm acts as an independent supervision agent for various AI/ML algorithms and alerts when risks are detected during operation. We can show that different reconfiguration options for a CPS with abnormal behaviour can be successfully evaluated in order to transfer the CPS as safely as possible to a new state.

1 Introduction

Cyber-Physical Systems (CPS) are very prevalent in our modern times, as the integration of microcomputers and other advanced technology offers a significant impact for a systems computational and communicative capabilities, see (Baheti 2011) and (Wolf 2009). To improve their performance a high level of technical expertise is required, which is often associated with high costs. Therefore, Machine Learning (ML) algorithms are often used for optimisation tasks based on existing data sets. However, once AI and data-based modelling determine the way a system operates, we lose predictability. This issue is of utmost importance because AI solutions, particularly data-based methods like many ML algorithms, typically create black box models based on given data (Tjoa and Guan 2021) and (Wan et al. 2021). When a system's behaviour is solely determined by measurement data, it cannot be fully defined. In safety-critical applications, this poses significant risks as infallibility cannot be verified. Although some solutions based on ML approaches, such as the Safety+AI approach of (Gheraibia et al. 2019), have been researched, we aim to focus on qualitative reasoning instead.

> "Reasoning about, and solving problems in, the physical world is one of the most fundamental capabilities of human intelligence and a fundamental subject for AI."

These words of Bredeweg (Bredeweg 2003) show very well our motivation for our approach. Our goal is to design a supervision agent that is able to monitor the behaviour of a system and to estimate the consequences of AI interventions. In theory an extensive numerical simulation would be able to evaluate those consequences very accurate, but especially for CPS, which combine computational science with engineering disciples this is not a trivial task and such a simulation is often not available. For this purpose, we investigate the possibility of using qualitative system models, based on a general system description, instead of complex simulations.

The benefits of such a prediction approach are examined in the joint project (K)ISS[1]. The aim of the project is to monitor the safety-critical life support system of the ISS module COLUMBUS. We aim to reconfigure the system by activating redundant components based on detected faults, ensuring effective recovery. To validate the reconfiguration process and assess different AI decisions, we successfully implement our approach for a supervision agent.

In section 2 we will explore general concepts related to qualitative system representations, and then in section 3 we will present our solution based on qualitative reasoning. The application of this will be in section 4 using a simulated environment. Finally, we will conclude this work in section 5 and provide an outlook on future tasks and challenges.

[1](K)ISS is part of dtec.bw®, see Acknowledgements for funding information
https://dtecbw.de/home/forschung/hsu/projekt-kiss

2 State of the Art

Before our solution is presented in section 3, we will first present a general overview of current approaches and explain their shortcomings, which we encountered during our research.

Safety Analysis of AI and the Shortcomings of Data-Based Models

As long as system measurement data is available the behaviour of a CPS can be learned. A basic application is to learn and formulate this behaviour in form of a timed automaton. As an example for how a automaton can be learned, we look at the algorithm of HyBUTLA, presented by (Niggemann et al. 2021). This algorithm constructs a timed automaton, which can be learned from system measurements, to describe the behaviour of a system, see Figure 1. In general the steps to learn the behaviour of such a system can be described with:

0: Record and synchronize the signals of the CPS.
1: Generate a list of discrete events.
2: Construct a tree based on the recorded events.
3: Simplify the tree by merging similar nodes.

The BUTLA algorithm, which depends on positive data examples, still has shortcomings. In certain cases, anomalies can occur that are not identified or are incorrectly identified. This happens because the data of error cases is not available and therefore there is a deficit of information.

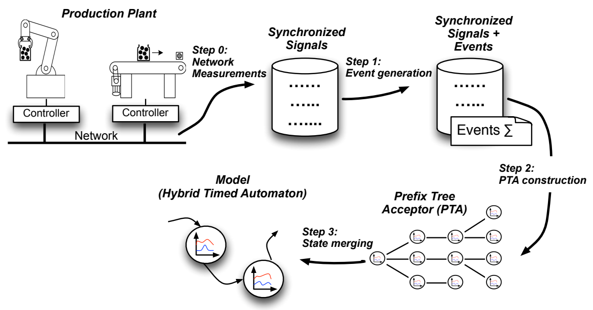


Figure 1: General concept of learning a timed automaton, the steps refer to the HyBUTLA algorithm, see (Niggemann et al. 2021)

Other data-based solutions, like many different ML algorithms, are commonly used, but their black-box nature hinders understanding and verification, especially of their internal workings - see (Tjoa and Guan 2020). The idea to translate and encode the behaviour of those AI models is part of the research of Explainable Artificial Intelligence (XAI). But most approaches of XAI are not universally valid. Instead, methods like saliency maps help with specific use cases like image recognition, but are more difficult to apply to decision process models. They can be used, but even if a correlation between input and output can be established, the result is by far not sufficiently precise enough to determine the internal decision process. Conventional algorithms such as decision trees, which do not operate on the same basis as AI algorithms, are much more reliable. But as shown in the work of (Wan et al. 2020), they lack in performance. They have shown the accuracy of decision trees in comparison to neural networks in image detection is behind by up to 40%.

Our approach emphasises the importance of decision tree reliability and combines it with qualitative reasoning for the safety assessment algorithm.

Qualitative Reasoning based on QSIM

In contrast to data based models, we can describe a system instead by its qualitative behaviour. The qualitative behaviour of a system is based on available system knowledge, which also grants information about non-measurable states. A promising concept about qualitative description of system behaviour was presented by (Kuipers 1986) in the papers about the QSIM algorithm. The used notation has been recognised by various scientists (Simon 1991; Say and Kuru 1996; Trave-Massuyes, Ironi, and Dague 2003), which is why we will also use its notation in this paper to describe a qualitative system.

The theory behind QSIM is to mimic the differential equations of classical systems with qualitative differential equations (QDE). A QDE would describe how a qualitative state can change. For each parameter P (which is basically a system variable) the qualitative state QS would be defined at a qualitative point in time t_i in the form of a tuple consisting of a discrete qualitative value and the direction of change. An example is given with:

$$QS(P, t_i) = \langle val, dir \rangle \tag{1}$$

The discrete value val can be defined as a single point value or as a pair of values specifying an interval in which the current qualitative value lies. In order to capture the change of a state, it is assigned an additional direction of change dir, which can take one of three variants: steady, increasing or decreasing. In addition, there is a discrete range of values for each parameter called the quantity space, which contains all known discrete values of that parameter - known as landmark values. To include multiple qualitative states in this kind of formulation a set F containing multiple parameters $F = \{P_i, ...\}$ can be created. Based on this a whole system can be defined by $QS(F, t_i)$.

To represent the QDE, which define the qualitative behaviour of a system, a set of constraints is needed, each limiting the possible transitions of the qualitative states opf the parameters. A comprehensive list is shown in Table 1. To depict a more complex ordinary differential equation, the equation can be separated in multiple elementary functions, which can then be translated in qualitative constraints. In some cases a constraint might change if the system reaches or leaves a set operating point. This can be handled by defining restrictions, which define which constraints apply for a given set point. A geometric function such as the sine can represent its cyclic effect with restrictions and alternating between M+ and M- constraints.

ADD(X, Y, Z)	$Z(t) = X(t) + Y(t)$
DERIV(X, Y)	$dX/dt = Y(t)$
M+(X, Y)	$X(t) = f(Y(t))$, where $f' > 0$
M-(X, Y)	$X(t) = f(Y(t))$, where $f' < 0$
MINUS(X, Y)	$X(t) = -Y(t)$
MULT(X, Y)	$Z(t) = X(t) * Y(t)$
CONST(X)	$X(t) =$ constant value

Table 1: List of Qualitative Constraints, complemented version of (Say and Kuru 1996)

Qualitative reasoning, similar to QSIM, has been researched and developed in the field of discrete model diagnosis. These approaches are often specific to certain toolboxes and proprietary applications, see (Williams et al. 2003; Struss and Price 2003). The fundamentals are widely known and were the focus of multiple research papers (de Kleer and Brown 1984; Dvorak and Kuipers 1989; de Kleer 1993), but since the year 2000 the application of qualitative simulation shifted. The numerical simulations became more reliable thanks to the increased computing power of computers, and the qualitative analyses were used more for the theoretical discussion of abstract systems and interrelationships, such as the effects on the population of species in (Salles and Bredeweg 2006).

In (Bredeweg 2003) the main issues and some open tasks of qualitative simulation back in 2003 were highlighted, particularly the modality of qualitative systems. On the one side this modality allows users to create diverse model libraries, which can be reused in different ways, but on the other hand each qualitative analysis needs a different degree of abstraction and detail and a uniform system did not exist back then. This problem continued with a lack of integration in standard engineering and research tools. In (Klenk et al. 2014) this problem got tackled by combining the usage of Modelica models with the ideas of qualitative reasoning. They achieved the goal to generate the qualitative model mapped upon existing modelica models, which negates the need for an additional modelling step. On the other hand we transferred the principles of QSIM and QDEs into the modern programming language python, which is especially well used in the machine learning community as another implementation. In this paper we will not further expand on the topic of implementation, but instead focus on the concept how this qualitative description can be used to evaluate the behaviour of a system. Still we are taking a custom take on the implementation to focus the constraints more on system dependencies instead of ideal QDEs.

Identification of Anomalies and Faults

For the sake of completeness, the need for identification and diagnosis of errors should be noted. One may assume that a failure is feasible via the QDEs defined above, but their algorithm, depending on implementation, cannot deduce the source of a defect. Still the underlying fundamentals of neglecting a specific mathematical model can be applied as well.

The work of (de Kleer and Williams 1987) shows how the shift of model-based diagnosis shifted from specific fault models towards the tracking of an inconsistent behaviour as indicator of a fault. Based on this, there are various alternative methods for detecting anomalies and faults that do not even require a mathematical model, as CPS usually provide a comprehensive database. These data-based algorithms can be evaluated as multi-time variant data sets and serve as a basis to describe the system behaviour of the plants from observations. Based on this data, it is then possible to create data-based models such as the Univariate Fully-Connected AutoEncoder (UAE), whose good performance was described by Garg et al. (Garg et al. 2022). However, their limitations were also pointed out, as these solutions are often limited to a specific use case, for example the UAE's performance decreased when used for a system with multiple operating states.

Still those algorithms perform well and there is no need to apply an additional supervision layer on top. In the later context, we assume that the identification of an anomaly and the diagnosis of faulty system components is available as a basis for the reconfiguration task.

3 Solution

In this section, we address implementing a qualitative monitoring agent for a CPS. We'll explain the generated input during reconfiguration, the use of QSIM basics in our supervision agent, and risk evaluation for predicted states. This guides the selection of a reconfiguration option with the lowest expected risk.

Assumptions for this paper: The system is faulty, but the cause is diagnosable and faulty components got detected. We aim to find a reconfiguration that adjusts the system structure to return to a safe workspace.

Reconfiguration

Generally, the goal of the supervision layer is to identify those possible configurations of the system that yield a safe and stable system. The actual identification of possible, valid configurations is typically performed by a reconfiguration program. Here, we would like to present the implemented reconfiguration algorithm *AutoConf* in brief, which is detailed and applied to ECLSS by (Kelm et al. 2022).

AutoConf, a qualitative model-based reconfiguration algorithm using Satisfiability Theory (SAT), was recently presented by Balzereit and Niggemann (Balzereit and Niggemann 2022). It can be used for the reconfiguration of hybrid systems and is divided into two main steps. In the first step a logical formula which represents the reconfiguration problem is created. In the second step, this formula is solved by a SAT solver.

The first step in creating the logical formula, known as the qualitative system model (QSM), involves generating causal graphs G that define the relationships between inputs and system states, e.g. a qualitative description of system dynamics. The inputs, represented as binary values (e.g., valve opened or closed), are denoted as $B = \boldsymbol{b}_1, ..., \boldsymbol{b}_k$. The causal graph is divided into positive ($G^+ = (V, E^+)$) and negative ($G^- = (V, E^-)$) subgraphs, indicating their influence on state variables. The nodes in the graphs include states and inputs ($V = \boldsymbol{x}_1, ..., \boldsymbol{x}_n, \boldsymbol{b}_1, ..., \boldsymbol{b}_k$), while the edges in the positive graph E^+ represent significant state increases when inputs are activated. The negative graph E^- represents significant state decreases.

Next, the algorithm encodes the causality into propositional logic by using symbols (low_{x_i} and $high_{x_i}$) to represent state limits. These symbols indicate whether a state is below the lower limit or above the upper limit and, consequently, imply the activation or deactivation of certain inputs. Binary logical connectives (implication [$\Rightarrow$], negation [$\neg$], conjunction [$\wedge$], and disjunction [$\vee$]) are used to formulate constraints. For instance, if a reservoir exceeds its limit, the formula implies either opening an outflow or closing an inflow.

In the second step, a logical SAT solver is employed to solve the logical formula, utilising logical reasoning. If the formula is satisfiable, it means there exists an assignment of input variables that satisfies the formula. This assignment corresponds to the new configuration required to achieve a valid system state within a specified reconfiguration time Δt_{rcfg}. If the formula is not satisfiable, a reconfiguration is not possible, and the system may need to be shut down.

Generally there are multiple valid configurations that are solutions of the reconfiguration problem, which can be iterative listed by negating the previously found solution and searching for another solution. To identify the best solution, e.g. the solution with the lowest risk of instability, a supervision layer is required.

Qualitative Supervision

To assure a safe operation of safety-critical systems during after a detected fault, the reconfiguration needs to be evaluated to prevent malfunctions. Therefore we want to design an supervision agent, to monitor the qualitative consequences of such actions.

Previously we presented the QSIM algorithm by (Kuipers 1986) and described how it can be used to abstract the behaviour of a system. In contrast to QSIM we added the F+ and F- functions. These function behave similarly to the M+ and M- functions in the original, but additionally investigate the dependencies of 0-values. The added F+ and F- functions are not monotonously increasing or decreasing functions, but allow a saddle point behaviour at a discrete value of $\langle 0 \rangle$. This is due to the fact that the dependencies on the input configuration represent a dependency on binary values, which can be implemented more efficiently by allowing a steady 0-value. In the case a system component is not needed and therefore shutoff, the function can be deactivated and then take on the classical M+ or M- behaviour once the component is reactivated.

The qualitative variables are initialised at t_0 in the form of:

$$
\begin{aligned}
QS(P_i, t_0) &= \langle val, dir \rangle \\
&\quad val \in [0, \textit{too low}, \textit{low}, \textit{norm}, \textit{high}, \textit{too high}, +\infty] \\
&\quad dir \in [\textit{dec}, \textit{std}, \textit{inc}]
\end{aligned} \tag{2}
$$

The qualitative values val of those variables are discretised measured values, which are categorised as low, $norm$ or $high$ depending on the known limits of their working range or *too high* and *too low* if the boundaries are exceeded. Additionally, their current change of direction is depicted with dir - increasing, steady or decreasing.

Combining qualitative findings with reliable system representation allows us to use a decision tree structure to understand system behaviour. We introduce the Qualitative Analysis Tree (QuAT) for this purpose. A simplified example is illustrated in Figure 2. Starting from an initial qualitative state 0, we assess its constraints to find possible transitions (e.g., a and b). As transitions occur, new qualitative states emerge, and their constraints are evaluated for predecessor states. If a transition leads to a steady state or detects a risk (e.g., transition b), further evaluations cease. The topic of risk assignment will be covered in the upcoming subsection. Nonetheless, to predict the comprehensive system state, we also consider subsequent states of successors, as they might appear deceptively safe, as seen in Figure 2 ($0 \Rightarrow a \Rightarrow 1 \Rightarrow d \Rightarrow Risk$).

If each successor state is evaluated we would obtain a qualitative description of the entire system like the original QSIM application. However, this approach becomes incredibly complex due to its combinatorial nature. To address this challenge, we reduce the number of iterations for our qualitative evaluation. Predicting the behaviour over a short abstract time horizon can still be highly effective, as each discrete qualitative time-step represents a specific event or a significant change of parameter values. Long-term analysis often isn't necessary as short-term defects have more serious consequences that require immediate prevention. Any negative long-term effects can be corrected with ongoing reconfiguration inputs.

The supervision agent's goal isn't finding optimal transitions but spotting safety-critical states after transitions. At a minimum, the next states, including all possible transitions, are analysed.

Validation of Analyzed States

Once the system's behaviour can be qualitatively analysed, it allows for evaluating its behaviour as a predictive model for future steps. By performing the qualitative algorithm for each discrete event, the upcoming behaviour can be analysed. As mentioned before we can create a QuAT whose tree structure consisting of successor states allows us to determine the qualitative system behaviour in the next discrete time points. Valid transitions can be assigned a positive score based on the operating range of each parameter, indicating that those states are considered acceptable.

But how is this score defined? The operating range for each variable is known and therefore we can estimate if a qualitative value becomes *too high* or *too low*. If these parameters exceed predefined safety limits, they are identified as risky states. Predicted states, which direction of change is not $steady$ pose a minor risk as they, potentially lead to limit violations later on. By combining the evaluation of the qualitative values and their direction of change, a risk score can be estimated and assigned to each state.

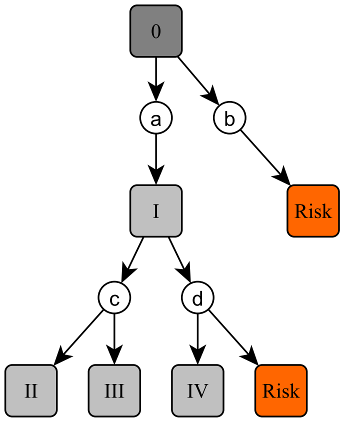


Figure 2: Exemplary QuAT for the representation of how supervision of system states is carried out. Each positional-state is represented with a box and roman numerals, while the possible transitional-states are marked with circles and small letters. States which oppose a risk are shown in orange.

Algorithm 1 Safety assessment based on qualitative risk assignment

Input: Current data of the system
Model: Qualitative system description, based on set F
Output: Risk-analysis of transitions

1: Discretize input data.
2: Initial qualitative state $QS(F, t_0)$ is set as $QS(active)$.
3: **while** qualitative prediction **do**
4: Analyze successor states of $QS(active)$.
5: Add all valid states to $ActiveList$.
6: remove $QS(active)$ from $ActiveList$
7: set next state from $ActiveList$ to $QS(active)$
8: **end while**
9: $\Rightarrow$ create $Tree$, with nodes of all qualitative states
10: **for** each $state$ in $Tree$ in bottom-up order **do**
11: **if** $state$ is leaf node **then**
12: Assign estimated risk
13: **else**
14: Update risk, based on successor nodes
15: **end if**
16: **end for**
17: **return** risk and qualitative behaviour

If further insights into the system are available and the risks associated with the interaction of specific parameters are known, these effects can be easily detected based on the qualitative state descriptions. Interdisciplinary effects should be considered when creating the qualitative state constraints. Similar to backpropagation in a neural network, risk estimations can be applied to predecessor states, enabling the assignment of a validity score to the entire QuAT, as during the creation of a QuAT the intermediate states can't be fully evaluated without knowledge about how their child states behaviour.

Algorithm 1 demonstrates an implementation example. Predefined qualitative system descriptions and system measurements are essential to define the initial qualitative state (Line 1-2). This includes the assignment of measurements to known qualitative discrete quantities, but also the representation of the system intervention that is to be studied. The current state undergoes qualitative simulation (Line 3-8) until the prediction horizon is reached or a steady state is attained. Analysed states are then organised into a tree structure, illustrating the system behaviour (Line 9). The risk assignment (Line 10-16) follows two main steps: Firstly, the tree is evaluated in a bottom-up manner, starting with the risk estimation of the leaf nodes. Afterwards their predecessors are updated primarily by their successors' risk. Once evaluation of all qualitative states is completed, the output contains the risk analysis of the current state's transition (Line 17).

The output of the safety assessment can depend on the use-case. One option would be to return the estimated risk for the current possible transitions, to validate if a specific transition should be avoided. Alternatively, the whole tree with the updated risk scores can be returned to present the system engineers a current overview of the system and its upcoming behaviour. The latter case is particularly important in situations where multiple safety-critical states are identified, requiring operators to navigate the system during challenging operations.

4 Application in Safety Assessment and Supervision of AI Solutions

This section covers the application of the monitoring agent for a CPS, here the COLUMBUS module of the ISS. The knowledge about upcoming system states, especially in terms of the assessed risk, is essential for a safe and secure operation.

CPS System Description - ISS Columbus ECLSS

The COLUMBUS module is the biggest contribution of the European Space Agency (ESA) to the International Space Station. Its purpose is to serve as a unique platform for different fields of research: Human physiology, biology, fundamental physics, material sciences and fluid physics. Furthermore, external experiment facilities allow the long-term and non-perturbed observation of the Earth and the universe. The European laboratory is operated by the COLUMBUS Control Center at the German Space Operations Center nearby Munich (Doyé 2012).

The most critical and vital system of the COLUMBUS module is the Environmental Control and Life Support System (ECLSS), whose topology is shown the process flow diagram in figure 3. It consists of a supply (ISFA) and return (IRFA) fan assembly, a redundant pair of cabin fan assemblies (CFA 1/2), a temperature control valve (TCV), which distributes the airflow into two redundant cooling and condensation cores (Core 1 and 2) within the condensate heat exchanger (CHX) to cool and dehumidify the air.

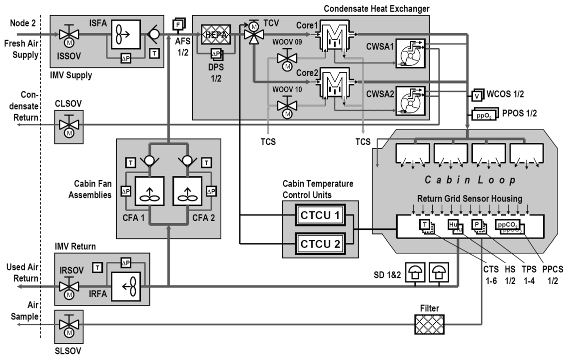


Figure 3: Cabin Loop of ISS ECLS-System by (Doyé 2012)

The airflow is then channelled into the cabin, where it mixes with the cabin air. To refresh the air and ensure smoke detection, a minimum volumetric flow rate has to be passed by the smoke detectors (SD 1/2) and is returned by the ISFA and recycled in part through the CFAs. The thermal control system (TCS) is composed of the Cores, the coolant and external heat exchangers and is controlled by the redundant cabin temperature control units (CTCU 1/2).

Additionally, there are multiple sensors, measuring the volumetric airflow (AFS), pressure differentials across fans and filter (ΔP or DPS), partial pressure of O_2 and of CO_2 gas (PPOS/PPCS), cabin temperature (CTS 1-6), humidity (HS 1/2) and the total pressure (TPS 1-4).

Reconfiguration of a Fault Case

Consider the following hypothetical failure case for illustration purposes: An accident occurs in the COLUMBUS module during an experiment, resulting in the failure of Cooling Core 1. The cabin's pressure has increased beyond the threshold due to gas leakage, and the hatch has been closed after the accident. The initial system state before reconfiguration is represented by

$$\begin{aligned} \boldsymbol{x}^0 &= [T_c, \phi_c, \dot{V}_{AFS}, p_c]^T \\ &= [303\,\mathrm{K}, 0.50, 500\,\mathrm{m}^3/\mathrm{h}, 103.5 \times 10^3\,\mathrm{Pa}]^T \end{aligned}$$

and the input configuration by

$$\begin{aligned} \boldsymbol{b}^0 = [&b_{\mathrm{ISFA}}, b_{\mathrm{IRFA}}, b_{\mathrm{CFA}_1}, b_{\mathrm{CFA}_2}, ... \\ &b_{\mathrm{TCV}_1}, b_{\mathrm{TCV}_2}, b_{\mathrm{C}_1}, b_{\mathrm{C}_2}]^T \\ = [&1, 0, 0, 1, 1, 0, 1, 0]^T. \end{aligned} \tag{3}$$

We thus have only ISFA, CFA2 and one cooling branch (TCV1, C1) activated, which corresponds to the default configuration, where the used air is returned over the hatch opening.

We also find, by an underlying fault diagnosis algorithm, that two actuators have failed. The health state is given by

$$\boldsymbol{h}^0 = [1, 1, 1, 0, 1, 1, 0, 1]^T. \tag{4}$$

Using the causal graph, the reconfiguration algorithm classifies the inputs into inflows and outflows. These are then transformed into a logical set of formulas using *AutoConf ext*. The formulas aim to answer the question:

> Which inputs do I need to open or close to bring the corresponding state within acceptable bounds?

An excerpt of the logical formula demonstrates the implications of a high temperature, where either one of the cooling cores (b_7 or b_8) or the ISFA fan (b_1) need to be activated:

It shows the implications of a high temperature, which are to switch on either one of the cooling cores (b_7 or b_8) or to switch on the ISFA fan. The negation of the pre-reconfigured inputs (b^0) excludes inputs that are already reconfigured. Actuator dependencies and internal flow structures are also included in the logical formula.

The logical formula is then checked for satisfiability using Z3. If it is satisfiable, a model (input assignment) that satisfies the formula can be obtained. In this fault case, the formula is satisfiable, and the algorithm proposes a new input configuration to recover the system:

$$\boldsymbol{b} = [1, 1, 0, 0, 0, 1, 0, 1]^T. \tag{5}$$

By activating the ISFA and the second cooling branch (TCV2 and C2), the pressure can be reduced, and the temperature can be lowered. Note that there exist multiple valid configurations (e.g. CFA1 could also be switched on). If the logical formula is not satisfiable, the system is shut down. Alternatively, constraints can be relaxed to lower the system requirements and prioritize certain state variables. The output is then presented as a list of possible configurations that solve the logical formula, the supervision agent will then select the safest system intervention.

Supervision of the Reconfiguration

The reconfiguration evaluates the current system state and determines a possible system configuration on the basis of the system's stability status, which is intended to bring the system to a stable state in the event of an anomaly. This procedure was described before using the system pressure and the cabin temperature as examples. As long as the logical formula can be solved with the *AutoConf ext* algorithm, several alternative configurations in the form of equation 6 can usually be determined. All of them can remedy the anomaly that has occurred as they solve the logical formula presented in the reconfiguration approach.

$$\begin{aligned} \boldsymbol{b}_0 &= [10001000]^T \\ \boldsymbol{b}_1 &= [11001000]^T \\ \boldsymbol{b}_2 &= [01101000]^T \\ \boldsymbol{b}_3 &= [00100100]^T \\ \boldsymbol{b}_4 &= [01100101]^T \end{aligned} \tag{6}$$

The next step is to select the most suitable system configuration. For this purpose, we use the qualitative evaluation procedure to determine the risk of the possible consequential states and to select the safest variant. In our application case we concentrate on the creation of the model on the basis of simplified system dependencies, because these can be derived from the system representation, see figure 3. This approach of using the knowledge of the system structure as a basis is always possible independent of the data basis and the existence of any simulation. With this knowledge we can formulate simplified qualitative equations for each state of equation 3 in the form of:

$$\begin{aligned} T_c &= +T_{Act} - T_{ISFA} - T_{C1||C2} \\ \phi_c &= +\phi_{Act} - \phi_{ISFA} - \phi_{C1|C2} \\ \dot{V}_{AFS} &= +\dot{V}_{ISFA} - \dot{V}_{IRFA} \\ p_c &= +p_{ISFA} - p_{IRFA} \end{aligned} \tag{7}$$

The equations 7 still need to be converted to the QSIM notation to be used for qualitative evaluation. Therefore we define the qualitative behaviour of the system based on its constraints and introduce auxiliary variables. the qualitative constraints are then shown using the example of the cabin temperature T_c in equation 8:

$$\begin{aligned} &F^-(T_{ISFA}, b_{\mathrm{ISFA}}), \\ &F^-(T_{CHX}, b_{\mathrm{C1||C2}}), \\ &F^+(T_{Act}, \mathit{Activity}), \\ &ADD(T_{ISFA}, T_{CHX}, T_{nSum}), \\ &ADD(T_{Act}, T_{nSum}, T_c) \end{aligned} \tag{8}$$

In this case, the temperature T_c can be understood as the sum of the negative and positive effective parameters. On the one hand, the astronauts' activity lead to an increase in temperature, and on the other hand, the colder supply air through the ISFA and the cooling core work against it.

Finally if all qualitative system equations are defined, the list of reconfiguration options can be tested and validated. Based on current system data the qualitative variables can be initialised and the algorithm 1 can be executed. The QuAT which was introduced before can't be utilised for the visualisation of the system, because it is far too complex to present the results here in this place as it contains thousands of states. Instead the Table 2 shows a validation of the different reconfiguration options. For each of the state variables, which were defined in equation 3, we can create their own QuAT and analyse the predicted risk for each reconfiguration option. Overall this allows an estimation of how a specific configuration affects the different state variables and therefore an initial guess on which reconfiguration to apply. The total risk assumptions can be compared to suggest the option with the least transitions into risky operations.

An experienced operator might favour a configuration with a better performance for one specific state, based on the current fault diagnosis, but we select the option with minimal expected overall risk. In this case reconfiguration option $\boldsymbol{b}_3$ is considered optimal with the least totaled calculated risk. It performs well because the states $\dot{V}_{AFS}$ and p_c are not directly affected by the configuration changes and therefore exist in a steady state without further disturbance, and therefore without any expected risk. It is arguable whether the qualitative equation of $\dot{V}_{AFS}$ defined in equation 7 should be affected by the fanspeed of the CFA_1 or CFA_2, but as long as the cabin door is shut, the circulating air is only defined by the supply (ISFA) and return (IRFA) fan assemblies.

	T_c	ϕ_c	$\dot{V}_{AFS}$	p_c
$\boldsymbol{b}_0$	42	42	33	38
$\boldsymbol{b}_1$	42	42	44	44
$\boldsymbol{b}_2$	34	34	38	33
$\boldsymbol{b}_3$	34	34	5	5
$\boldsymbol{b}_4$	42	42	38	33

Table 2: Risk score calculation for each reconfiguration option $\boldsymbol{b}_i$ based on the QuAT.

Measuring the effectiveness of qualitative state predictions is still an ongoing task in the project, but in its current form the supervision tool grants important insights by ranking the available reconfiguration options. For a given accident or failure multiple reconfiguration options can be identified, but in order to explicitly propose a solution and pave the way for autonomous deployment, a decision process must be integrated. By assessing the risk of upcoming qualitative states the decision can be forced to priories the well-being of the astronauts and a secure operation of the life support system.

5 Conclusion and future work

We present a novel approach that combines the fundamentals of qualitative system description with applications in artificial intelligence and system control theory. Our concept of qualitative prediction allows for the construction of an abstracted model based on fundamental knowledge of cause-effect relationships, enabling the prediction of complex system behaviour. Risk estimation plays a crucial role in selecting the appropriate configuration to recover from unintended system behaviour. However, the algorithm's performance currently hinders its application to systems with low response time. The combinatorial explosion of possible successor states is a computationally intensive task even with the proposed depth limitations. The operation time depends on factors such as the number of evaluated configurations, required depth, and the level of model detail. In our case the simplified qualitative equations in 7 analysed 2066 states in less than $30s$, increasing the amount of reconfiguration options to 10 increased the evaluation time to about $100s$ for roughly 7700 states and adding an additional state variables like the pressure at the first intersection increased the evaluation time to about $240s$. Of course the evaluation time depends on the used hardware, but the tendency is clear: Optimisation is necessary to improve the algorithm's efficiency.

The generation and definition of qualitative equations still requires expertise, and a poorly constructed model can limit overall functionality. Additionally, the abstract nature of qualitative solutions can pose challenges when converting them back into numerical contexts. To address these issues, the work of Say (Say and Kuru 1996) and Niggemann (Niggemann et al. 2021) shows promise in including system identification and merging of learned system behaviours, respectively. Incorporating these advancements into our approach of constructing the qualitative analysis tool (QuAT) can enhance its capabilities. To build upon these ideas it might be worthwhile to include data based concepts to set probabilities for the state transitions to account for normal behaviour and the most probable transitions. This could help to predict the risk of an action more accurate, or rather to help to identify planned and safe transitions. On the other hand the probability for failures and anomalies can't be based on data-sets, if those issues only occur in rare instances especially if the supervision tool is meant to supervised data based methods.

Furthermore, the presented qualitative evaluation can be used in other tasks. The evaluation of predicted system states is of particular interest in the task domain of approaches based on neural networks. In this context, we want to research the possibility to apply the qualitative reasoning to reinforcement learning by integrating the prediction of expected system states as action masking in internal reward policies. With this approach risky actions will be avoided during training. By doing so, we hope to optimise the learning behaviour and drastically reduce learning effort.

6 Acknowledgments

This research is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr. dtec.bw is funded by the European Union – NextGenerationEU. We're grateful for the support.

References

Baheti, R. 2011. Cyber-physical systems. *The impact of control technology*, 12.1: 161–166.

Balzereit, K.; and Niggemann, O. 2022. AutoConf: New Algorithm for Reconfiguration of Cyber-Physical Production Systems. *IEEE Transactions on Industrial Informatics*, 19(1): 739–749.

Bredeweg, B. 2003. Current Topics in Qualitative Reasoning. *AI Magazine*, Volume 24(Number 4).

de Kleer, J. 1993. *A view on qualitative physics*. MIT Press Cambridge, MA.

de Kleer, J.; and Brown, J. 1984. A Qualitative Physics Based on Confluences. *Artificial Intelligence*, 24: 7–83.

de Kleer, J.; and Williams, B. C. 1987. Diagnosing multiple faults. *Artificial Intelligence*, 32(1): 97–130.

Doyé, J. 2012. An Advanced Columbus Thermal and Environmental Control System. In *SpaceOps 2012 Conference*. Stockholm, Sweden: American Institute of Aeronautics and Astronautics.

Dvorak, D.; and Kuipers, B. 1989. Model-Based Monitoring of Dynamic Systems. In *IJCAI*, 1238–1243.

Garg, A.; Zhang, W.; Samaran, J.; Savitha, R.; and Foo, C.-S. 2022. An Evaluation of Anomaly Detection and Diagnosis in Multivariate Time Series. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6): 2508–2517. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

Gheraibia, Y.; Kabir, S.; Aslansefat, K.; Sorokos, I.; and Papadopoulos, Y. 2019. Safety + AI: A Novel Approach to Update Safety Models Using Artificial Intelligence. *IEEE Access*, 7: 135855–135869. Conference Name: IEEE Access.

Kelm, B.; Balzereit, K.; Moddemann, L.; Myschik, S.; and Niggemann, O. 2022. Application of a Model-based Reconfiguration Approach for the ISS COLUMBUS Environmental Control and Life Support System (ECLSS). *Proceedings of the 33nd International Workshop on Principle of Diagnosis, Toulouse, France*.

Klenk, M.; De Kleer, J.; Bobrow, D.; and Janssen, B. 2014. Qualitative reasoning with modelica models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Kuipers, B. 1986. Qualitative Simulation. *Artificial Intelligence*, 29(3): 289–338.

Niggemann, O.; Stein, B.; Vodencarevic, A.; Maier, A.; and Kleine Büning, H. 2021. Learning Behavior Models for Hybrid Timed Systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1): 1083–1090.

Salles, P.; and Bredeweg, B. 2006. Modelling population and community dynamics with qualitative reasoning. *ecological modelling*, 195(1-2): 114–128.

Say, A.; and Kuru, S. 1996. Qualitative system identification: deriving structure from behavior. *Artificial Intelligence*, 83(1): 75–141.

Simon, H. A. 1991. *Qualitative simulation modeling and analysis*, volume 5. Springer-Verlag.

Struss, P.; and Price, C. 2003. Model-based systems in the automotive industry. *AI magazine*, 24(4): 17–17.

Tjoa, E.; and Guan, C. 2020. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE transactions on neural networks and learning systems*, 32(11): 4793–4813.

Tjoa, E.; and Guan, C. 2021. A Survey on Explainable Artificial Intelligence (XAI): Towards Medical XAI. *IEEE Transactions on Neural Networks and Learning Systems*, 32(11): 4793–4813. ArXiv:1907.07374 [cs].

Trave-Massuyes, L.; Ironi, L.; and Dague, P. 2003. Mathematical foundations of qualitative reasoning. *AI magazine*, 24(4): 91–91.

Wan, A.; Dunlap, L.; Ho, D.; Yin, J.; Lee, S.; Jin, H.; Petryk, S.; Bargal, S. A.; and Gonzalez, J. E. 2020. NBDT: neural-backed decision trees. *arXiv preprint arXiv:2004.00221*.

Wan, A.; Dunlap, L.; Ho, D.; Yin, J.; Lee, S.; Jin, H.; Petryk, S.; Bargal, S. A.; and Gonzalez, J. E. 2021. NBDT: Neural-Backed Decision Trees. ArXiv:2004.00221 [cs].

Williams, B. C.; Ingham, M. D.; Chung, S.; Elliott, P.; Hofbaur, M.; and Sullivan, G. T. 2003. Model-based programming of fault-aware systems. *AI Magazine*, 24(4): 61–61.

Wolf, W. 2009. Cyber-physical systems. *Computer*, 42(03): 88–89.

Preliminary Experiments using LLMs for Design

John T Maxwell III , Johan de Kleer , Ion Matei , Maksym Zhenirovsky
Palo Alto Research Center
SRI International
{maxwell, dekleer, matei, mazhenir}@parc.com

Abstract

In analogy to using an LLM to generate a story on some topic, or Dall-E to generate an image, we can use LLMs to design a physical system to achieve a function. A system such as ChatGPT isn't a great designer, but it has two significant advantages. First, the designs it produces are approximately correct and thus we can use it as a starting point for developing a practical design. Second, it has a vast amount of knowledge about physical domains and is not limited to one domain. All other design tools have severe domain limitations. In short, it is an extremely general but sloppy designer. In this paper we show how the sloppy designs produced by ChatGPT can refined to produce practical designs.

1 Design experiment methodology

For all our design experiments we start with a known design (e.g., low-pass filter, power-train, op-amp) and construct a data set by simulating its behaviors over time that characterize its function. For example, for a low-pass filter we simulate its behavior at a frequency within its pass-band, and outside of it. We provide those input/output sequences to our automated designer to construct a system which produces the same input/output sequences. There are often multiple ways of achieving the same input-output behavior, so the designed system may not have the same topology or parameters as our original system. We are not trying to recreate the original design, but rather to automatically construct a design which behaves in the same way. There are usually an infinite number of ways a desired function can be achieved. Our approach finds a simpler one simply because the LLM [Devlin *et al.*, 2019] will typically find simpler ones.

An immediate challenge to using ChatGPT [Roumeliotis and Tselikas, 2023] is that it has a poor grasp of mathematics. Hence, we use ChatGPT only to generate a topology of components. ChatGPT cannot assign parameter values to components. That task is left to an optimizer which picks values for component parameters such that the function of the system is achieved.

2 Low-pass filter

Consider designing a low pass filter. A low pass filter is a circuit which reduces the high frequencies in the input signal and passes through unaltered low frequencies (hence the name). Figure 1 is a simple example of a low-pass filter. To construct the input-output data set let $R = 1K\Omega$ and

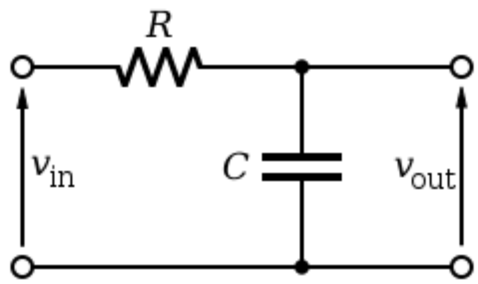


Figure 1: Low pass filter

$C = 0.1\mu F$. With those values the circuit has a cutoff frequency of approximately $1.6KHz$ (the frequency at which there is $3db$ loss in amplitude. To frame our experiment we simulate our circuit at $f = 1KHz$ and $f = 2KHz$. Assume the load impedance is $10K\Omega$. We then simulate the low-pass filter to construct two sequences of input/outputs. The output voltage time series corresponding to a sinusoidal input at $1KHz$ and $2KHz$ frequencies are shown in Figure 2. These time series will serve as ground truth for the design challenge.

3 Modelica and ChatGPT

We use Modelica [Fritzson, 2004] as our primary modelling tool. Modelica consists of a modeling language and a simulator so that we can test any designs that are discovered. The ChatGPT training set includes enough Modelica models that we can use ChatGPT 4.0 directly. We first construct a prompt which includes the library of Modelica components to choose among, the interface of the desired system, and the natural language description of the desired function. In order to design the low-pass filter we provide ChatGPT 4.0 the following prompt:

```
Here is a library of Modelica components:
```

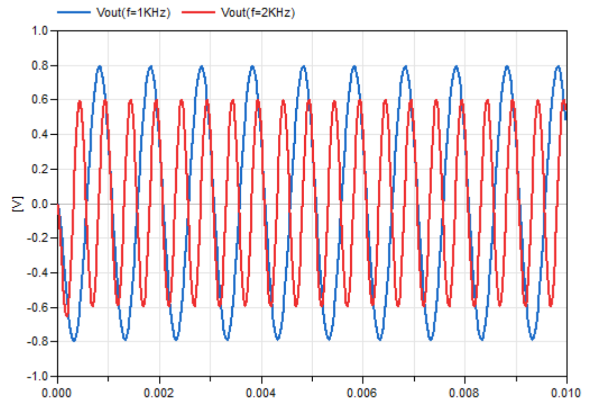


Figure 2: Output voltage for a sinusoidal input voltage $1KHz$ and $2KHz$ frequencies.

```
model Modelica.Electrical.Analog.Basic.Capacitor
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
end Modelica.Electrical.Analog.Basic.Capacitor;

model Modelica.Electrical.Analog.Basic.Ground
  Modelica.Electrical.Analog.Interfaces.Pin p;
end Modelica.Electrical.Analog.Basic.Ground;

model Modelica.Electrical.Analog.Basic.Inductor
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
end Modelica.Electrical.Analog.Basic.Inductor;

model Modelica.Electrical.Analog.Basic.Resistor
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
  Modelica.Thermal.HeatTransfer.Interfaces.HeatPort_a heatPort;
end Modelica.Electrical.Analog.Basic.Resistor;

model Modelica.Electrical.Analog.Ideal.IdealOpAmp3Pin
  Modelica.Electrical.Analog.Interfaces.PositivePin in_p;
  Modelica.Electrical.Analog.Interfaces.NegativePin in_n;
  Modelica.Electrical.Analog.Interfaces.PositivePin out;
end Modelica.Electrical.Analog.Ideal.IdealOpAmp3Pin;

Here is the interface for a low pass filter:

model two_port_interface
Modelica.Electrical.Analog.Interfaces.PositivePin p_in;
Modelica.Electrical.Analog.Interfaces.NegativePin n_in;
Modelica.Electrical.Analog.Interfaces.PositivePin p_out;
Modelica.Electrical.Analog.Interfaces.NegativePin n_out;
end two_port_interface;

Connect components from the library into a well-formed
Modelica model to create a low pass filter that extends
two_port_interface. Name the Modelica model ``low_pass_filter''.
```

ChatGPT outputs the following Modelica code:

```
model low_pass_filter "Low Pass Filter"
  extends step_function_test;

  Modelica.Electrical.Analog.Basic.Resistor
     R1(R=1000);
  Modelica.Electrical.Analog.Basic.Capacitor
     C1(C=1e-6);

equation
  connect(p_in, R1.p);
  connect(R1.n, C1.p);
  connect(R1.n, p_out);
  connect(C1.n, n_out);
end low_pass_filter;
```

We will address the fact that the indicated parameters are incorrect for the task. ChatGPT cannot find correct parameters even if they exist because it understands so little about equations. We have built a simple automatic renderer for Modelica models, Figure 3 is drawn by our renderer, not ChatGPT. Hence, the drawings appear awkward.

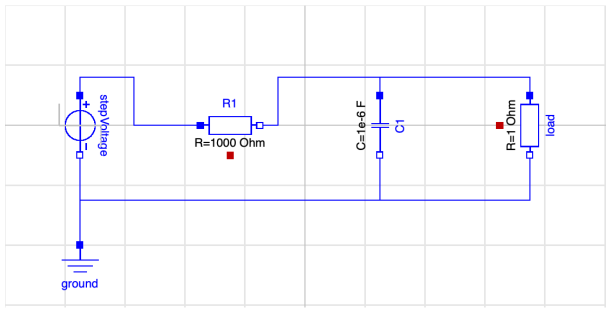


Figure 3: Low pass filter

4 Generate and Test

Figure 4 illustrates our approach to using ChatGPT in design. Once we have a Modelica model topology, we use the Sizer to

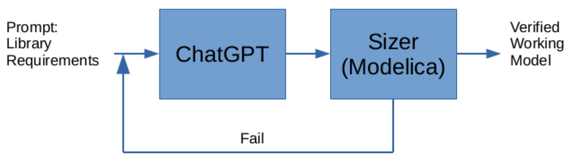


Figure 4: Ideal Framework: ChatGPT constructs a Modelica topology from the requirements and optimization is used to determine the exact parameters needed. If no parameters can be found for that topology that achieve the desired function, the generate and test loop continues.

determine the values of the parameters so that the input/output sequences match as closely as possible (through optimization).

We use Dymola to convert the Modelica model into an FMU. We then use a gradient-free descent method (Powell) to determine the best values for the parameters. We optimize the parameters against the desired input-output function data set. Thus we use ChatGPT to determine a topological model from specifications written in English, and continuous optimization to determine the parameters of the topology that match the requirements. In the case of the RC circuit, only the product RC is relevant, thus the optimizer can find various values for R and C, depending on the initial conditions of the algorithm. For example for one run with random initial conditions in $[0, 1]$ and $1KHz$ input frequency, the Sizer finds $R = 0.00969\Omega$, $C = 0.01041F$. Their product is roughly 10^{-4}, which is exactly the product of the R, C parameters for the ground truth case.

Note however that ChatGPT will not always generate topologies for which the Sizer can find any appropriate values for the system parameters. For example, in the case of a purely resistive circuit, the Sizer will never find correct parameters that can match the ground truth. Worse, the models ChatGPT constructs can be syntactically incorrect. These errors occur because LLMs are not general AIs and make many mistakes. The surprise is more than ChatGPT can often find almost correct designs.

Since ChatGPT is stochastic, every invocation of ChatGPT often yields a new topology. Therefore, one could just keep on calling ChatGPT over and over again until, hopefully, it comes up with a valid topology. This obviously yields very poor performance. Figures 5 and 6 show two faulty topologies ChatGPT generates.

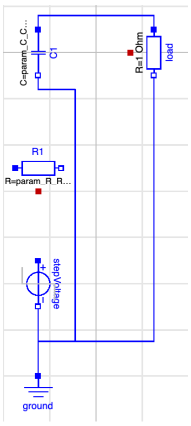


Figure 5: Bad low pass filter. This has a disconnected input and its output is always 0. It attenuates all frequences.

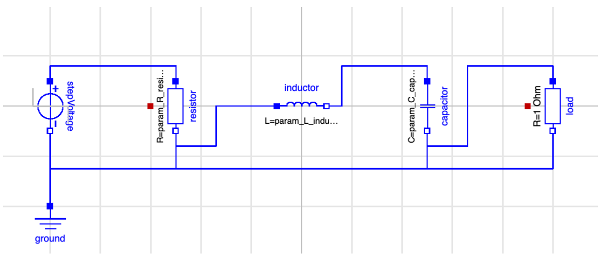


Figure 6: Another bad low pass filter. This one is syntactically correct, but output is always 0. It attenuates all frequences.

5 Generate and Repair

Instead of taking a pure "generate and test" approach, we could try to repair the designs that are syntactically invalid or that do not have the proper behavior.

One way to try to repair the designs would be to ask ChatGPT to redesign a faulty design given some information of what was wrong with it. We did some experiments to try this out. Since ChatGPT is stochastic, it was hard to tell whether the new information made a difference. So we tried each repair ten times both with and without the new information. For a few repairs it was more likely to produce a good design with the new information, but for most repairs there was no difference. In this case, using ChatGPT for repair was effectively a "generate and test" approach.

To perform our experiments we introduce a 3rd module to our framework. The repair module detects whether there is some simple syntactic reason the Modelica topology produced by ChatGPT cannot function. (It also checks for duplicate topologies.) It then attempts to repair the topology with a a simple local transformation. This is much less expensive than starting the Modelica optimizer. Figure 7 is the framework we use in our experiments.

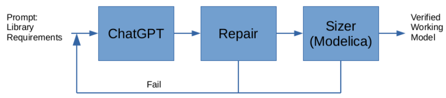


Figure 7: More efficient framework which repairs syntactically incorrect Modelica topologies.

Although we could have the repair module generate a sequence of designs to test, we instead have the repair module generate a model that represents a space of designs where the different design choices are represented by switches. Currently, we generate a sequence of designs to test from the space of designs, but in the future we hope to use ATMS[de Kleer, 1986] reasoning to search the space more efficiently than by using exhaustive enumeration.

The first thing that the repair module does is repair syntactically invalid models. For instance, ChatGPT sometimes leaves out the 'equation' keyword from the model. The repair module detects this and inserts an 'equation' keyword between the components and the connections.

The next thing the repair module does is to look for possible topological errors such as disconnected ports (Figure 5) and positive ports connected to ground (Figure 6). For each possible error, it generates a range of alternatives. For instance, it converts Figure 5 into Figure 8, and Figure 6 into Figure 9.

Once we have a repair space, we can generate candidate designs by enumerating switch values that are consistent with the 'oneof' constraints and passing the resulting design to the Sizer to determine optimal parameter values. Figures 8 and 9 contain valid low-pass filter designs, so this process produces a successful design in these cases.

6 Design of a Power Train

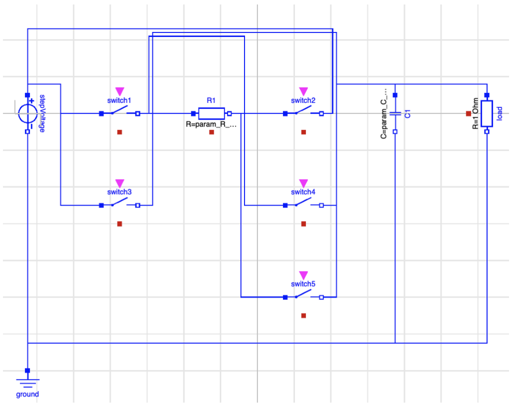
The Modelica Standard Library (MSL) contains an extensive collection of power train components. Figure 10 illustrates a simple vehicle power train. This model has a simple model of the road and driver.

The input-output function we desire is produced by a simulated driver (Figure 11).

The following ChatGPT prompt generates powertrain topologies.

```
Here is a library of Modelica components:

model Modelica.Mechanics.Rotational.Components.Brake
  Modelica.Mechanics.Rotational.Interfaces.Flange_a flange_a;
```




```
oneof(switch1,switch2,switch3)
oneof(switch4,switch1)
oneof(switch5,switch2)
```

Figure 8: Repair space for Figure 5.

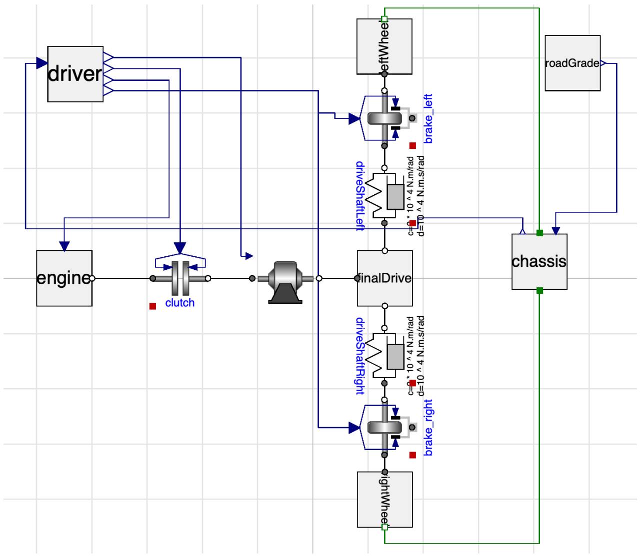
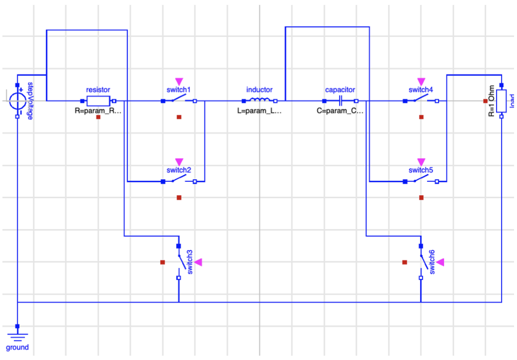


Figure 10: A working powertrain




```
oneof(switch1,switch2) oneof(switch1,switch3)
oneof(switch4,switch5) oneof(switch4,switch6)
```

Figure 9: Repair space for Figure 6.

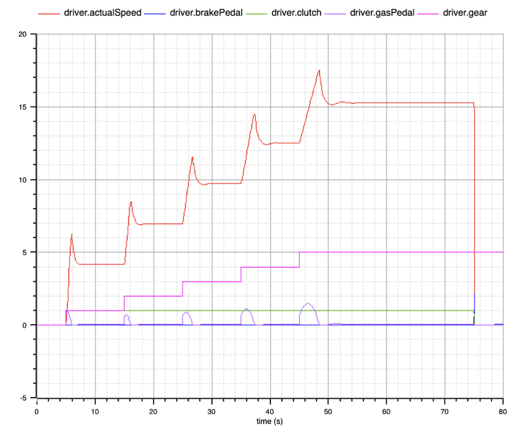


Figure 11: Desired power train behavior

```
   Modelica.Mechanics.Rotational.Interfaces.Flange_b flange_b;
   Modelica.Blocks.Interfaces.RealInput f_normalized;
end Modelica.Mechanics.Rotational.Components.Brake;

model Powertrain.Components.Chassis
   Modelica.Blocks.Interfaces.RealOutput speed;
   Modelica.Mechanics.Translational.Interfaces.Flange_a flange_a;
   Modelica.Mechanics.Translational.Interfaces.Flange_a flange_b;
   Modelica.Blocks.Interfaces.RealInput grade;
end Powertrain.Components.Chassis;

model Modelica.Mechanics.Rotational.Components.Clutch
   Modelica.Mechanics.Rotational.Interfaces.Flange_a flange_a;
   Modelica.Mechanics.Rotational.Interfaces.Flange_b flange_b;
   Modelica.Blocks.Interfaces.RealInput f_normalized;
end Modelica.Mechanics.Rotational.Components.Clutch;

model Powertrain.Driver.Driver
   Modelica.Blocks.Interfaces.RealInput actualSpeed;
   Modelica.Blocks.Interfaces.RealOutput gear;
   Modelica.Blocks.Interfaces.RealOutput clutch;
   Modelica.Blocks.Interfaces.RealOutput gasPedal;
   Modelica.Blocks.Interfaces.RealOutput brakePedal;
end Powertrain.Driver.Driver;
```

```
model Powertrain.Components.Engine
   Modelica.Blocks.Interfaces.RealInput pedal;
   Modelica.Mechanics.Rotational.Interfaces.Flange_b flange_a;
end Powertrain.Components.Engine;

model Powertrain.Components.GearBox
   Modelica.Mechanics.Rotational.Interfaces.Flange_a flange_a;
   Modelica.Mechanics.Rotational.Interfaces.Flange_b flange_b;
   Modelica.Blocks.Interfaces.RealInput gear;
end Powertrain.Components.GearBox;

model Powertrain.Driver.RoadGrade
   Modelica.Blocks.Interfaces.RealOutput grade;
end Powertrain.Driver.RoadGrade;

model Powertrain.Components.SimplifiedwheelRoad
   Modelica.Mechanics.Rotational.Interfaces.Flange_a flange_a;
   Modelica.Mechanics.Translational.Interfaces.Flange_b flange_a1;
end Powertrain.Components.SimplifiedwheelRoad;

Connect components from this library in a well-formed Modelica model
to create a drive train.
```

7 Generate and Behavioral Repair

ChatGPT 4.0 mostly generates bad topologies which are not repairable with the syntactic techniques just outlined. The Sizer does not find parameters which yield a correct function for any of the repairs. Three such bad designs are: Figures 12, 13, and 14.

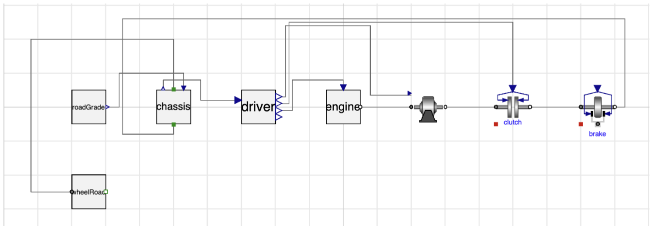


Figure 12: Problems: Brake output should be connected to wheelRoad, not chassis. Rotational flanges should not be connected to translational flanges. Missing a connection between driver.brakePedal and brake input.

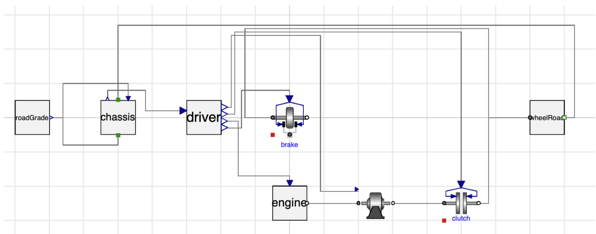


Figure 13: Problems: Brake input is connected to wheelRoad instead of output. Clutch is connected to wheelRoad instead of brake input. roadGrade is connected to chassis.flange_b (wrong type)

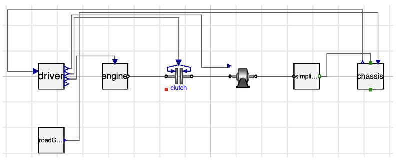


Figure 14: Problem: Missing brake

Consider the design in Figure 14. There is nothing in the design structure to indicate that the brake is missing. The problem only shows up in the behavior. In the ideal behavior, we see that when the brake pedal is pressed, the vehicle sharply decelerates (see Figure 11). This can be seen more clearly in Figure 15, where the extraneous variables have been removed.

Figure 14 doesn't use the brake pedal, so that is a clue. But what should the brake pedal be connected to? How do we know to add a brake?

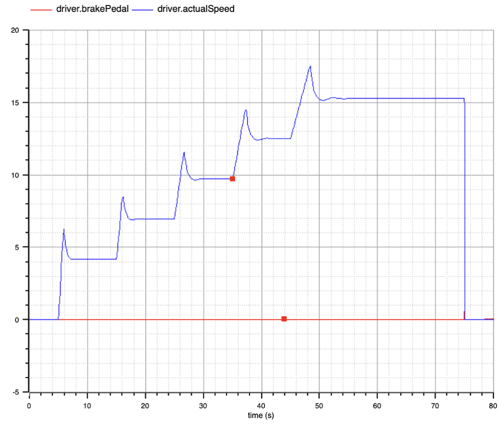


Figure 15: Behavior of brake pedal in power train

One way to determine that we are missing a brake is to look at the qualitative behavior of the brake pedal versus the speed in Figure 15. When the brake pedal is zero, then the speed can be positive or zero, and the first derivative can be positive, negative, or zero. However, when the brake pedal is positive, then the speed is always decelerating. So we can look for a component in our library that has that behavior.

Each component in the library has a unit test that exhibits the behavior of that component. The unit test for the brake is shown in Figure 16 and its behavior is shown in Figure 17.

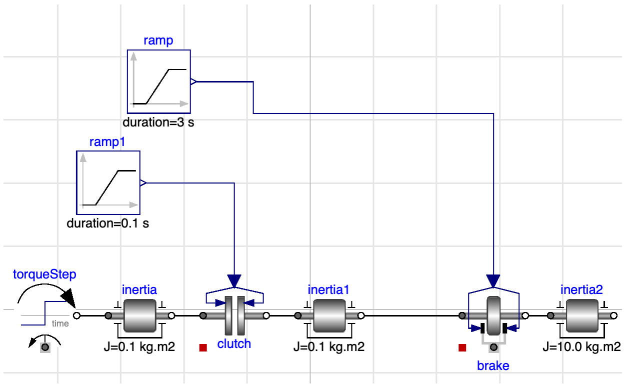


Figure 16: Unit test model for brake

Although it is not easy to see in Figure 17, the output variable starts decelerating when the input variable is (very slightly) positive. So this suggests that adding a brake might repair Figure 14. The first step to adding Figure 16 to Figure 14 is to convert the design by replacing the unit test scaffolding with the power train scaffolding. This produces Figure 18.

The next step is to merge Figure 18 and Figure 14. There are many ways to merge these two designs. If we assume that linear designs tend to remain linear, and that duplicate components are shared, then we can use code to zipper the

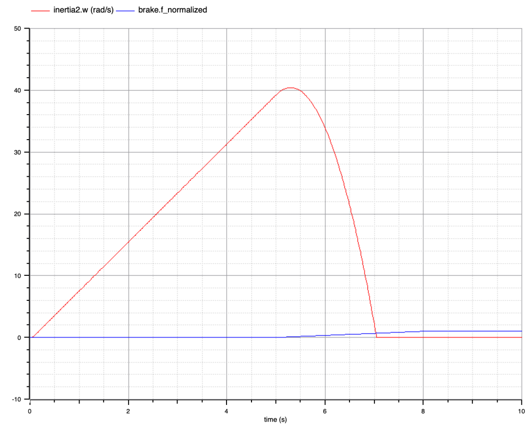


Figure 17: Behavior of brake unit test

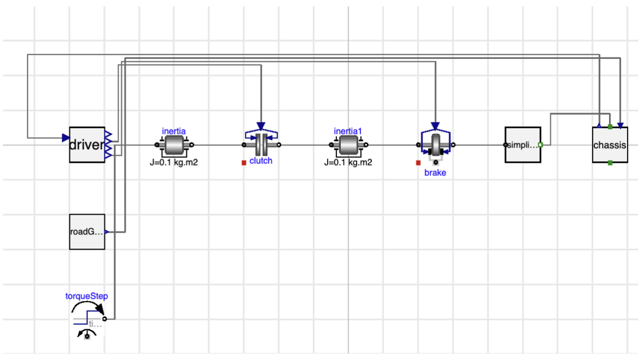


Figure 18: Brake unit test converted to power train

designs together to produce the power train in Figure 19.

We can also ask ChatGPT to merge the two designs through the following prompt:

```
Here are two Modelica models:

within Powertrain;
model vehicle1
   Powertrain.Driver.RoadGrade roadGrade;
   Powertrain.Driver.Driver driver;
   Powertrain.Components.Engine engine;
   Modelica.Mechanics.Rotational.Components.Clutch
      clutch (mue_pos=[0, 0.3], peak=1.1, cgeo=0.176, fn_max=6973);
   Powertrain.Components.SimplifiedwheelRoad
       simplifiedwheel_road(vehicle_mass=1520, radius=0.3);
   Powertrain.Components.Chassis chassis(vehicleMass=1520);
   Modelica.Mechanics.Rotational.Components.Brake
       brake(locked(fixed=true,start=true),
                 mue_pos=[0,0.3], fn_max=293200.0);

equation
   connect(engine.flange_a, clutch.flange_a);
   connect(driver.clutch, clutch.f_normalized);
   connect(driver.gasPedal, engine.pedal);
   connect(simplifiedwheel_road.flange_a1, chassis.flange_a);
   connect(roadGrade.grade, chassis.grade);
   connect(driver.actualSpeed, chassis.speed);
   connect(driver.brakePedal, brake.f_normalized);
   connect(brake.flange_a, clutch.flange_b);
   connect(brake.flange_b, simplifiedwheel_road.flange_a);
end vehicle1;

within Powertrain;
model vehicle2
   Powertrain.Driver.RoadGrade roadGrade;
   Powertrain.Driver.Driver driver;
   Powertrain.Components.Engine engine;
   Modelica.Mechanics.Rotational.Components.Clutch
      clutch(mue_pos=[0, 0.3], peak=1.1, cgeo=0.176, fn_max=6973);
   Powertrain.Components.GearBox gearBox;
   Powertrain.Components.SimplifiedwheelRoad
      simplifiedwheel_road(vehicle_mass=1520, radius=0.3);
   Powertrain.Components.Chassis chassis(vehicleMass=1520);

equation
   connect(engine.flange_a, clutch.flange_a);
   connect(clutch.flange_b, gearBox.flange_a);
   connect(driver.gear, gearBox.gear);
   connect(driver.clutch, clutch.f_normalized);
   connect(driver.gasPedal, engine.pedal);
   connect(gearBox.flange_b, simplifiedwheel_road.flange_a);
   connect(simplifiedwheel_road.flange_a1, chassis.flange_a);
   connect(roadGrade.grade, chassis.grade);
   connect(driver.actualSpeed, chassis.speed);
end vehicle2;

Merge these two modelica models to produce a new Modelica model
named vehicle within Powertrain.
The new Modelica model should represent a drive train.
```

This sometimes produces Figure 19.

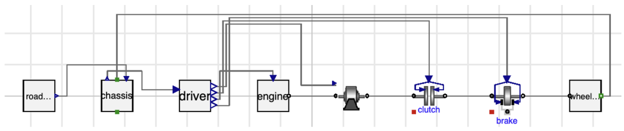


Figure 19: A valid powertrain generated by ChatGPT.

We can extend this qualitative behavior analysis to the other components in the library. For instance, the unit test for the engine is shown in Figure 20 and its behavior is shown in Figure 21.

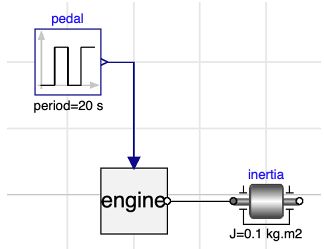


Figure 20: Unit test model for engine

When the engine pedal is zero, the output can be zero or positive and the first derivative can be positive, zero or negative. However when the engine pedal is positive at around 3 seconds, the output is accelerating. This suggests that the engine acts as an accelerator.

The unit test for the clutch is shown in Figure 22 and its behavior is shown in Figure 23.
When the clutch input is zero, then the output is zero. When the clutch input is positive then the output is positive and accelerating. This suggests that the clutch acts as an accelerator or as an on/off switch.

The unit test for the gear box is shown in Figure 24 and its behavior is shown in Figure 25.

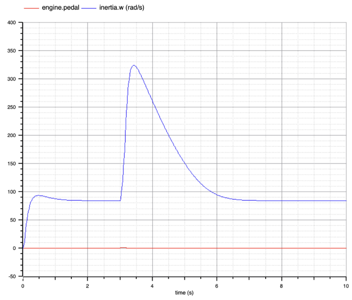


Figure 21: Behavior of engine unit test

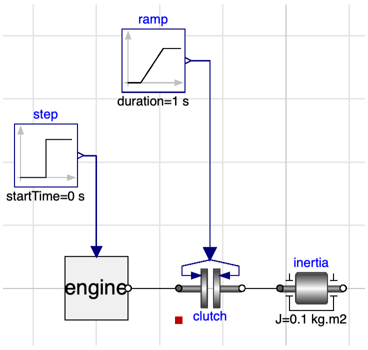


Figure 22: Unit test model for clutch

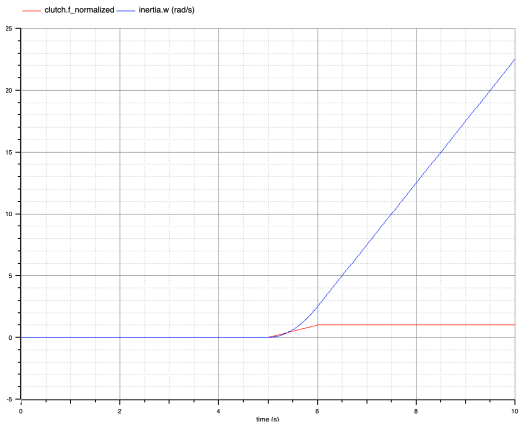


Figure 23: Behavior of clutch unit test

When the gear box input is zero, then the output is zero. When the gear box input is positive, then the output is positive and accelerating. When the gear box input is 2, then the average output is higher than when the gear box input is 1. This suggests that the gear box is acting as some sort of selector.

The qualitative behaviors of the brake, engine, clutch, and gear box can be detected in the desired behavior in Figure 11. This suggests that all of these components are necessary to

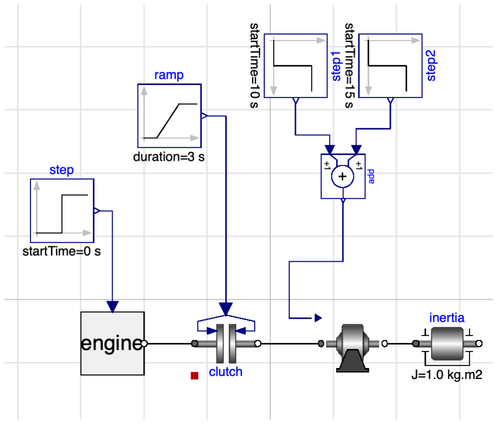


Figure 24: Unit test model for gearbox

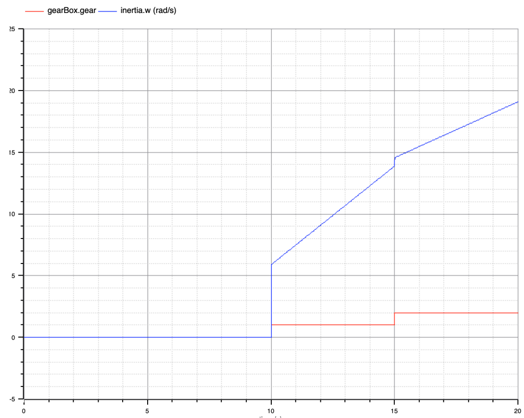


Figure 25: Behavior of gear box unit test

make a power train. If a candidate design is missing one of these behaviors, then it is worth trying to add the corresponding component.

This sort of qualitative analysis of component behavior can be extended to multi-component behavior as well. It works if a library of known designs has designs with qualitative behavior that can be detected. If a qualitative behavior is in the target behavior but not in a candidate's behavior then we can try adding the known design with that behavior to the candidate design. If a candidate design has a qualitative behavior that the target design does not have and the candidate has a known design with that behavior embedded in it then we can try deleting the known design from the candidate design.

8 Conclusion

The Sizer algorithm can be surprisly expensive and numerical simulation is often fragile within the optimization loop. Therefore, it makes more sense to qualitatively simulate every design before attempting to find the values needed for the parameters. [Klenk *et al.*, 2012].

This paper has illustrated that ChatGPT can be a powerful tool as part of a automated design process. In future work we plan to perform experiments on a wide variety of design problems and determine how large a space of designs can be covered. Introducing more QR promises to greatly speed up the search for possible designs.

References

[de Kleer, 1986] Johan de Kleer. An assumption-based tms. *Artificial Intelligence*, 28(2):127–162, 1986.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[Fritzson, 2004] P. Fritzson. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. Wiley-IEEE Press, Piscataway, NJ, 2004.

[Klenk *et al.*, 2012] Matthew Klenk, Johan de Kleer, Daniel G Bobrow, Sungwook Yoon, John Hanley, and Bill Janssen. Guiding and verifying early design using qualitative simulation. In *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 1097–1103. American Society of Mechanical Engineers, 2012.

[Roumeliotis and Tselikas, 2023] Konstantinos I. Roumeliotis and Nikolaos D. Tselikas. Chatgpt and open-ai models: A preliminary review. *Future Internet*, 15(6), 2023.

Building Domain Theories for Commonsense Reasoning from Language-Grounded Ontologies

Kenneth D. Forbus
Qualitative Reasoning Group, Computer Science Department, Northwestern University
2113 Tech Drive, Evanston, IL, 60208, USA
forbus@northwestern.edu

Abstract

One of the signature properties of commonsense reasoning is its breadth. Qualitative domain theories have been successfully built both by hand and by learning for small sets of phenomena, but scaling remains an issue. This paper describes an approach to achieving breadth by leveraging a large commonsense ontology. The idea is that a small set of concepts in the ontology corresponding to continuous processes and event types are identified, called *anchor concepts*. The subclasses of these anchor concepts form specializations of processes and types of events which then provide the desired breadth, e.g. that snowboarding is a form of motion. Pre-existing role relations for concepts of events and processes provide information about participants for QP model fragments and encapsulated histories. We show how this approach produces partial information about a broad range of continuous processes and event types. Rather than the usual carefully curated and bounded domain theories used in QR for modeling scientific and engineering reasoning, this approach to building domain theories is more open-textured. For example, the surface over which snowboarding occurs is usually snow and/or ice, something not currently stated in the ontology. The idea is that the rest of the specifications for any particular subclass of process will need to be completed by other means, e.g. instruction, experimentation, or hand-engineering.

Introduction

One of the original motivations for qualitative reasoning was to support commonsense reasoning. Even when the focus of QR is scientific or engineering reasoning, one of the jobs of qualitative models is to help in model formulation. Model formulation involves mapping from the unruly open everyday world to the tightly constrained formalisms often used in professional reasoning. For AI systems to be as helpful as a person in model formulation, they must have a reasonable understanding of the everyday world. Most qualitative domain theories have been generated by hand. Hand generation has been effective for many aspects of professional knowledge (e.g. aspects of physics, engineering thermodynamics, chemical engineering), but is daunting when considering the range of everyday phenomena. After all, people – with sensorimotor systems and learning abilities that are far more data-efficient than today's ML – take a decade or two to achieve broad commonsense knowledge, gleaned from a combination of direct experience and cultural inputs, including direct instruction. Progress has been made on building out knowledge bases using learning by reading, but most approaches require simplified text. Large language models should be useful in helping to expand the range of texts that can be processed. However, LLMs make poor knowledge bases for two reasons. First, their exposure to language is not grounded in the everyday world. Second, their success criterion is generating statistically plausible text, not correct reasoning. As the confabulation problems with LLMs show, these are at best only correlated. Hence we, like many others, continue to focus on using knowledge graphs as knowledge bases. Fortunately, there are now multiple large knowledge graphs such as Wikidata (Vrandecic & Krotzsch, 2014) that can provide broad knowledge (Forbus & Demel, 2022).

One of our hypotheses is that qualitative process theory captures aspects of natural language semantics (Forbus, 2019). One consequence of this hypothesis is that the underlying ontology in a commonsense knowledge base should in part reflect representational concerns relevant to qualitative reasoning. The everyday world includes many patterns of behavior that we can think of in terms of spatio-temporal units, the idea of *histories* introduced by Hayes (1985). Histories for objects are temporally extended but spatially bounded[1]. Histories are often defined in terms of the kinds of behavior happening in them. For example, one can think of filling a coffee cup or a swimming pool (or a basement). Filling can be accomplished by pouring

[1] By contrast, in the situation calculus, each situation is indeterminant temporally but spatially unbounded, which is a source of the frame problem.

from a pot in the case of a coffee cup, or pouring from buckets or a hose in the case of a swimming pool. These episodes are often delimited by qualitative changes in properties, e.g. for filling, the amount of fluid in the container being filled should be increasing during that episode. Histories can be hierarchical, e.g. filling a swimming pool using buckets will involve many filling/emptying of buckets, the emptyings of which all contribute to the filling of the swimming pool. The ability of qualitative representations to help segment perceptual information suggests that an important component of human commonsense knowledge is a broad vocabulary of descriptions of such types of events. Such events play a role in professional reasoning, since analyses are often couched in terms of them. Determining when to fire retro-rockets in a Mars lander, for example, requires conceptualizing the relevant part of its motions as a descent involving gravity, and solving for a firing time that will enable the lander to touch down safely. Event descriptions help provide boundary conditions, like the landing site and the desired speed on landing. *Encapsulated histories* in QP theory have been used to provide qualitative and quantitative models for such events that can be used in professional reasoning (e.g. Klenk & Forbus, 2009). Encapsulated histories can be learned via analogical generalization over descriptions of behaviors (Friedman & Forbus, 2008;2009). However, this has only been done for a small number of types of events.

Histories describe what is happening, but they do not explain why it is happening. QP theory introduced a notion of *continuous process* that provides a model for causal mechanisms in continuous domains. Pouring and filling in the examples above, for instance, would be explained in terms of a liquid flow process. The effects of such processes are compositional, so that models for specific systems can be formulated by combining them. Consider for example pouring water into a leaky bucket. There is a flow of water in, and a flow of water out – the intended flow and the leak are both explained in terms of the same type of continuous process. But whether or not the bucket is filling or not depends on the relative rates of the two flows. Thus the flows explain the filling episode. The everyday world contains many kinds of phenomena that we think of as continuous processes, such as motion, flows, phase changes, and so on. These general processes manifest in many ways. For example, motion can involve projectile motion through the air or empty space, moving along a surface, or various forms of water falling from the sky (e.g. rain, snow, hail). Hand-engineering model fragments for the full range of processes that manifest in our everyday world from scratch is daunting.

How can we leverage a broad commonsense ontology to build a commonsense QR domain theory? (Or, alternately, how to we bring the fruits of QR into efforts to ontologize commonsense knowledge?) Suppose we can identify within an ontology a set of high-level event types and processes that can serve as *anchor concepts* for a QP domain theory. That is, an anchor concept inherits from the concept of a type of encapsulated history or continuous process expressed in QP theory (e.g. motion), such that all of its more specialized concepts are aptly characterized by that domain theory construct. This provides a way of using the broad ontology to leverage well-engineered domain theory components. Moreover, if the ontology has mappings to natural language, then that ontology can be used in communicating with human partners, another requirement to achieve human-like model formulation.

This paper reports on work in progress exploring the use of a broad commonsense ontology to build a QP domain theory for commonsense reasoning. We start by summarizing the relevant background: aspects of QP theory and the NextKB knowledge base we are using. Then we discuss the issues involved in integrating QP theory with a broader domain theory, including processes versus events and continuous versus discrete levels of representations. A mapping of a small QP domain theory to NextKB is described next, demonstrating that this approach enables the range of phenomena that can be discussed to be considerably magnified. Finally, we discuss conclusions and future work.

Background

Qualitative process theory postulates *continuous processes* as the mechanisms for change in systems governed by continuous parameters. This model breaks down in some domains, e.g. analog electronics is better modeled by a component-centered ontology (de Kleer, 1984), and does not capture many of the spatial properties of motion (Forbus et al. 1991). Nevertheless, it appears applicable to a broad range of everyday phenomena. Recall that a QP domain theory consists of a set of schema, called *model fragments*, which can be instantiated to assemble models for particular scenarios and systems. Model fragments are specified by *participants* which indicate the kinds of entities it can be instantiated on, *conditions* which indicate when an instance of that model fragment is active, and *consequences* which are statements that hold for any time in which the conditions are true. Continuous processes are a subclass of model fragment that have *direct influences*, i.e. partial specifications of the derivative of some quantities of its participants, such that making a closed world assumption over the set of instantiated continuous processes specifies (qualitatively) the derivatives of those parameters.

As noted above, the consequences of processes hold at every instant within an interval over which that process is acting. To describe the cumulative effects of such processes requires histories for the objects affected, as per our example of filling a bucket earlier. To provide causal and mathematical constraints on episodes of histories, QP theory also provide a formalism for *encapsulated histories*, which can reference the temporal and spatial aspects of the episode they describe. These schemas are applied like model fragments, in that they have participants, conditions, and consequences. The consequences can be qualitative, e.g. the distance travelled in an episode of motion is qualitatively proportional to the time travelled. The consequences can also be quantitative, e.g. an equation describing distance travelled as a function of initial velocity and constant acceleration.

QP theory can be formalized in a variety of ways. Here we use an implementation grounded in the NextKB knowledge base, which is summarized below. This implementation has been used in several previous experiments and its details are not relevant for understanding this paper.

The NextKB knowledge base is an open-license resource being built at Northwestern University to support research in knowledge-rich AI and cognitive science. It builds on Cycorp's OpenCyc ontology, which provides a massive set of formally represented concepts and relationships. OpenCyc is an open-license subset of the Cyc ontology. Concepts are formally represented by *collections*, which can intuitively be considered as sets. For example, the collection Container represents all of the containers that there are, have been, will be, or might be. The relationship isa indicates that an individual can be considered an instance of that concept, e.g. (isa KenCollegeMug Container). There are inheritance relationships between concepts. The genls relation indicates inheritance between collections, e.g. (genls LiquidStorageTank Container) indicates that things which are storage tanks for liquids are also containers. There are also inheritance relationships among predicates, e.g. (genlPreds containerEntered toLocation) indicates that containerEntered implies toLocation holds between its arguments. The OpenCyc ontology is more expressive than most. For example, type-level predicates enable it to express higher-order statements, and modal operators (e.g. knows, beliefs) are included. This makes formalizing many concepts substantially easier than less-expressive ontologies. For example, (behaviorIncapable P1 SolvingAProblem thingAnalyzed) indicates that the problem P1 cannot be solved. There are many consistency constraints in the ontology. For example, disjointWith indicates that an instance of one collection cannot be a member of the other, e.g. (disjointWith Herbivore Carnivore). There are type constraints on arguments, arity, and the range of logical functions.

Some form of context mechanism is crucial for any representation system capable of considering alternative qualitative states, alternate perspectives in modeling (e.g. Falkenhainer & Forbus, 1991), or alternate domain theories. OpenCyc uses *microtheories* to provide a mechanism for contexts. Every fact holds in one or more microtheories. Microtheories inherit from each other via the genlMt relation. For example, (genlMt HumanSocialLifeMt HumanActivitiesMt) indicates that every fact believed in HumanActivitiesMt is also believed in HumanSocialLifeMt. Inheritance in all cases is monotonic. There are non-monotonic predicates to express dependence of some conclusions on the epistemic state of the system, e.g. believing something because one cannot infer its negation is a strategy that can be expressed and localized, rather than "wiring in" negation by failure as a global policy.

We distilled NextKB's ontology from the four available versions of OpenCyc. NextKB[2] includes over 82,000 collections, 26,000 relationships, 5,000 logical functions and 700,000 facts. We note that this is a small subset of the Cyc ontology, as found in the commercial version of Cyc and in ResearchCyc, both of which also have massively more axioms constraining the concepts and relationships in the ontology as well as a powerful reasoning engine that supports useful commonsense inferences complete with explanations based on dependency traces. For example, ResearchCyc can conclude that Earth cannot run a marathon, because no inanimate object can. We used the ResearchCyc knowledge base productively for a long time, but finally switched to OpenCyc to support dissemination and replication of our work.

In addition to OpenCyc contents, NextKB contains extensions for qualitative reasoning, including both QP theory and qualitative spatial reasoning, as well as visual/spatial capabilities used in CogSketch, our high-level vision system and sketch understanding system (Forbus et al. 2011; Forbus & Lovett 2021). Reasoning in these extensions is often conducted via procedural attachments to predicates, for efficiency. Analogical reasoning and learning is handled similarly. NextKB also has substantial natural language resources for English. It has a large lexicon, derived in part from a public-domain version of Webster's dictionary. Its semantics are organized using FrameNet frames, which have been mapped by hand to concepts in the OpenCyc ontology. FrameNet thus serves as a bridge between words and OpenCyc concepts. The lexicon has over 190,000 words and over 69,000 semantic translations. As noted above, AI assistants that help in model formulation need such broad language coverage, in order to communicate with their human partners.

[2] https://www.qrg.northwestern.edu/nextkb/index.html

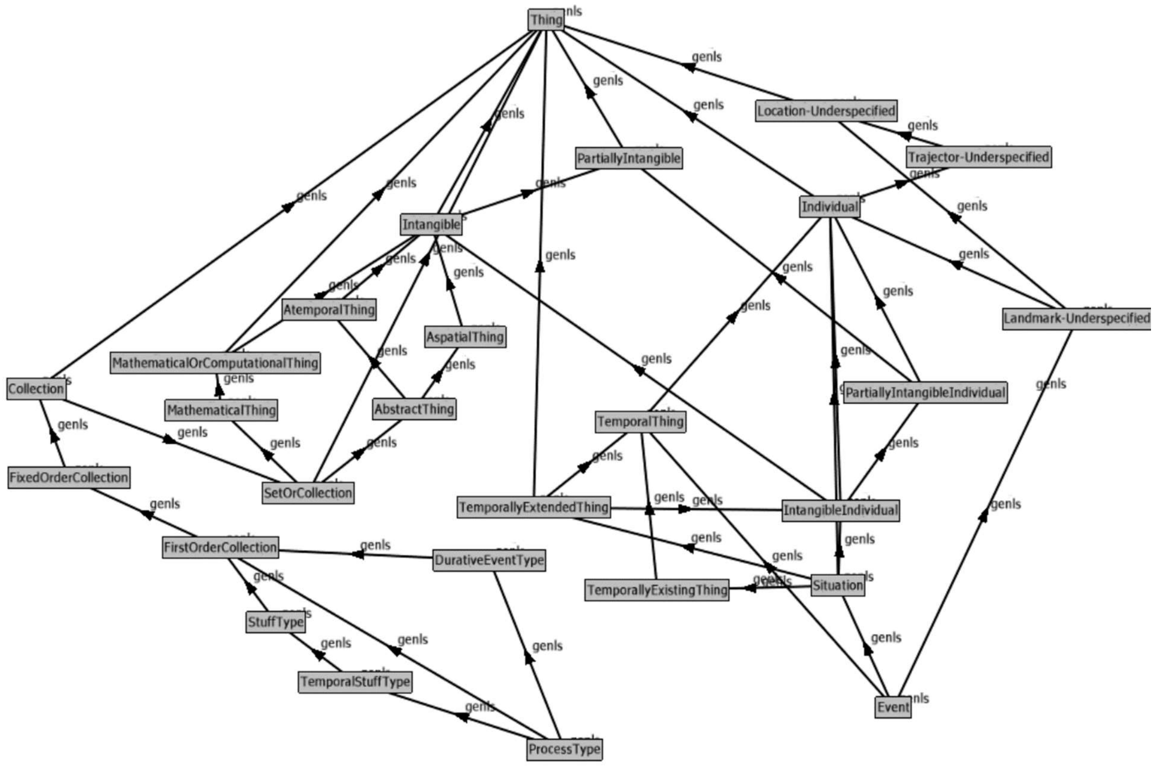


Figure 1: Partial view of the OpenCyc upper ontology showing where events and process types are grounded. It contains over 82,000 such concepts.

Ontological Grounding for Processes

All commonsense ontologies include some notion of event. Figure 1 shows how the general concepts of events and types of processes are related in the OpenCyc ontology. Generally there is a notion of sub-events, e.g. a wedding ceremony might include guests arriving, the exchange of vows, and merry-making. Processes are often represented in a similar way, with the difference being that the same properties are true of all of the sub-intervals within an occurrence of that process. This is compatible with the QP theory notion of a process being active whenever its conditions hold. Whether or not a phenomenon is treated as continuous or discrete depends on the granularity used in its description. A robust commonsense ontology must be able to support multiple levels of granularity, and OpenCyc does a reasonable job of this. For example, OpenCyc treats walking as a process, which is useful for estimating things like distance covered and effort expended. But it also provides support for describing the particular movements of legs up and down, discrete events within walking that are useful for purposes of physical therapy, for example. Another example is OpenCyc's concept of PrecipitationProcess, which is viewed as continuous, even though at a finer granularity, the movement of each raindrop or piece of hail can be viewed as a discrete event. Prior qualitative reasoning research has intermingled continuous and discrete perspectives in a similar way. For example, Rickel & Porter (1994) used time-scales in multiple perspective modeling of biological phenomena, given a particular time-scale of interest, their domain theory treated slower phenomena as exogenous constraints and quicker phenomena as functional connections.

Concepts describing processes form natural anchor points for QP-style continuous processes. That is, QP-style continuous processes are formalized as collections, and existing elements in the ontology inherit from them, thereby inheriting their schema. Figure 2 illustrates. However, not all commonsense processes are aptly described as continuous processes in the QP theory sense. For example, the concept of ProcessType in OpenCyc combines TemporalStuffType (thereby capturing the idea that the subintervals are the same) and DurativeEventType (thereby capturing the idea that occurrences of processes take time) and has 654 instances. Some of these are nicely expressed by QP theory, such as FluidFlow-Translation and PrecipitationProcess. Many others are not, including InternetSearching and IgnoringSomething. The difference is whether there are continuous parameters that aptly characterize the changes within an occurrence of a process. Uniformity in subintervals does not necessarily imply the existence of such parameters. Sometimes there are metaphorical extensions that can be applied. For example, an Internet search might be characterized in terms of progress towards the information-seeking goals for that search, or a decision-maker's thinking reaching a level of certainty about an action they are contemplating. We will not consider such metaphorical examples further here, but return to them in proposed future work below.

Linking QP continuous processes and encapsulated histories also requires linking the relationships that specify the participants for a model fragment. In English, for example, the subject of a motion verb indicates the object that is moving. The NextKB resources provide objectMoving as a relationship which formalizes this notion, enabling NLU systems to propose it as a possible meaning. Other spatial prepositions capture properties of an episode of motion. The spatial prepositions "from" and "to" can indicate the start and end of a motion, with "along" or "via" indicating its path. For example, From-TheWord has semantic translations that includes startOfPath (a spatial interpretation), intervalStartedBy (a temporal reading), and from-Generic (a more abstract version that includes the other two, but also the giver of a gift).

Analysis: Anchor Concepts in OpenCyc

To explore these ideas, we used pre-existing model fragments and encapsulated histories from QP domain theories for exploring the roles of qualitative reasoning in elementary school science tests (Crouse & Forbus, 2016), for learning textbook problem solving via cross-domain analogies (Klenk & Forbus, 2013), and some classic QP domain theories (Forbus, 1984). The goal is to estimate two properties: (1) How much leverage does the ontology provide us, in terms of additional phenomena covered? (2) Do the anchor

```
(in-microtheory PrecipitationQPMt)

(genlMt PrecipitationQPMt ScienceTestCollectorQPMt)
(genlMt ScienceTestInferenceQPMt PrecipitationQPMt)

;; model fragment definition
(isa NaivePrecipitationProcess QPProcessType)
(comment NaivePrecipitationProcess
  "Precipitation occurs when a liquid is in exposed to the air and its temperature is less than boil-
ing point but greater than its freezing point. The result of the process is that the liquid vaporizes
into an atmosphere.")

(mfTypeParticipant NaivePrecipitationProcess ?liquid LiquidTangibleThing liquidOf)
(mfTypeParticipant NaivePrecipitationProcess ?sub ChemicalCompoundTypeByChemicalSpecies substanceOf)
(mfTypeParticipant NaivePrecipitationProcess ?atmosphere GaseousTangibleThing atmosphereOf)
(mfTypeParticipantConstraint NaivePrecipitationProcess (substanceOfType ?liquid ?sub))
(mfiReverseConsequenceOf NaivePrecipitationProcess (and (isa ?rain RainProcess)
                                                        (products ?rain ?liquid)))
(mfTypeCondition NaivePrecipitationProcess (qGreaterThan
                                            (AmountOfFn ?sub Liquid-StateOfMatter ?atmosphere)
                                            SaturationPoint))
(mfTypeBiconditionalConsequence NaivePrecipitationProcess (hasQuantity ?self
                                                            (PrecipitationRateFn ?self)))
(mfTypeConsequence NaivePrecipitationProcess (qprop (PrecipitationRateFn ?self)
                                                    ((QPQuantityFn Temperature) ?liquid)))
(mfTypeConsequence NaivePrecipitationProcess (i+ (AmountOfFn ?sub Liquid-StateOfMatter ?liquid)
                                                 (PrecipitationRateFn ?self)))
(mfTypeConsequence NaivePrecipitationProcess (i- (AmountOfFn ?sub Gaseous-StateOfMatter ?atmosphere)
                                                 (PrecipitationRateFn ?self)))

;;; Anchor process
(genls PrecipitationProcess NaivePrecipitationProcess)
```

Figure 2: Example of a QP-style process anchored to the OpenCyc ontology

concepts provide connections to language that can be exploited by cognitive systems? To estimate leverage, we examine the subclasses of the anchor concepts. How many are there, and are they all reasonable? To estimate language coverage, we count the number of lexical items connected to the conceptual space covered by the anchor concept.

Table 1 shows the results for number of subclasses and words for reasonable anchor concepts for a set of pre-existing model fragments[3]. The anchor concepts were chosen to maximize applicability of the model fragment to the subclasses. This was straightforward for a number of model fragments, in particular, the basic processes involving fluids, heat, and phase changes. For example, the subclasses of liquid flow include DrinkingEvent and hence the words "drink", "imbibe", "quaff", "slurp" and "swill", among others. For heat flow, the subclasses include various forms of cooking (baking, barbecuing, steaming, roasting, and grilling). Not everything in the ontology is commonsense, e.g. the subclasses here include some ways that heating is used in semiconductor manufacturing, as well as global warming. This ability to expand to incorporate professional knowledge is a major advantage of starting with a broad ontology, and should simplify model formulation.

There are cases where the model fragments are somewhat too specific compared to the anchor process. Precipitation is an example: The model fragment concerns liquid leaving the atmosphere (as shown in Figure 2), whereas the PrecipitationProcess includes HailStormProcess, where what comes from the sky is ice. This could be resolved either by choosing a more specific subclass (e.g. RainProcess) or by slightly generalizing the model fragment. This issue comes up most strongly in motion, where there are general properties that hold (e.g. an episode in a motion history has a start, end, and velocity – motion that returns to its starting point is included) but also additional complications due to particular conditions, such as friction when sliding or gravity for projectiles. This has suggested ways to refactor our QP domain theories, i.e. to introduce encapsulated histories using purely qualitative mathematics for very abstract concepts of processes, to better capture the commonsense inferences that they license.

Motion is especially prolific. The 355 subclasses include things like snowboarding, flying by flapping wings, and parkour in addition to more traditional concerns of QR like projectile motion and sliding. It should be noted that in the ontology, PreciptationProcess entails motion, hence the words for that process and its specializations are a (small) subset of the words that refer to types of motion. Some of these subclasses have additional entailments over the basic QP model of motion, e.g. sliding entails the possibility of friction, and flying by flapping wings entails the use of energy supplied by the organism/artifact locomoting that way, as do walking and running. These additional distinctions could be captured by model fragments that elaborate motion, anchored to those concepts. For example, Flying-FlappingWings is a subclass of LocomotionProcess-Animal, so the common need for energy to accomplish locomotion, by whatever means, can be expressed once anchored on LocomotionProcess-Animal and also inherited.

We note that anchor concepts for some of the categories used in the participant constraints for model fragments are easily found, but others are not. An easy case is the general concept of container. The concept as used in these model fragments is reasonably captured by the collection Container, which has 2,787 subclasses and 1,757 words, although it includes many subclasses that someone might not usually think of in this way, e.g. dance clubs, airplane cabins, and a gigantic list of types of cars.

By contrast, it is difficult to find an anchor concept for the general concept of physical object (Physob, in classic QP domain theories). The closest is PartiallyTangible, which includes 42,339 subclasses, including things like butterflies and stores, but also concepts that are poor fits, such as the space under coffee tables. Similarly, concepts like thermal or volumetric objects, regularly used in compositional modeling for engineering domains, are not distinctions that the

Phenomena	**Model Fragment Type**	**Anchor**	**Sub-classes**	**# Words**
Liquid flow	LiquidFlowProcess	LiquidFlowEvent	26	15
Heat flow	HeatFlowProcess	HeatingProcess	44	50
Boiling	BoilingProcess	Boiling	5	3
Evaporation	Evaporation	Evaporation	0	1
Precipitation	NaivePreciptiationProcess	PreciptationProcess	14	21
Floating	ObjectFloatingInFluid	FloatingInASubstance	34	14
Motion	Motion	Movement-TranslationProcess	355	170
Friction	FrictionBetweenSolids	FrictionProcess	21	22

Table 1: Anchoring QP model fragments in NextKB

[3] We do not describe anchoring encapsulated histories to the OpenCyc ontology because our existing encapsulated histories, being developed later, were already integrated with OpenCyc because it is a subset of ResearchCyc.

OpenCyc ontology designers were concerned with. For such cases, it is straightforward to add the desired concepts to the ontology and incorporate subclasses of PartiallyTangible as appropriate. Moreover, such decisions can be incrementally learned from examples (Klenk et al. 2008).

So far we have looked at how much language coverage is added by anchoring QP constructs into the OpenCyc ontology. Are there words that are relevant to QP constructs that are not covered by anchor concepts? Yes. The exact number is hard to calculate, since it requires examining all of the lexicon. But, for example, the word "flow" uses FluidFlow-Translation, which includes both liquid and gas flow as subclasses. The QP models could be re-factored into a general fluid flow process with model fragments for liquids and gasses being model fragments specializing that one, or a system seeking to construct a qualitative model from a natural language description could gather candidate model fragments from subclasses of the mapped concept.

Conclusions and Future Work

The breadth of commonsense is a daunting challenge for qualitative reasoning. This paper argues that using a large-scale commonsense ontology (OpenCyc) that is tied to language (via NextKB) can help provide such breadth. The ability to find anchor concepts for model fragments and encapsulated histories from previous efforts is encouraging. The broad convergence in conceptual structure which makes FrameNet and OpenCyc mappable in the first place suggests that these commonalities are likely to be found in other resources, informed by the same cultural constraints. How this would vary given different cultures is a fascinating question. For example, how information is packaged into verbs varies across languages. In English one might say "The bottle floated into the cave" but in Spanish one would say the equivalent of "The bottle entered the cave, floating." Will those differences lead to cross-cultural differences in qualitative models?

We plan three lines of future work. First, we plan to refactor the QP model fragments and encapsulated histories to provide some of the intermediate representations that are currently missing, as well as use the ontology to help determine gaps where additional coverage is needed. Second, we plan to use this augmented domain theory to explore the construction of high-precision mental models during learning by reading, in order to learn new domain theory constructs and to solve problems expressed via language and sketching. Third, we plan to examine whether extending QP domain theories to more metaphorical uses supports inferences consistent with human metaphors (Lakoff & Johnson, 1981).

Acknowledgments

This research was supported by the Machine Learning, Reasoning, and Intelligence Program of the US Office of Naval Research, Grant N00014-23-1-2294.

References

Crouse, M. and Forbus, K. (2016). Elementary School Science as a Cognitive System Domain: How Much Qualitative Reasoning is Required? In *Proceedings of the Fourth Annual Conference on Advances in Cognitive Systems*. Evanston, IL.

de Kleer, J. (1984) How Circuits Work. *Artificial Intelligence*, **24**:205-280.

Falkenhainer, B. and Forbus, K. (1991). Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51, 95-143.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24, 85-168.

Forbus, K. (2019). *Qualitative Representations: How People Reason and Learn about the Continuous World*, MIT Press.

Forbus, K. & Demel, W. (2022). Integrating QR Quantity Representations with the Semantic Web. *Proceedings of QR 2022.*

Forbus, K., Nielsen, P., & Faltings, B. (1991). Qualitative Spatial Reasoning: The CLOCK project. *Artificial Intelligence*, **51**:417-471.

Forbus, K. & Lovett, A. (2021). Same/Different in Visual Reasoning. *Current Opinion in Behavioral Sciences*, 37:63-68, https://doi.org/10.1016/j.cobeha.2020.09.008

Forbus, K., Usher, J., Lovett, A., Lockwood, K., and Wetzel, J. (2011). CogSketch: Sketch understanding for Cognitive Science Research and for Education. *Topics in Cognitive Science*, 3(4), pp 648-666.

Friedman, S. and Forbus, K. (2008). Learning Causal Models via Progressive Alignment and Qualitative Modeling: A Simulation. *Proceedings of the 30th Annual Conference of the Cognitive Science Society (CogSci).* Washington, D.C.

Friedman, S. and Forbus, K. (2009). Learning Naïve Physics Models and Misconceptions. *Proceedings of the 31st Annual Conference of the Cognitive Science Society.* Amsterdam, Netherlands.

Hayes, P. (1985). Naïve Physics I: ontology for liquids. In J.R. Hobbs & R. Moore (Eds.), *Formal Theories of the Commonsense World*. Ablex: Norwood, NJ.

Klenk, M., Friedman, S., and Forbus, K. (2008). Learning Modeling Abstractions via Generalization. Proceedings of the 22nd International Workshop on Qualitative Reasoning. Boulder, CO.

Klenk, M. & Forbus, K. (2009). Analogical Model Formulation for AP Physics Problems. *Artificial Intelligence*, 173(18), 1615-1638. doi:10.1016/artint.2009.09.003

Lakoff, G., & Johnson, M. (1981) *Metaphors we live by*. University of Chicago Press.

Rickel, J. & Porter, B. (1994). Automated Modeling for Answering Prediction Questions: Selecting the Time Scale and System Boundary. *Proceedings of AAAI94*.

Vrandecic, D. & Krotzsch, M. (2014). Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM*, 57(10):78-85, October, doi:10.1145/2629489.

Proposal for a Project on Knowledge-based Decision Support for Water Treatment

Peter Struss
Technical University of Munich
struss@in.tum.de

Abstract

The proposal is related to Goal 6 of the SDGs, “Clean water and sanitation”. The general goal of the proposal is promoting the establishment of facilities for water treatment, improving their scientific and technical foundations, and providing education and advice to local operators of plants, which might be non-experts. This is meant to be achieved by a web-based decision support system (DSS) that contains a repository of formal representations of treatment technologies and relevant natural processes and, based on them, an environment that supports different tasks, such as the design and operation of treatment systems.

Problem Addressed: Access to Drinking Water

In its resolution 70/1 “*Transforming our world: the 2030 Agenda for Sustainable Development*” [UN70/1, 2015], the UN general assembly committed to “*the human right to safe drinking water*” and established as part of Goal 6 “*Clean water and sanitation*”: “**By 2030, achieve universal and equitable access to safe and affordable drinking water for all**”. Here, “all” means 100 % of the people living on this planet.

In 2018, the general assembly emphasized in its resolution 73/226 [UN73/226, 2018] “*that water is critical for sustainable development and the eradication of poverty and hunger*”, but had to note *“that the world is not on track to achieve water-related Sustainable Development Goals and targets at the global level by 2030 at the current rate of progress*”.

The Sustainable Development Goals Report 2022 [SDG report, 2022] reviews the progress achieved in the 2030 Agenda. Regarding Goal 6, it reports: “*The proportion of the global population using safely managed drinking water services increased from 70 per cent in 2015 to 74 per cent in 2020. Still, 2 billion people were without such services that year, including 1.2 billion people lacking even a basic level of service. ... At the current rate of progress, the world will reach 81 per cent coverage by 2030, missing the target and leaving 1.6 billion people without safely managed drinking water supplies*” (p. 38). It concludes that “*To reach universal coverage by 2030, current rates of progress would need to increase fourfold*”, and that “*Achieving these targets would save 829,000 lives annually.*”

Not surprisingly, suffering from this situation is not evenly distributed over the planet. The report states that “*Eight out of 10 people who lack even basic drinking water service live in rural areas, and about half of them live in LDCs.*” (LDC: Least developed countries) – a conflict with the *Leaving No One Behind* (NLOB) action framework which declares *Equality and Non-Discrimination at the Heart of Sustainable Development* [LNOB, 2016].

As a consequence, improving the situation and speeding up the progress towards the 2030 goal has to focus on rural areas, exp. in the LDCs. Reaching the goal requires a number of actions, such as regulations and technological solutions that help to prevent pollution, improve water harvesting, reduce excessive freshwater withdrawal, water-use efficiency, and establish a nexus of water, energy, and food production. The problem is not simply access to a sufficient quantity of water, but to **safe** drinking water (or water for other purposes, such as irrigation), **facilities for water treatment** are needed. Esp. in rural areas, treatment facilities have to be distributed and run locally to avoid problems in transporting water over long distances.

An obstacle to establishing a larger number of treatment plants in places where they are most urgently needed is, besides the lack of financial resources, that in LDCs and esp. their rural areas, there may be a lack of expertise in designing, building and operating such plants. Even though there may be some standard technology available, there could be a need for adaptation to specific local conditions. Also, when facing disturbances of the plant operation, less experienced operators may need support.

In line with the LNOB policy “*Cooperate in technology transfer to promote greater equality*”, our proposal is to develop an intelligent decision support system (DSS) ([Dhar-Stein, 1997], [Sanchez-Marre, 2022]). Such systems have been built for several domains, including water treatment ([Poch et al., 2012], [Mannina et al., 2019]). Our proposal aims at making technological knowledge and scientific results more accessible, improving the transfer of experience and best practices to other locations, and providing problem solving algorithms that support or automate the performance of various tasks during the life cycle of water treatment facilities.

Targeted AI Contributions

The general goal of the proposal is promoting the establishment of facilities for water treatment, improving their scientific and technical foundations, and providing education and advice to local operators of plants, which might be non-experts. This is meant to be achieved by a web-based decision support system (DSS) that contains

- a repository of formal representations of treatment technologies and relevant natural processes and, based on them,
- an environment that supports different tasks, such as the design and operation of treatment systems.

In contrast to other decision support systems in the area of drinking or waste water treatment that provide support to controlling and troubleshooting special (standard) kinds of treatment plants (activated sludge, constructed wetlands, …)(see [Poch et al., 2012], [Mannina et al., 2019]), our aim is to cover a wide range of combinations of different technologies and process steps and, in particular, to support the task of configuring solutions tailored to particular conditions and requirements, rather than taking the plant structure as given.

The Knowledge Repository

In the proposed project, we take a model-based approach ([Heller-Struss, 2002], [Wotawa et al., 2010]): expert knowledge about the water treatment domain is not represented in terms of verbal descriptions, data charts etc. but in the form of models, i.e. executable formal expressions. These models are not describing complete treatment systems, but, in a reductionist way, individual process steps in a context-free manner, stating their preconditions and inputs and their outcome, i.e. some cause-effect relation. This way, such model fragments can be assembled (automatically or manually) to form a plant model. In addition, the repository has to comprise models of the natural (physical, chemical, biological) phenomena that occur and have an impact on the performance of the systems, including ones that might disturb or prevent the proper operation of a plant. Finally, descriptions of possible human interventions (such as changing the amount of added substances) can be part of the repository.
In the project, we build on a previously developed theory and prototype ([Heller-Struss, 2002], [Roque et al., 2003]) which adopted the approach of process-oriented modeling [Forbus, 1984]. The model fragments ("**process types**") in the repository are considered to be the elementary phenomena in the domain, in particular, treatment steps and natural processes that may occur intentionally or due to abnormal conditions in the plant. A process is represented as a pair of conditions and effects, which both contain assertions about structural aspects, i.e. existing objects and their relations (such as particles of a certain kind contained in the water), and about resulting restrictions on quantities associated with the objects (e.g. the concentration of a substance is reduced to zero). Turning the informal semantics of a process, namely that the effects will be established whenever the preconditions are satisfied, into logic, a process becomes an implication:

$$\text{StructuralConditions} \wedge \text{QuantityConditions} \Rightarrow \text{StructuralEffects} \wedge \text{QuantityEffects},$$

QuantityEffects can contain special expressions, called influences, that capture the impact of a process on the dynamics of the systems, i.e. how quantities change, but, nevertheless, are beyond the expressiveness of differential equations. In an approximate way, influences specify a partial derivative of a quantity. The actual change of a quantity can only be determined when all influences on it have been determined (which involves a closed world assumption; see [Heller, 2001].
for details).
Assembling a model of a system from instantiated process types in the repository requires that their representation uses a particular ontology, which is the second ingredient of the repository. Otherwise, expressions in effects and conditions could not be matched, e.g. to detect that one process triggers another one or that several processes affect the same quantity. This ontology has to introduce types of objects, their characterizing quantities along with the respective domains and types of relations between objects, specifying their signature in terms of object types and their properties, e.g. being symmetric.

Example

In the water treatment domain, the involved **types of objects** include

- water containers, basins etc.
- devices, such as valves, pumps, and mixing devices
- water bodies: inflow/outflow, water in containers
- ingredients of the water, like organic matter, dissolved substances, pollutants
- substances added during the process (oxidation agents, coagulants, …).

The **relations** are mainly needed to express

- connectivity of containers/water bodies
- component connections
- containment in water bodies (suspended_in, dissolved_in).

Typical **types of quantities** involved in the description of conditions and effects are

- attributes of water bodies (pH, temperature, , …)
- attributes associated with relations, mainly concentration specifying a containment relation.

As a side note, a design decision has to be taken whether to represent the water ingredients explicitly as objects and tie their concentration to the containment relation or to represent the various concentrations just as quantities associated with water bodies (refer to the challenges section of this paper).
The various kinds of process steps fulfill mainly the task of removing particular unwanted elements from the water or modifying them, usually in a sequential manner as depicted in Figure 1.
Process-oriented models of the steps have to capture the transformation of the water, relating the types of input properties with those of the output. Since conditions and effects of different processes refer to the same features of water, they can capture the treatment by the entire plant.

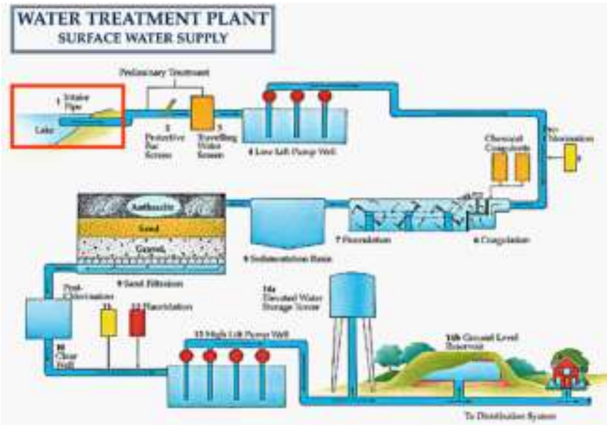


Figure 1 A typical treatment plant (Source:Drewes, Lecture Notes “Advanced Water Treatment Engineering and Reuse”, TUM 2021)

To illustrate the above, we consider the removal of colloidal particles (with a size between 0.001 mm and 0.01 mm), which can be carried out by the sequential steps of coagulation, flocculation, and sedimentation (refer to Figure 1). (Also, larger parts, up to 0,1 mm, may be treated here).
In the first step, coagulants (e.g. ferric sulfate or aluminum chloride) are added with the effect of neutralizing the charges of the particles. Thus, repelling forces between them are eliminated which enables the step of flocculation: under the influence of mixing devices (which have to be run with an appropriate speed) the discharged particles collide and aggregate to form larger and heavier flocs, which, in the sedimentation step, sink to the ground and are, thus, extracted from the flow of water.
In a simplified description of the process types, the preconditions of the **coagulation process** include the incoming water body and the contained colloidal particles with a particular concentration (zero or positive) and the added coagulant, while the effects specify (ideally) a zero concentration of (charged) colloidal particles in the outflow and uncharged particles contained with a concentration equaling the concentration of the incoming colloidal particles. Of course, all other objects contained in the water inflow will remain unaffected and simply transported to the output. Implementing this trivial, but essential feature turns out to be an instance of the infamous frame problem and is actually a challenging task, as discussed in the respective section.
The effects of the **flocculation process** include a zero concentration of discharged particles and flocs, whose concentration (qualitatively) equals the concentration of the incoming particles, with properly working mixers also in the precondition. Note, if (mis)behavior of involved devices, such as the mixers in flocculation (or their power supply and so on) are to be considered, e.g. in trouble shooting, we need to embed behavior models of components in the process-oriented modeling paradigm (again, refer to the challenges section).

A **sedimentation process** has larger particles (including e.g. clay, silt, etc., but also flocs) in its input, and the effects specify that the concentration of particles with a higher specific weight compared to water in the output will be zero, while the amount of the sediment is increased or stable (which may by modified by a removal process). Particles with a lower specific weight will just be moved from input to output. This context-independent representation of the process allows us to use it in a flexible way. For instance, in practice it is also used before the coagulation step.
This way, the repository contains elements whose combination yields an executable model. It differs from other simulation systems, because it potentially expands its structure by including process instances that are entailed by others.
It forms a firm theoretical and technical basis for various task-specific tools which support problem solving with different degree of automation, as outlined in the following sections.

Plant Design

There is a well-established set of treatment steps and a fairly standard mainly linear arrangement of these steps to form a treatment plant. Its individual treatment steps are captured as process types in the repository. In addition, there other types of treatment systems (e.g. constructed wetlands, delivering purified, but non-potable water) and more advanced technologies, such as membrane processes. For a particular area and application, designing a proper system means deriving a selection and arrangement of process steps that reflect the specific characterization of the incoming water and the operational conditions, as well as a set of requirements on the quality of the output water.
In our solution, this means finding a combination of elements from the repository that transforms the input into the output. Based on the cause-effect representation of the process types in the repository, the DSS can assist manual design by a human in offering candidate processes whose effects imply (some of) the output requirements.
When given the structure of a designed system, S, and a specification of the input and the contextual conditions (such as ambient temperature), INPUT, the DSS can create a system model MODEL(S, INPUT) as a collection of processes. Note that it has to be “causally complete” in the sense that it does not only contain the intended process steps of S, but also all processes that are triggered by them under the specified INPUT (recursively). I.e. the DSS constructs the “deductive hull” of the causal structure given the repository and, thus helps to reveal potential unwanted “side-effects”.
If the intended operation is specified by a set of requirements, GOALS, which are usually restrictions on the output water (thresholds for concentrations of substances, etc.), the DSS can check whether the designed system solves the task, i.e. the GOALS are entailed by the model:

$$\text{MODEL(S, INPUT)} \vDash \text{GOALS} \quad (1)$$

Since the repository is considered to be complete, i.e contains all available water processing steps as well as natural phenomena relevant to the domain design proposals could, in principle, also be automatically generated by the DSS, which may be less complex than expected, because the search is focused by both INPUT and GOALS. This may generate novel solutions, which, however, may be unintuitive or violating restrictions that cannot be expressed in the repository or in GOALS (e.g. because they are related to structural aspects and not local w.r.t. individual steps). Therefore, the first case studies will aim at interactive solutions.

Trouble Shooting

We assume that a system, S, that is deployed has been properly designed, which means if all elements of the plant work as expected and the contextual conditions stay within the anticipated range, the intended effects will be accomplished, which is expressed by (1) in the previous section. Observations of the actual system performance, OBS, may indicate a deviation from the expected operation, which is detected by the DSS as a contradiction between the assumption of the nominal INPUT and the system working according to MODEL(S, INPUT) and OBS:

$$\text{MODEL(S, INPUT)} \wedge \text{OBS} \vDash \bot \qquad (2)$$

For an operator, the task may then be identifying the cause behind the deviation from nominal behavior, if this is considered significant. In the DSS, this means hypothesizing

- an unanticipated INPUT_f (e.g. pH outside the expected range) that triggers unwanted or inhibits intended process steps, and/or
- a fault in the structure, S_f, (e.g. a valve being stuck, or a mixing device without power) which impairs the nominal operation.

Finding such causes, which we call **situation assessment**, can be guided by the repository by checking whether preconditions of expected processes could be invalidated or hypothesizing additional influences created by processes whose preconditions are satisfied unexpectedly (and then. perhaps, recursively searching for reasons for this).

As for design, the DSS may just be supportive to a human analyst in offering elements from the repository that might be involved in the disturbance. Alternatively, it might itself generate solutions and offer them to the operator for assessment (there will often be several potential explanations). The foundation for this are consistency-based diagnosis techniques, that were first developed for finding component faults [de Kleer-Williams, 1987] and then extended to process-oriented models ([Collins, 1993], [Heller, 2001], [Struss, 2008]). An illustrative example is presented in [Heller-Struss, 2002], [Struss 2020].

In any case, the criterion for a solution, i.e. a pair (S_f, INPUT_f) is that the hypothesized modification is consistent with the observations:

$$\text{MODEL}(S_f, \text{INPUT}_f) \wedge \text{OBS} \nvDash \bot \qquad (3)$$

which, again, can be automatically checked by the DSS.

Although this indicates the plant operates in an unexpected way, this does not necessarily imply that the GOALS cannot be achieved (The behavior could be simply unexpected, but not harmful). This can be done again by the DSS in a model-based way by checking whether the result of situation assessment (definitely or possibly) violates the GOALS:

$$\text{MODEL}(S_f, \text{INPUT}_f) \wedge \text{GOALS} \vDash \bot \qquad (4)$$

or, weaker,

$$\text{MODEL}(S_f, \text{INPUT}_f) \wedge \neg\text{GOALS} \nvDash \bot \qquad (5)$$

This means fault detection can be performed by the system, esp. in cases where not all GOALS are monitored explicitly continuously.

Intervention Proposal

If a (potential) violation of requirements has been detected in the previous step (by (4) or (5), remedial actions may need to be carried out that trigger a mitigation of the negative impact and/or a re-establishment of the proper performance. Actions that can possibly be carried by an operator can be included in the repository in a smooth way be representing them as processes that have a described effect, but no preconditions other than the decision to carry them out. It turns out that determining appropriate actions is similar to situation assessment (and can use the same algorithm), but aiming at consistency with GOALS, rather than with OBS(see (6) below).

The first question to be answered is which GOALS may require corrective actions. This can be answered by the DSS as a result of the checks (4) or (5), which will not only derive an inconsistency with the entire set, but with individual requirements. This determines a starting point and focus for searching the repository.

In an interactive solution, the DSS is able to identify active processes in the model that have an impact on the deviation from a violated goal and also ways to weaken or strengthen this impact by manipulating its input. Furthermore, it can identify process types from the repository that might have effects that counteract the deviation when introduced, e.g. an oxidation process reducing the concentration of dissolved iron which exceeds a certain threshold. Usually, actions will affect quantities only via a causal chain of triggered (natural or technical) processes (e.g. the action may be opening a valve, which triggers a flow of chlorine into the tank, which starts an oxidation process, which reduces the iron concentration).

Like in design, the DSS is able to apply the criterion for a solution, i.e. a set of interventions, ACTIONS, which, when applied to S_f promises to re-establish the GOALS:

$$\text{MODEL}(S_f, \text{INPUT}_f \cup \text{ACTIONS}) \vDash \text{GOALS'} \qquad (6)$$

which in a way shows intervention proposal as a form of redesign.

An important remark is that, in this step, we deliberately refer to a modified set of GOALS'. This reflects the fact that if a continuous quantity has a value that violates a certain requirement, it will do so for a while. Actions usually cannot cause discontinuous changes, and, hence, cannot be consistent with the original goal, but, rather, replaced by a restriction on its derivative in order to bring the magnitude into the proper range - ultimately.

On the other hand, the non-violated goals should be maintained, such that the check (6) can reveal if proposed actions restore some goals, but have side-effects that violate others.

As for the other tasks, the DSS functions can be exploited on demand as a support to a human, but also as a completely automatic search for a solution (see [Struss 2020] for an example), which will terminate, because the repository and the set of objects is finite, unless a modeling fault allows for the unbounded creation of object instances.

Education and Training

The knowledge captured by the repository and the functions that perform reasoning on its basis can support the education of non-experts in several ways.

The simplest form of supporting education is retrieval from the repository, e.g. by searching for processes that have an impact on particular characteristics of the processed water. This is actually planned to be the first function to be realized in the project, because it provides a benefit right away and is also necessary for populating and debugging the repository.

Beyond this, by supporting a What-if analysis, the DSS would critically analyze design activities of students and trouble shooting and corrective actions in hypothetical situations by plant operators.

Explanatory Capabilities

In particular, for educational purposes, it is important to note that the DSS does not just offer a solution or deny a proposed one, but can **generate comprehensible explanations** of its results and judgements. This is due to the fact that the model has a causal structure, as opposed to, for instance, a numerical simulator that can only generate data (sequences) based on equations.

For instance, if a design is refuted due to the violation of requirements, the DSS cannot only identify the violated goals, but also display the underlying causal structure (or the lack of such a structure). If an intervention is proposed, the system can explain in what way it contributes to achieving the goals in terms of a causal chain.

Challenges for AI Research

Building the envisioned DSS comprises a number of software engineering tasks regarding a web-based, multi-lingual solution, editors and GUIs, data storage for individual applications, etc.

Beyond this, producing a useful and useable tool, raises number of issues challenging AI, some of which are instances of more general and classical AI problems, which, however, need to and can be solved in the context of the special approach followed in the project. Our work can build on previous and ongoing research and some prototypical solutions and case studies ([Roque et al., 2003], [Struss-Selvamani, 2022]). Currently, the foundation for the repository is developed in a joint project of researchers and students from the Technical University of Munich and the Vellore Institute of Technology in Chennai.

These activities have shown the principled feasibility of the approach, but also highlighted a number of limitations and problems that need to be addressed – not for the sake of academic merits, but in order to be able to deliver a tool that provides real support in practice. We discuss what we consider to be the most important ones, in the order of urgency as we assess it at this stage. Indeed, one of the first tasks of the project will be producing a pragmatic plan for tackling them, in balancing the benefit w.r.t. the project objectives, i.e. ultimately measurable progress regarding SDG Goal 6, and the feasibility of obtaining a working solution in due time.

- **Integration of Component-oriented and Process-oriented Modeling**: While the dynamics of the treatment plant can be essentially represented by the combination of certain process steps, the structure of the plant is described by a number of components, such as containers and pipes, and the performance of the processes depends on the functioning of components like valves, mixing elements, etc. Hence, we need a systematic and seamless integration of component-oriented and process-oriented modeling and diagnosis (A proposal for such an integration is presented in [Struss-Selvamani, 2022]). Such a representation is mandatory for trouble shooting, because component failures may be the root cause of a malfunction of the plant. In design and education activities, it will usually be assumed that all elements function correctly and an explicit representation of components will be dispensable (unless the response of the system under a fault is to be analyzed in order to assess its resilience).
- **The Frame Problem**: A fundamental classical AI problem is raised in our context due to very practical requirements on how to represent the process steps in the treatment, which usually involve transportation of water from input to output. Such a step transforms only certain ingredients of the water while leaving others unaffected and transporting them to the output. What we would like to express in a formal way is “the step transforms ingredients a, b, c to a’, b’, c’, and all others are transported unchanged to the output”. The problem lies in representing “all others”. Although the ontology will capture what can potentially be contained in water, listing them as being simply transported by the water flow would not only lead to large models of process steps that have to deal with many ingredients that are not relevant in a particular problem, it is not feasible, if we consider that the ontology will evolve and that, for instance, adding new substances would require to modify all process types. We need to find not a general solution to the frame problem, but a manageable one in the restricted context of our approach.
- **Boundary of a Model and the Reasoning**: Constructing the system model in a “causally forward” direction means iteratively including newly triggered process instances and their effects. For a well-defined process type repository which does not allow loops in creating new object and relation instances, this process will always terminate. Trouble shooting and intervention proposal, however, include expanding the model in a “causally backward” direction (perform abductive reasoning). The

underlying algorithm, after having found a cause, will attempt to find a cause for it, and, hence, tends to be unbounded, chaining "why?" questions as children often do. It will terminate if there exist no process types whose instances provide a causal account, and will declare the model as inconsistent. Our current solution addresses this problem by allowing some elements to be "introducibles", i.e. they do not require a causal explanation in the model. However, the problem arises how to determine the introducibles. It will usually be impossible to expect a user to define them in a comprehensive way beforehand. After all, this would require anticipating the potential causal explanations generated by the DSS. The only feasible solution appears to be an interactive one, where the user decides on the fly, whether or not something needs further causal analysis.

- **Temporal Reasoning:** The current solution supports only snap-shot-like analysis, i.e. it assumes that for a particular situation, a causal explanation can be constructed within a (qualitative) temporal snapshot. More technically, the analysis does not go backward beyond integration steps. If they are included, there could be concurrent changes in the system, and different orders of their temporal occurrences would have to be considered. The resulting complexity may render the analysis (practically) intractable. Similarly, the generated interventions are currently only collections of actions, executed in parallel, rather than in a particular order or a certain point in time.
- **Focused Reasoning**: The automatic composition and analysis of a model aims at being comprehensive and, hence, will often include aspects and causal interdependencies that are relevant for solving a particular problem; overcoming this deficiency requires mechanisms for focusing. A number of problems studied in the AI fields of reasoning about actions and time and planning need to be solved – not in principle, but in the context of the chosen model-based approach.
- **Human-Machine Interaction:** the creation of the repository and its underlying ontology requires support to non-AI users in displaying their content in a natural, comprehensible way and allowing navigation through it. Also, generating explanations of solutions or inconsistencies and deficiencies is non-trivial, because it has to avoid excessive detail and address the user's view on the problem and systems. Semi-automatic solutions that involve human decisions at certain steps do not reduce, but emphasize the problem, because the user needs to be provided with information about the internal state of the problem solving.

Project Schedule

Our proposal aims at a contribution to speeding up the establishment of clean drinking water facilities. Given that activities related to this goal are significantly behind the schedule of SDG, the project cannot be run in a way, that it works on solving research problems for a while and after several years delivers a tool (or not). It has to be run like some kind of anytime algorithm, i.e. produce first results quickly that already have a practical impact and over time deliver a sequence of tools each of which adds to the functionality of the DSS. The ultimate criterion for planning this has to be the impact on the number of people who get access to safe drinking water as early as possible.

Therefore, in a first planning phase, the project has to

- determine a focus on treatment technologies that are expected to be the easiest available and most effective ones for the targeted regions and conditions
- assess the time needed to develop the various DSS functions, distinguishing between different features, esp. concerning the degree of automation.
- produce a project schedule based on a combination of the two criteria
- define appropriate case studies that allow to assess the respective solution.

Obviously, the first tasks to be carried out are the **realization of the representation of the repository** along with editors and retrieval functions as well as creating the software engineering foundations for the web-based solution. Actually, the former has already been started in the mentioned collaboration of the Technical University of Munich and the Vellore Institute of Technology.

The result is a prerequisite for the domain experts' task of populating the repository, but also allows to use it for **education and training** purposes.

With respect to other DSS functions, reflecting the feasibility of solutions, we currently propose to continue by realizing the **design support** function in an application where the user configures a plant based on the retrieval of process steps, and exploits the DSS for checking the result (according to (1) in section 2.2). The justification for this is that this solution requires only having the system build the model in the "causally forward" direction, i.e. collecting the impact of the proposed structure, and then checking its consistency with the requirements. In contrast, letting the DSS search for a solution, involves searching in the "causally backward" direction, is more complex and will require interaction with the user.

A similar argument applies to the **trouble shooting** task: a user could generate hypotheses about causes for behavior deviations which are then checked by the system. However, retrieving reasonable hypotheses is certainly more difficult for the user than selecting water treatment steps from the repository. Therefore, the user would benefit from the system extracting more information from the model of the misbehaving system, which lets this task appear more difficult than design.

Finally, **intervention generation** could also be driven by the user exploring the impact of hypothetical actions. However, this requires the result of situation assessment and also an appropriate representation of actions in the repository.As a result, we obtain an order of the high-level tasks. Please, note that the implementation of the algorithms solving the different tasks share a significant amount of software, in particular the automatic model configuration and the consistency check.

There are at least two dimensions that guide the expansion of the achieved results:

- **Growth of the repository:** start with the commonly available and effective technologies, then for trouble shooting add disturbances and/or add more technologies, for intervention proposal add actions to the repository
- **Degree of automation:** from user driven problem solving to more autonomously generated (partial) solutions.

Summary

The work on the proposed project does not start from scratch, and some development activities have already started. However, it needs additional resources to be able to contribute have an impact on progress regarding SDG Goal 6 in reality. This holds, in particular, for the acquisition of domain expertise and opportunities for carrying out realistic case studies in order to be able to focus the work on accomplishments that are needed and effective.

The project is intended to be very focused. Regarding the application, it will first consider drinking water treatment. We anticipate that much of the principled solutions can also be applied to waste water treatment. With respect to the methods and techniques applied, the first solutions will be exclusively exploit process-oriented modeling and problem solving. In the future, other techniques may be applied, for instance case-based reasoning (e.g. for proposing an initial design), data analysis and abstraction (to feed the high-level representation used in the DSS), or numerical modeling.

While there are still problems to be solved by, we are confident that we can nevertheless produce a sequence of results that promote the establishment and improved operation of treatment facilities with increasing power.

Acknowledgments

Previous work that forms the foundation for the proposal has been supported by Bavarian-Indian Center in its Virtual Collaboration Program, the Global Incentive Funds of Technical University of Munich, and Vellore Institute of Technology.

References

[Collins, 1993] J.W. Collins. Process-based diagnosis: An approach to understanding novel failures. PhD thesis, Institute for the Learning Sciences, Northwestern University,1993.

[de Kleer-Williams, 1987] de Kleer, J., Williams, B.C.: Diagnosing multiple faults. Artificial Intelligence 31(1) (1987)

[Dhar-Stein, 1997] Dhar, V. and Stein, R. Intelligent Decision Support Methods: The Science of Knowledge Work. Prentice-Hall, 1997

[Forbus, 1984] Forbus, K.: Qualitative process theory. Artificial Intelligence 24 (1984)

[Heller, 2001] Heller, U.: Process-oriented Consistency-based Diagnosis-Theory, Implementation and Applications. Dissertation, TU München (2001)

[Heller-Struss, 2002] Heller, U.,, Struss, P.: Consistency-Based Problem Solving for Environmental Decision Support. Computer-Aided Civil and Infrastructure Engineering 17, 79-92. (2002)

[LNOB, 2016] https://digitallibrary.un.org/record/1628748.

[Mannina et al., 2019] G. Mannina, T. Ferrerira Rebouças, A. Cosenza, M. Sànchez-Marrè y K. Gibert (2019). Decision Support Systems (DSS) for Wastewater Treatment Plants – A Review of the State of the Art.

[Poch et al., 2012] M. Poch, U. Cortés, J. Comas, I. Rodríguez-Roda, M. Sànchez-Marrè. (2012). Decisions on Urban Water Systems: some Support. Servei de Publicacions, Universitat de Girona. ISBN 978-84-8458-401-8. September 2012.

[Roque et al., 2003] Roque, W., Struss, P., Salles, P., Heller, U.: Design de um sistema de suporte à decisão baseado em modelos para o tratamento de ÁGUA. In: I Workshop de tecnologia da informação apliacada ao meio ambiente - cbcomp2003, 2003, Itajaí, SC , Anais do III Congresso Brasileiro de Computação, pp. 1894-1906. (2003)

[Sanchez-Marre, 2022] M. Sànchez-Marrè. Intelligent Decision Support Systems. Springer Nature Switzerland AG, March 2022. ISBN: 978-3-030-87789-7.

[SDG report, 2022] https://ustats.un.org/sdgs/rport/2022/The-Sustainable-Development-Goals-Report-2022.pdf

[Struss, 2008] Struss, P.: Model-based Problem Solving In: van Harmelen, F., Lifschitz, V., Porter, B. (eds.). Handbook of Knowledge Representation, Elsevier, pp. 395-465. (2008)

[Struss 2020] P. Struss: Model-based Decision Support Systems - Conceptualization and General Architecture. In: Trends in Artificial Intelligence, 33rd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2020, Kitakyushu, Japan, September 22-25, 2020

[Struss-Selvamani, 2022] P. Struss and R. Selvamani, Decision Support for Water Treatment – Challenging Model-based Reasoning, 35th International Workshop on Qualitative Reasoning

[UN70/1, 2015] https://www.un.org/en/dvelopment/desa/population/migration/generalassembly/docs/globalcompact/A_RES_70_1_E.pdf

[UN73/226, 2018] https://documents-dds-ny.un.org/doc/UNDOC/GEN/N18/460/07/PDF/N1846007.pdf?OpenElement

[Wotawa et al., 2010] Wotawa, F., Rodriguez-Roda, I. & Comas, J.: Environmental decision support systems based on models and model-based reasoning. Environmental Engineering and Management Journal 9(2), 189–195 (February 2010).