Journal of Logic, Language, and Information **6:** 5–31, 1997.
© 1997 *Kluwer Academic Publishers. Printed in the Netherlands.*

Zooming In, Zooming Out

PATRICK BLACKBURN
Computerlinguistik, Universität des Saarlandes, Postfach 1150, D-66041 Saarbrücken, Germany
E-mail: patrick@coli.uni-sb.de

MAARTEN DE RIJKE
Department of Computer Science, University of Warwick, Coventry CV4 7AL, U.K.
E-mail: mdr@dcs.warwick.ac.uk

(Received 8 November 1994; in final form 14 May 1996.)

Abstract. This is an exploratory paper about combining logics, combining theories and combining structures. Typically when one applies logic to such areas as computer science, artificial intelligence or linguistics, one encounters hybrid ontologies. The aim of this paper is to identify plausible strategies for coping with ontological richness.

Key words: Combinations of logics, complex structures, mathematics of modeling, modularity, modal logic, representation languages, transfer results

AMS Subject Classification (1991): 00A71, 03B45

CR Subject Classification (1991): F.4.0, F.4.1, I.2.4

1. Day 1: Examples

Zi: There's only two things I want to say: (a) Take things seriously, and (b) Let them talk to each other.

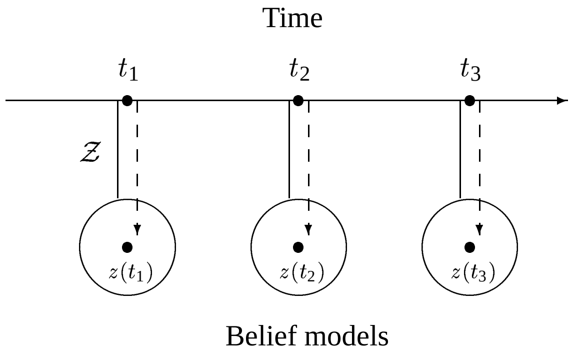
Zo: I'd appreciate a few more details. Over the past few weeks you've been talking about everything from Object Oriented Programming to Lexical Functional Grammar in the same breath. You've been saying that applied modal logic and situation theory are in the same business, and you've been using a lot of words I don't understand: "Communicating Structures", "Combined Logics", "Multiple Ontologies", "Layered Languages" . . . can't we go through it a little more systematically?

Zi: Well, sure. The important point is that real world systems are not flat, monolithic domains – they have a rich internal organization which guides the flow of information. The task of logical modeling is to capture the essence of this wealth. This suggests that it will be fruitful to investigate approaches in which multiple information sources and their interactions are the focus of attention.

To make this concrete I'll consider various combined structures. I'll start with a particularly simple type – what I call *refinement structures* – move onto the richer *classification structures*, and finish with *fully fibered* structures. But remember one thing. These are just special cases of a general idea, and it's the general idea I really want to get at.

Zo: I'll bear that in mind.

Zi: OK, let's begin with *refinement structures*. A nice example is the way Finger and Gabbay (1992) add a temporal component to reasoning about beliefs. They work with the following structures:



At the top you see a flow of time, a structure $(T, <)$, where $<$ linearly orders T. Each of the entities attached to T by $\mathcal{Z}$ is a belief structure. These are triples $(W, R, \{Q_p\}_{p \in P})$ where W is a collection of belief states, R transitively orders these states, and each Q_p is a unary relation on belief states (these tell us how the atomic information is distributed). To use Gabbay's terminology, these composite structures consist of a temporal flow *fibred* over a collection of belief models.

Zo: I've seen similar structures before, for example in first-order modal logic. But what have they got to do with *refinement*?

Zi: Because "refinement" sums up how the logics of the temporal and belief domains are combined. Clearly, these structures provide a semantics for two distinct languages: a temporal language $\mathcal{L}^{Tem}$ and a language for belief logic $\mathcal{L}^{Bel}$. (Finger and Gabbay choose $\mathcal{L}^{Tem}$ to be propositional Until/Since logic and $\mathcal{L}^{Bel}$ to be uni-modal belief logic.) The question is, how are we to reason about the composite ontology? We must combine the two languages, but how should this be done? Finger and Gabbay adopt a particularly simple solution: they use the $\mathcal{L}^{Bel}$ wffs as the atomic wffs of $\mathcal{L}^{Tem}$. That is, they build the $\mathcal{L}^{Tem}$ wffs in the usual way, but out of *structured* atomic wffs, namely the $\mathcal{L}^{Bel}$ wffs. Let's call this language $\mathcal{L}^{Tem}(\mathcal{L}^{Bel})$, the language $\mathcal{L}^{Tem}$ *layered over* the language $\mathcal{L}^{Bel}$.

Zo: Fine. But I still don't see what this has to do with refinement.

Zi: Consider how you'd evaluate a wff ϕ of the layered language. In general, ϕ will contain occurrences of the Until and Since operators, and they will move us round the time stream in the usual way. We keep evaluating subformulas in the familiar manner until we come to the "atomic" level. Usually we would simply invoke an assignment or valuation to see whether the atom was true; but now our atoms are structured and we have further work to do. We *zoom in* from the time of evaluation t to the associated belief model $\mathcal{Z}(t)$ and start evaluating the "structured atom" at $z(t)$.

Zo: I get it. The tense operators move us round the temporal level. While we're exploring this level we ignore the fact that each point of time is associated with structured information. However, when we get to the atomic level we take a more refined view. We zoom in on the associated belief models, and start exploring this lower level using the belief operator.

Zi: Exactly. It's a bit like working with the Macintosh's graphical user interface. The desktop may contain several icons, but as long as you're not doing anything in particular these icons are essentially just blobs. They may contain pictures, programs, text files or a variety of other things – but this complexity is hidden until it is actually needed. When we want to perform a certain task, we take a more refined view: we double-click on an icon, and zoom in to another level of structure.

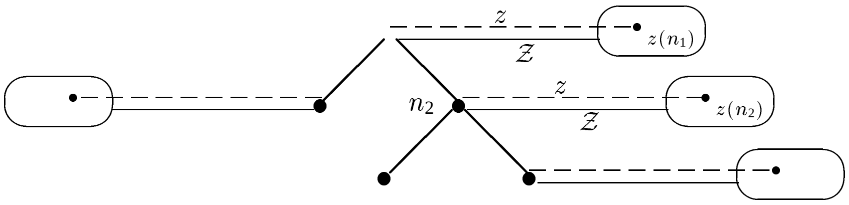
Zo: Refinement seems a fairly simple way of combining structures. You've got two ontologies, but the interaction between them is rather limited: the lower level just provides a refined notion of atomic information. Once you've worked your way down into the belief model, there's no way back up. Because temporal operators never occur under the scope of belief operators, you can't access the temporal flow from the belief structure. To use your Mac analogy, it's as if you couldn't close an icon once you'd opened it.

Zi: Right. You can't *zoom out*. This restriction has a number of pleasant consequences. There are often elegant ways of combining completeness and decidability results for the component logics into completeness and decidability results for the layered language; Finger and Gabbay give a number of examples and their results can be generalized.

However, you said something I don't like: "*because* temporal operators never occur under the scope of belief operators, you can't access the temporal flow from the belief structure". That's true enough, but it's a very syntactical way of viewing matters. The layering process whereby one language is embedded in another at the atomic level is certainly natural, but it's essentially syntactic sugar. We could have freely combined the two languages, allowing temporal operators inside the scope of belief operators. But if we did this any such wffs would evaluate to false at any node in the belief structures. You can't zoom out, and this is a *semantic* fact: the nodes in the belief models simply don't have permission to access the temporal structure. But when we get to fully fibered structures, things will be more democratic . . .

Zo: The idea of refining atomic information seems fairly natural. Do you have any other examples?

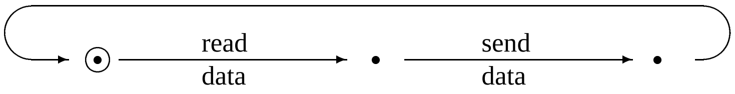
Zi: There's one in generative linguistics: the Generalized Phrase Structure Grammar (GPSG) of Gazdar et al. (1985) views linguistic structure as a combined ontology, namely finite trees fibred over finite feature structures. I don't have to tell you what a finite tree is. As for feature structures, they're just multi-modal Kripke models in which every relation is a partial function. When we fibre trees over feature structures we get entities of the following kind:



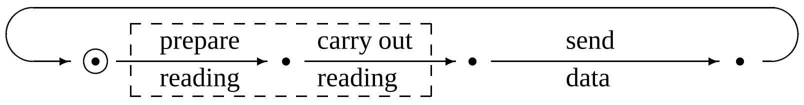
In GPSG, the feature structures are used to refine the notion of grammatical category. In contrast to the usual practice in formal language theory, where the nodes of parse trees are decorated with "indivisible" information about categories (for example NP for Noun Phrase, or VP for Verb Phrase) GPSG splits the atom: an NP becomes a structured object, a feature structure, that contains information about various subatomic features and values. GPSG is usually formulated using a mixture of formal language theory and feature logic, but it's straightforward to recast its central ideas using a language of trees layered over a language of feature structures. This is done by Blackburn et al. (1994). The tree language moves us around the tree, while the feature language allows us access to the inner structure of grammatical categories. Because such a layered language is all that's required, it's possible to prove completeness and decidability results for some quite expressive systems; Blackburn and Meyer-Viol (1996) supply details.

Zo: Still, like the earlier Finger and Gabbay example you gave, GPSG is only concerned with zooming in on objects or states. If you're interested in dynamic phenomena, or more generally, in notions of change, you should zoom in on actions or transitions.

Zi: Absolutely. For example, in the top-down design of distributed systems one uses actions and states on an abstract level to represent complex processes on a more concrete level. Van Glabbeek (1990) considers the design of an input device, repeatedly reading data and sending it off. A first, and highly abstract, description is given by the following picture:



On a slightly less abstract level of description the action "read data" breaks up in two parts: "prepare reading" and "carry out reading". This corresponds to the following refined picture:

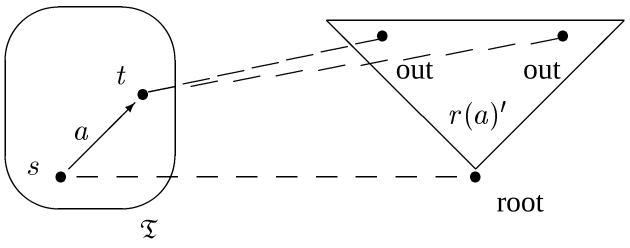


In this setting you find not just refinements of states (as in the earlier Finger and Gabbay, and GPSG examples), but also *refinement of actions* or *transitions*.

Another well known example is the *push down automata* that parse context free languages. A push down automata is a whole hierarchy of Kripke models: the models lower down in the hierarchy spell out the details of the transition possibilities that are invisible higher up.

Zo: Hmm. It seems you're after some sort of idea of substitution of structures. I'd like a more precise formulation.

Zi: OK, one way to formulate it is as follows. The relation between a labeled transition system $\mathfrak{T}$ and a structure $\mathfrak{T}'$ obtained from $\mathfrak{T}$ by refinement (or substitution) of actions, is given by a function from (atomic) actions to rooted transition systems. Let r be such a function; the *refinement of* $\mathfrak{T}$ *by* r is the structure that is obtained as follows. For $s \xrightarrow{a} t$ an edge in $\mathfrak{T}$ let $r(a)'$ be a new copy of $r(a)$; identify s with the root of $r(a)'$, identify t with all end nodes of $r(a)'$, and remove the edge $s \xrightarrow{a} t$. In other words, instead of making an a-transition at s, we now start at the root of $r(a)'$, traverse a terminating path through $r(a)'$, and then continue from t onwards.



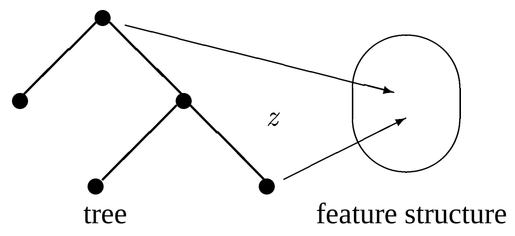
Zo: OK, that's clear. Is there anything more I should know about refinement?

Zi: Well, I think it would be interesting to look systematically at the converse of refinement: abstraction. For example, it's quite natural to take a structure $\mathfrak{A}$ to be an abstraction of a structure $\mathfrak{B}$ if $\mathfrak{A}$ is a quotient of $\mathfrak{B}$ under an appropriate notion of morphism. A general approach would allow for refinement/abstraction over any kind of item in ones structures ... but if we start following this up we'll be here all night.

∗ ∗ ∗

Zo: Tell me about classification structures.

Zi: OK. Let's return to generative grammar. Lexical Functional Grammar (LFG) is a nice example of classification structures at work; see Kaplan and Bresnan (1982). LFG, like GPSG, views syntactic structures in terms of composite entities made from finite trees and finite feature structures, but it glues them together differently. The basic picture to bear in mind is the following:



Here you see a single finite tree and a single finite feature structure linked by a partial function z. This feature structure induces a *classification* on the tree nodes via z.

Zo: I don't understand what you mean.

Zi: LFG theorists see sentences as simultaneously embodying two levels of structure. One level is called *constituent structure* and is represented using a tree. The feature structure represents *grammatical relations* such as subject, object, and indirect object. Syntactic explanations in LFG are couched in terms of classifications that grammatical relations induce on trees. LFG doesn't try to reduce grammatical relations to tree geometry: it insists that we are dealing with two independent, though interacting, ontologies.

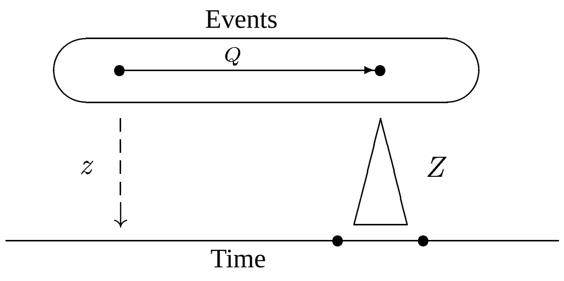
Representing natural language purely in terms of trees leads to a number of difficulties: sometimes representations of a sentence may place two components very far apart – and maybe even give them different category labels – while intuitively they "belong together". The classification semantics solves such puzzles. Two tree nodes t and t' may be distinct, but if $z(t) = z(t')$ they are functionally identical.

Zo: I have the feeling that we'll need more than layered languages to talk about classification structures.

Zi: Indeed. The name of the game isn't refinement any more, rather it's about ensuring that the internal structures of the two ontologies correctly "match". LFG does this using phrase structure rules annotated with equations. For example, the annotation $\uparrow = \downarrow$ means that if I move up from a tree node t to its mother node t' and then zoom into the feature structure, I arrive at the same point I would have reached by zooming in directly from t. This can be formulated very nicely in Propositional Dynamic Logic (PDL). Recall that PDL is a modal language where the modalities have internal structure; see Harel (1984) for further details. This is a very natural way of boosting expressive power; and, in the present case, it offers exactly what we need to capture the "commuting paths" idea. By making use of the *intersection constructor*, we can simply write down $\langle (up\,; zoom\_in) \cap zoom\_in\rangle \top$. This formula forces the desired matching of structure.

Zo: I think I'm beginning to get the idea. Do you have any other examples?

Zi: Sure. Here's one from formal semantics. There's been a lot of debate about whether point, interval, or event based systems give the best account of tense and temporal reference in natural language. This has given rise to interesting work – for example, various constructions for reducing one ontology to another – but it's not always obvious that such reductions should be made. Why not combine structures instead? For example, Blackburn, Gardent and de Rijke (1996) introduce Back and Forth Structures (BAFs). In their simplest form they look like this:



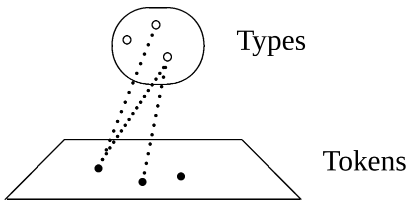
Here you see an interval structure and a simple event structure linked by a relation Z. The relation Z is constrained in various ways but the details aren't particularly important here. What *is* important is the extra flexibility this relation gives us. We're not forced to *identify* events and intervals, any more than in LFG we're compelled to identify grammatical relations with tree geometry. Rather, events are indirectly classified as having a certain temporal location. This enables us to take a more fine grained approach to temporal quantification. For example, following Moens and Steedman (1988), you can capture the intuition that the present perfect is "a past tense of present relevance". Basically, to evaluate a formula of the form PRES-PERF ϕ at an interval i, we search back for an interval j such that $j < i$ and with the additional property that if we zoom into the event structure at j we find the event ϕ. But this ϕ must also be of "present relevance". That is, ϕ must be Q-related to an event ψ such that by zooming back out to the interval structure at ψ we arrive at an interval k that overlaps our starting point i. We "complete a square" in the two structures back to our starting point.

Zo: I guess we'll need a fairly expressive language for coping with BAFs. If we work with a modal language we'll need at least the PRES-PERF operator in addition to modal operators local to each ontology. The PRES-PERF operator seems rather powerful: it wants to find its way back to (an interval overlapping) its starting place. Hey! That's just PDL program intersection in disguise!

Zi: Quite. But let's move on. I want to argue that there are deeper reasons for being interested in classification structures: whenever there are regularities and a flow of information, classification structures provide a natural modeling medium.

Zo: Hmm. This reminds me of something. The idea that constraints between structures guide information flow has always been present in Situation Theory. It's analyzed in detail in Seligman's (1990) Channel Theory.

Zi: Indeed. There are obvious links between Channel Theory and the present discussion. Seligman defines *classifications* to be triples $\mathbf{A} = \langle \mathit{tok}(\mathbf{A}), \mathit{typ}(\mathbf{A}), : \rangle$, where $\mathit{tok}(\mathbf{A})$ and $\mathit{typ}(\mathbf{A})$ are non-empty sets (of tokens and types, respectively), and : is the *classification relationship* between tokens and types. Here the types of



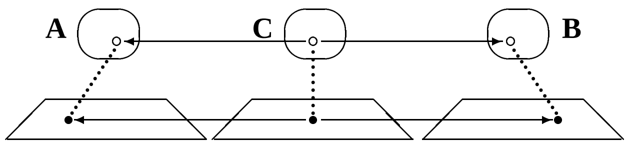
A classification

$\mathbf{A}$ classify the tokens of $\mathbf{A}$, and the types induce a natural equivalence relation $\sim$ of indistinguishability on tokens: $a \sim b$ iff for all types α we have $a : \alpha$ iff $b : \alpha$. As with LFG and its annotated phrase structure rules, further restrictions may be imposed on the way types and tokens interact.

Zo: But there's more to it than that. Classifications aren't considered in isolation, there's another level of stacking to internalize the notion of information flow.

Zi: Right – and so *channels* are introduced. A channel is something which directs information flow between classifications. First, a notion of information preserving morphisms between classifications $\mathbf{A}$ and $\mathbf{B}$ is defined as a certain kind of bi-function $f : \mathbf{A} \rightrightarrows \mathbf{B}$. Then, a channel $\mathbf{C} : \mathbf{A} \Longrightarrow \mathbf{B}$ is a classification $\mathbf{C}$ together with morphisms $\mathrm{left}_C : \mathbf{C} \rightrightarrows \mathbf{A}$ and $\mathrm{right}_C : \mathbf{C} \rightrightarrows \mathbf{B}$.

Roughly, the tokens of $\mathbf{C}$ are used to model connections between the tokens of $\mathbf{A}$ and the tokens of $\mathbf{B}$, and the types of $\mathbf{C}$ are used to express constraints between the types of $\mathbf{A}$ and the types of $\mathbf{B}$; and a connection is classified by a constraint



A channel

just in case information flows along the connection in a way that conforms to the constraint.

Both Channel Theory and classification structures emphasize the importance of highly structured universes in which some domains induce equivalence relations on others. However they tend to exploit these ideas differently. Channel theorists have made connections with the proof theoretic ideas underlying linear and relevance logics. I'm interested in exploring the connections with PDL and its cousins.

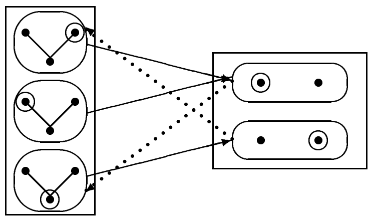
* * *

Zo: In all of the examples you've given so far there's an obvious master/slave relationship. In refinement structures one component is clearly boss; the others merely refine the transitions or states. Even with classification structures the information flow is essentially one way; one structure classifies the elements of the other. So I guess we haven't reached the end of the story yet.

Zi: Indeed not. In a number of recent talks and papers Dov Gabbay has advocated the idea of *fully fibering* two sets of semantic entities over each other; see Gabbay (1994). Roughly, a fully fibered structure consists of two classes of models, each class with its own language, plus a function between the classes that tells you how to evaluate formulas belonging to the one language inside structures of the other.

Zo: Give me a concrete example.

Zi: OK. Let's fully fiber finite trees and finite equivalence relations. For the sake of this example we assume that we have two mono-modal languages: $\mathcal{L}^T$ for talking about trees, and $\mathcal{L}^E$ for talking about equivalence relations.



Fibering a tree and an equivalence relation

First of all, let a *model-state* pair be a pair $(\mathfrak{M}, s)$where $\mathfrak{M}$ is a model based on a finite tree or on a finite equivalence relation, and s is an element of $\mathfrak{M}$. Second, let M_T, M_E be non-empty sets of model-state pairs whose first component is a finite tree or a finite equivalence relation, respectively, and such that if $(\mathfrak{M}, s) \in \mathsf{M}_T \cup \mathsf{M}_E$ and $s' \in \mathfrak{M}$, then $(\mathfrak{M}, s') \in \mathsf{M}_T \cup \mathsf{M}_E$. Now for the fibering function: let F be a pair of functions (F_T, F_E) with $F_T : \mathsf{M}_T \to \mathsf{M}_E$ and $F_E : \mathsf{M}_E \to \mathsf{M}_T$ such that model-state pairs that are mapped onto each other agree on all atomic symbols common to both languages; this function (pair) regulates the interaction between the classes of structures M_T and M_E. Finally, for F a fibering function, the F-fibered structure over M_T and M_E is the triple (W_F, R_F, V_F) such that

1. W_F is $\mathsf{M}_T \cup \mathsf{M}_E$,
2. R_F is $\{ ((\mathfrak{M}_1, s_1), (\mathfrak{M}_2, s_2)) : \mathfrak{M}_1 = \mathfrak{M}_2$ and $Rs_1s_2 \}$,
3. V_F is the union of the component valuations.

Zo: I see. And as to evaluating complex formulas, tree formulas are interpreted in M_T as usual, and likewise for $\mathcal{L}^E$-formulas and M_E. If we hit a tree formula while evaluating in M_E, we apply the fibering function F to the current model-state pair, and continue evaluating in its associated model-state pair in M_T; a similar move is made when we hit an $\mathcal{L}^E$-subformula while evaluating in M_T. We can zoom in and out with complete freedom.

Zi: You've got it. Incidentally, more involved definitions of fibering and similar constructions are possible – but the one I've given demonstrates that more equitable power relations are conceivable.

Zo: OK, let's try and summarize today's discussion. Interacting ontologies seem to abound in many applications. The essential task of the logician is to take this diversity seriously and to look closely at the lines of communication involved – I guess that's the content of the two slogans you started with.

Once you start looking at concrete applications you realize that there are many different ways that ontologies can communicate. Some of them are quite weak. For example, you have the notion of refinement where one ontology essentially serves to flesh out the low level information of another. But you also get stronger notions: forcing one ontology to "mimic" or "match" another in some way (your "classification structures"), or even allowing full two way communication ("full fibering").

At the logical level we are dealing with a hierarchy of constraint languages. With relatively weak notions, like refinement, there seem to be simple and well behaved ways of combining the "local logics"; for example, you can layer one language over another. But in general, the richer the interactions, the richer the language needed to exploit it. For example, we may need to add explicit zooming operators, or modality constructors à la PDL – and I guess it's conceivable that we may need to move up to full first-order expressive power or beyond. Does that sum it up so far?

Zi: It certainly does. Well, why don't we go grab a coffee?

2. Day 2: Trios

Zo: Good morning. Yesterday's examples were nice, but now I'd like some details. What kind of mathematical framework do you have in mind?

Zi: And good morning to you! Here's a first pass at a framework for combining structures. Let A and B be two classes of structures, and let Z be a collection of relations between the elements of A and those of B. Then the triple $(\mathsf{A}, \mathsf{Z}, \mathsf{B})$ is called a *trio*. The classes A and B are called the *left* and *right continents* of the trio, respectively, and Z is called its *bridge*.

Zo: I can see how some of yesterday's examples fit this scheme. In the case of Finger and Gabbay style refinement, for example, the trios have a left continent consisting of a single structure $\mathfrak{A}$, a right continent consisting of $|\mathfrak{A}|$ many structures $\{ \mathfrak{B}_a \mid a \text{ in } \mathfrak{A} \}$, and a bridge consisting of an injective function linking each point a of $\mathfrak{A}$ to an element in $\mathfrak{B}_a$.

Oh, and what you seem to need for the LFG style classification is trios that have both their left and right continents consisting of a single structure, and a bridge that is simply a partial function from the left continent to the right one. Is that correct?

Zi: Yes, that's right. Of course the notion of a trio is a very general one, and it remains to be seen whether they are mathematically interesting in their own right – but certainly *logical* questions concerning them abound, and that's what I'd like to talk about today. First I'll talk about logics of specific trios, then about analyzing specific bridges, and finally about the classification of bridges.

Consider two mono-modal languages $\mathcal{L}(\langle a \rangle)$ and $\mathcal{L}(\langle b \rangle)$ with modal operators $\langle a \rangle$ and $\langle b \rangle$ respectively, and suppose that we want to combine two structures $\mathfrak{A} = (W, \xrightarrow{a}, V)$ and $\mathfrak{B} = (W', \xrightarrow{b}, V')$ for those languages. As I am interested in the *connection* between the two structures, I'll explicitly add a binary relation Z between $\mathcal{L}(\langle a \rangle)$-structures and $\mathcal{L}(\langle b \rangle)$-structures; let me call such trios $(\mathfrak{A}, Z, \mathfrak{B})$ *connected trios*.

A first decision we have to make is: what language do we use to talk about connected trios? Given that we have components $\mathfrak{A}$, $\mathfrak{B}$ and Z, the natural set-up has two constants `left` and `right` to denote $\mathfrak{A}$ and $\mathfrak{B}$ respectively, and three modalities $\langle a \rangle$, $\langle b \rangle$ and $\langle z \rangle$, where $\langle z \rangle$ allows us to zoom in from $\mathfrak{A}$ to $\mathfrak{B}$ via Z. Actually, we'll soon want to add the converse modality $\langle z^{-1} \rangle$ to let us zoom back out – but let's look at this simpler language first.

An obvious first question is: what is the minimal logic of connected trios?

Zo: I think I have the answer; you need the **K** axioms and rules of inference for each of $\langle a\rangle$, $\langle b\rangle$ and $\langle z\rangle$. You also need:

1. $\texttt{left} \vee \texttt{right}$ and $\neg(\texttt{left} \wedge \texttt{right})$, to force every state into exactly one continent;
2. $\phi \leftrightarrow (\phi \wedge \texttt{left})$ for all $\mathcal{L}(\langle a\rangle)$ formulas, and likewise with $\texttt{right}$ and $\mathcal{L}(\langle b\rangle)$ formulas, to force $\texttt{left}$ and $\texttt{right}$ to denote the points in $\mathcal{L}(\langle a\rangle)$ models, and $\mathcal{L}(\langle b\rangle)$ models, respectively;
3. $\langle a\rangle\phi \to \texttt{left} \wedge \langle a\rangle(\texttt{left} \wedge \phi)$, to force $\xrightarrow{a}$ to be defined only on, and take values only in, the left continent. An analogous schema for $\langle b\rangle$;
4. $\langle z\rangle\phi \to \texttt{left} \wedge \langle z\rangle(\texttt{right} \wedge \phi)$, to force $\langle z\rangle$ to zoom in from the left continent to the right one.

I guess that's about it.

Zi: You're right. Completeness is easy to establish using a canonical model construction; the interpretations of the constants $\texttt{left}$ and $\texttt{right}$ determine the two continents of a trio. A straightforward filtration argument yields decidability.

So the basic logic is fairly simple. Things become more interesting when you constrain the connections between the component models; indeed, depending on the structure of the continents and of the bridge, you may want to add further items to your language. Let me give you an example involving bisimulations.

Recall that a non-empty binary relation $\leftrightarroweq$ between the domains W and W' of two models $\mathfrak{A} = (W, \xrightarrow{a}, V)$ and $\mathfrak{B} = (W', \xrightarrow{b}, V')$ is a *bisimulation* whenever it only relates points with the same atomic information and satisfies a back-and-forth condition: if $x, y \in \mathfrak{A}$, $x' \in \mathfrak{B}$, $x \xrightarrow{a} y$ and $x \leftrightarroweq x'$, then there is a $y' \in \mathfrak{B}$ such that $x' \xrightarrow{b} y'$ and $y \leftrightarroweq y'$ (and likewise in the opposite direction). Consider *bisimilar (connected) trios* $(\mathfrak{A}, \leftrightarroweq, \mathfrak{B})$ where $\leftrightarroweq$ is a bisimulation between $\mathfrak{A}$ and $\mathfrak{B}$. Because of the back-and-forth nature of bisimulations, the simplest modal language appropriate for talking about bisimilar trios has constants $\texttt{left}$, $\texttt{right}$, and modal operators $\langle a\rangle$, $\langle b\rangle$, $\langle z\rangle$ as before, as well as a modality $\langle z^{-1}\rangle$ to allow us to move from $\mathfrak{B}$ to $\mathfrak{A}$: we need to be able to zoom out to exploit the bisimulation effectively. To axiomatize validity on bisimilar trios you take the earlier axioms and add:

1. the **K** axioms for $\langle z^{-1}\rangle$, and $\langle z^{-1}\rangle\phi \to \texttt{right} \wedge \langle z^{-1}\rangle(\texttt{left} \wedge \phi)$;
2. the usual axioms from temporal logic saying that the relations used to interpret $\langle z\rangle$ and $\langle z^{-1}\rangle$ should be each others converse;
3. $\texttt{left} \wedge p \to [z]p$, and $\texttt{right} \wedge p \to [z^{-1}]p$ (*for* p *atomic*!), to force the condition on atomic information;
4. $\langle a\rangle\phi \to [z]\langle b\rangle\langle z^{-1}\rangle\phi$ and $\langle b\rangle\phi \to [z^{-1}]\langle a\rangle\langle z\rangle\phi$ to force the back-and-forth conditions.

There is no axiom forcing $\leftrightarroweq$ to be non-empty; this condition is simply not expressible.

To prove completeness, build the canonical model $(\mathfrak{A}, \leftrightarroweq, \mathfrak{B})$. It follows straightforwardly that the relations in this structure have almost all the required properties – the only possible defect is that $\leftrightarroweq$ may be empty. This is easily repaired: add to each of $\mathfrak{A}$ and $\mathfrak{B}$ a single isolated point in which (for example) all proposition letters are true; then extend $\leftrightarrows$ by adding to it the pair consisting of the two newly added points. This establishes completeness.

Zo: I get the general idea. But both of your completeness results concern rather abstract classes of trios – what about trios that arise in real applications? And do you have any results for refinement or classification structures? For example, if we think about GPSG in terms of trios we have to consider the following class of refinement structures: trios of the form $(\mathfrak{T}, Z, \{\,\mathfrak{F}_t\,\}_{t\in T})$. Here $\mathfrak{T} = (T, \leq, \mathit{root})$ is a *finite rooted tree*; Z is the *refinement relation* $\subseteq T \times \{\,\mathfrak{F}_t\,\}_{t\in T}$ that assigns to each tree node a unique $\mathfrak{F}_t$; the $\mathfrak{F}_t$'s are mutually disjoint *feature structures* $(W, \{R_\alpha\}_{\alpha\in A}, V, w)$ where W is a non-empty set, for each $\alpha \in A$, R_α is a binary relation on W that is a partial function, V is a function that assigns to each propositional symbol a subset of W, and $w \in W$ is the entrance to the feature structure where you end up after zooming in from the tree. I guess we could call such trios *GPSG trios*. What sort of logic do they have?

Zi: First we need to fix our language. Let's stick with the format we used for connected and bisimilar trios – logical aspects of the layered language approach are discussed by Blackburn and Meyer-Viol (1996). So, we have two constants `left` and `right` which tell us whether we're in the tree or in one of the feature structures. We also have a tree language $\mathcal{L}^T$ for moving around the tree, a modal language $\mathcal{L}^F$ with diamonds $\langle\alpha\rangle$ ($\alpha \in A$) for moving around the feature structure, and a device for zooming in: a diamond $\langle z\rangle$.

You axiomatize the logic of GPSG trios by taking the axioms for connected trios (adapted to $\mathcal{L}^T$ and $\mathcal{L}^F$), adding to it a complete axiomatization for finite trees, a complete axiomatization for feature structures (suitable calculi may be found in Blackburn and Meyer-Viol (1996), and Blackburn et al. (1996)), and finally an axiom to guarantee that each tree node is related to a single feature structure via the refinement relation Z:$\langle z\rangle\phi \leftrightarrow [z]\phi$.

Zo: Of course. So these results come relatively easily – but then, refinement of states is a fairly simple form of interaction between structures. What about refinement of transitions?

Zi: That's going to be more complex. Let's first make precise what we mean by action refinement. As a concrete example I'll consider PDL enriched with a mechanism for action refinement. Recall that in standard PDL programs $\alpha \in \Pi$ are generated from atomic programs a by the rule $\alpha ::= a \mid \alpha \cup \alpha \mid \alpha\,;\alpha \mid \alpha^*$; for simplicity I'll leave the test relations out.

Now, a *refinement function* is a function from atomic programs to programs. I'll talk about them as follows. Let $\mathcal{R}$ be a collection of refinement functions, and A a set of atomic programs. The programs of r-PDL – that is, PDL with action refinement – are produced by the following rules

$$\beta \ ::= \ a \mid r(a)$$
$$\alpha \ ::= \ \beta \mid \alpha \cup \alpha \mid \alpha \,;\, \alpha \mid \alpha^*$$

where $r \in \mathcal{R}$.

Zo: OK. That's simple enough. How do you interpret r-PDL?

Zi: First we assign meanings to refinement functions. For each atomic program a, $[\![r(a)]\!]$ is simply a finite rooted PDL model (that is, a labeled transition system) such that there is a leaf l that is reachable from the root via an $r(a)$-transition.

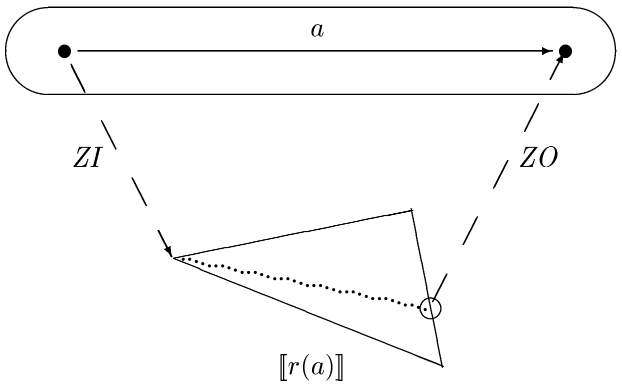
Zo: Am I right to assume that you'll now refine individual a-transitions by mirroring them by an $r(a)$-transition that lives inside a copy of $[\![r(a)]\!]$?

Zi: Yes. Although the general idea of adding structure to transitions by means of refinements is easy enough, the formal details are a little forbidding. First, a *witness* for a transition $w \xrightarrow{a} v$ is an isomorphic copy $[\![r(a)]\!]_{w \xrightarrow{a} v}$ of $[\![r(a)]\!]$; we assume that different transitions have disjoint witnesses.

Now, let $\mathfrak{M} = (W, \{\xrightarrow{\pi}\}_{\pi\in\Pi}, V)$ be a PDL model. Informally, the *refinement of* $\mathfrak{M}$ *by* $[\![\cdot]\!]$ is obtained by "mirroring" old transitions by witnesses of their refinements. Formally, it is the model

$$\mathfrak{M}' = (W', \xrightarrow{\pi}{}', V'; [\![\cdot]\!], ZI, ZO),$$

in which we match each transition $w \xrightarrow{\pi} v$ with the configuration $[\![r(a)]\!]_{w \xrightarrow{a} v}$:



That is: to each transition $w \xrightarrow{a} v$ we associate an isomorphic copy of $[\![r(a)]\!]_{w \xrightarrow{a} v}$, together with a *ZI*-link between w and the root of $[\![r(a)]\!]_{w \xrightarrow{a} v}$, and a *ZO*-link between the leaves of $[\![r(a)]\!]_{w \xrightarrow{a} v}$ that are $r(a)$-reachable from the root, and v. If you want, we can write down the formal details – but I don't think these will make things more perspicuous.

Zo: I think I can do without them. Roughly, what you're doing is taking the old model $\mathfrak{M}$ and adding to it, for every old transition a, a unique copy of an appropriate labeled transition structure, and you link these LTS's to $\mathfrak{M}$ by means of the relations *ZI* and *ZO*. In terms of trios, we have a trio $\mathfrak{M}' = (\mathfrak{M}, \{ZI, ZO\}, \biguplus\{[\![r(a)]\!]_{w \xrightarrow{a} v} \mid w \xrightarrow{a} v \text{ in } \mathfrak{M}\})$ consisting of the old model $\mathfrak{M}$ as the left continent, the disjoint union of the refined transitions as the right one, and the relations ZI, ZO as our bridge.

Zi: That's the main point, yes. Of course, the next step is to say how we interpret the formulas of r-PDL. Let $\mathfrak{M}$, $\mathfrak{M}'$ be as before. First of all, evaluation takes place in the left continent, that is: at states of the original, unrefined structure $\mathfrak{M}$. Formulas built according to the old PDL syntax are interpreted as before. Now, consider a new formula $\langle r(a)\rangle\phi$. Then $\mathfrak{M}', w \models \langle r(a)\rangle\phi$ iff

$$\exists v \in W\ (w \xrightarrow{a} v \wedge \exists xy \in [\![r(a)]\!]_{w \xrightarrow{a} v} \wedge ZI(w,x) \wedge ZO(y,v) \wedge v \models \phi).$$

So, first we look for an old a-successor v of w among the old states; we then zoom in from w to the refined transition $[\![r(a)]\!]_{w \xrightarrow{a} v}$, follow a terminating $r(a)$-path, and zoom out to v.

Zo: I get it. But it seems that you have to do a lot of work to get action refinement working, and I don't immediately see that it leads to interesting novel logical theorizing. Any comments?

Zi: If you mean that the logic of r-PDL models is essentially PDL combined with some of the ideas used for the logic of bisimilar trios, you're right; Blackburn and de Rijke (1996) can tell you more about that. But by moving from PDL models to r-PDL models we have greatly enhanced our capacity to model change realistically: we no longer have to think solely in terms of input/output pairs. As it turns out, this doesn't seem to lead to radically new logics – but that's just the way it goes!

Zo: Fair enough. Maybe we can forget about logics for refinement now, and discuss logics for other kinds of interaction. I guess that the languages you would devise for LFG or Channel Theory style classification would be variations of PDL, perhaps with additional structure on the programs?

Zi: Well, that's what I'll discuss here – though as I mentioned yesterday, there's interesting work relating Channel Theory to substructural logics. PDL and its cousins – especially those with some sort of intersection construct – are a natural tool for talking about classification structures. For, when working with classification structures one usually wants to demand that certain paths commute.

We covered some of these ideas in yesterday's discussion of LFG. Let's formalize them using trios. An *LFG trio* is a trio $(\mathfrak{T}, z, \mathfrak{F})$ in which $\mathfrak{T}$ is a single finite tree, $\mathfrak{F}$ a single finite feature structure, and z a partial function from the nodes of $\mathfrak{T}$ to the nodes of $\mathfrak{F}$. Let's assume that our favorite tree language $\mathcal{L}^T$ has a diamond $\langle t \rangle$ for moving down the tree, and let's also assume that our feature language $\mathcal{L}^F$ has diamonds $\langle f \rangle$, for every feature f. As you've probably guessed by now, $\mathcal{L}^{LFG}$, our language for talking about LFG trios, has constants denoting the continents, and a diamond $\langle z \rangle$ referring to the bridge. But, in addition, we're going to allow ourselves to build complex modalities – so we'll be working in a fragment of PDL. In particular, we'll use both the composition constructor ";" and the intersection constructor "$\cap$". This will enable us to construct diamonds such as $\langle (\mathit{up}\,;\mathit{zoom\_in}) \cap \mathit{zoom\_in} \rangle \top$, thus capturing the effect of the LFG equation $\uparrow = \downarrow$.

Zo: OK, that's clear. But what sort of results do you have for classification trios? I guess we could look at axiomatic issues again – but it may be more interesting to look at decidability and complexity questions. My feeling is that classification structures give you considerable coding power – even when the base languages for each continent have relatively easy satisfiability problems.

Zi: You've hit the nail on the head: the move from refinement to classification (or: from GPSG trios to LFG trios) is not as innocent as it looks. Because classification languages enable us to insist that sequences of transitions in the two structures commute, it's straightforward to code unsolvable problems. Here's a nice example. Consider type 0 grammars. Suppose we're given an alphabet $\mathcal{A}$. A *production* on $\mathcal{A}$ is an (unrestricted) rewrite pair $l \to r$, where l and r are words on $\mathcal{A}$. A *grammar* on $\mathcal{A}$ is a finite non-empty set of productions on $\mathcal{A}$. Is it possible to write a program that takes a grammar and a pair of words i and o and determines whether i can be rewritten to o using $\mathcal{A}$ productions? No, it's not. In fact it's a paradigmatic example of a computationally unsolvable task.

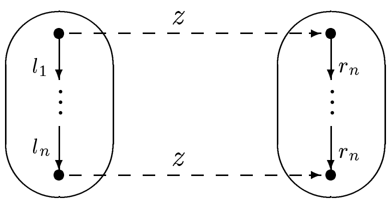
Zo: Hang on a minute ... I see how to code it. Let's work with classification structures in which each continent has partial functions corresponding to the elements of $\mathcal{A}$, and the bridge is a partial function z from the left continent to the right one. I'm going to use the left continent to code the antecedents of the productions, and the right continent to code the consequents.

Zi: What sort of language will you use?

Zo: I'll keep it simple. We'll have a modality $\langle a \rangle$ corresponding to each element a of $\mathcal{A}$ and an additional zoom-in modality $\langle z \rangle$. We'll also need a way of enforcing intercontinental synchronization; I'll assume that we can write down Kasper Rounds style path equations. That is, we have formulas $\langle m_1 \rangle \ldots \langle m_n \rangle \doteq \langle m'_1 \rangle \ldots \langle m'_k \rangle$ which equate two sequences of modalities. Such equations will hold at a node if there are transition paths from that node corresponding to both sequences, and, in addition, there is at least one node that is the joint destination of two such sequences; see Kasper and Rounds (1990) for details. We'll need the booleans, but we *won't* need any propositional variables; we can build all the formulas we'll need out of the path equations.

Zi: That's basically a fragment of PDL with intersection: over this class of trios the path equation $\langle m_1\rangle \ldots \langle m_n\rangle \doteq \langle m'_1\rangle \ldots \langle m'_k\rangle$ is just $\langle (m_1 ; \ldots ; m_n) \cap (m'_1 ; \ldots ; m'_k)\rangle \top$.

Zo: Indeed. Now, we don't need all the path equations. I'm only interested in enforcing commuting conditions *between* the continents, not *within* them, so we can make do with equations of the following form: $\langle l_1\rangle \ldots \langle l_n\rangle\langle z\rangle \doteq \langle z\rangle\langle r_1\rangle \ldots \langle r_k\rangle$, where $l_1, \ldots, l_n$ and $r_1, \ldots, r_k$ are all elements of $\mathcal{A}$. Such equations say that by taking the transition sequence l_1 through l_n in the left hand continent and then zooming in, we can get to a point that is also reachable by immediately zooming in and following the sequence r_1 through r_k in the right continent. The two paths commute.



Given such a language it's easy to encode grammars. A word $a_1 \cdots a_n$ on $\mathcal{A}$ is represented using the sequence $\langle a_1\rangle \cdots \langle a_n\rangle$. To each production $l \rightarrow r$ there is a corresponding path equation $\langle l\rangle\langle z\rangle \doteq \langle z\rangle\langle r\rangle$, where $\langle l\rangle$ and $\langle r\rangle$ are the modality sequences representing the words l and r respectively. A grammar $\mathcal{G}$ is represented by the conjunction of the corresponding equations; call this formula Γ. The problem of determining whether a word i can be rewritten to a word o using $\mathcal{A}$ productions amounts to asking whether we can find a classification structure that validates Γ and contains a node satisfying $\langle i\rangle\langle z\rangle \doteq \langle z\rangle\langle r\rangle$. As the rewriting problem is unsolvable, so is the corresponding satisfiability problem. This means that it's very easy to find undecidable logics of classification structures.

Zi: Exactly. Their architecture renders them perfect for coding arbitrary computations. It's quite fun – and fairly straightforward – to directly encode Turing machines, the Post completeness problem, and unbounded tiling problems. Classification structures are intrinsically dangerous! But in spite of this I'd argue that they're the appropriate setting for analyzing many practical problems.

Zo: That I don't see.

Zi: Look – working in ontologically rich settings gives you the opportunity to formulate problems naturally. When problems are formulated naturally it is easier to bring additional insights to bear. These may greatly simplify the inherent computational complexity. There's a classic example of this. It's long been known that undecidable problems can be formulated in the unrestricted LFG architecture. Nonetheless, precisely because LFG represents the various sources of linguistic information and their interactions so perspicuously, its easy to formulate linguistically natural conditions which regain decidability. Kaplan and Bresnan (1982) proposed such a condition, the *off-line parsability constraint*. This forbids the empty string from annotating any terminal tree node, and also prohibits the same syntactic category label from appearing twice on any non-branching chain of the tree. Linguistically these constraints are natural: they insist that all the information in the tree must be witnessed by lexical items. Their technical effect is to give a (grammar dependent) upper bound on both the height and depth of the possible parse trees for a given string. The existence of such upper bounds ensures that the parsing problem is decidable.

* * *

Zi: Let's change our perspective somewhat. Assume that we're interested in one particular kind of bridge, say X-bridges. What do we know about trios built around X-bridges, provided that we have 'nice' results for the continents? As an example, if we are given two continents, both with a complete and decidable logic, and if we build an X-bridge between them, will the resulting trio have a complete and decidable logic as well?

Zo: Why don't you give me an example of such a result, together with a detailed proof? Is anything known, for example, about the special case where the bridge is refinement of states?

Zi: Yes. Quite a few results are known here. Let me prove a fairly general one for layered languages. Recall that I write $\mathcal{L}^T(\mathcal{L}^B)$ to denote the language $\mathcal{L}^T$ (the *top* language) layered over $\mathcal{L}^B$ (the *bottom* language). For the result that I want to prove, I'll assume that both languages are modal languages, and that the top language $\mathcal{L}^T$ contains some means for randomly accessing nodes in its structures; I'll give a precise definition shortly. What I want to prove is that if you start with a complete logic $\mathbf{L}^T$ for the top language $\mathcal{L}^T$ and a complete logic $\mathbf{L}^B$ for the bottom language $\mathcal{L}^B$, then there exists a complete logic $\mathbf{L}^T(\mathbf{L}^B)$ for the layered language $\mathcal{L}^T(\mathcal{L}^B)$. This generalizes a result of Finger and Gabbay (1992).

Zo: OK. I'm all ears.

Zi: Right. Let's go to the blackboard then. Assume that the component logics $\mathbf{L}^T$ and $\mathbf{L}^B$ are both given as Hilbert-style systems that are complete for validity in $\mathcal{L}^T$ and $\mathcal{L}^B$, respectively. Equivalently, every $\mathbf{L}^T$-consistent formula has an $\mathcal{L}^T$-model, and every $\mathbf{L}^B$-consistent formula has an $\mathcal{L}^B$-model. Moreover, as I said, I will assume that the top language has a 'random access' operator E such that for any $\mathcal{L}^T$-model $\mathfrak{M}$ we have $\mathfrak{M}, w \models E\phi$ iff there exists some node v in $\mathfrak{M}$ such that $\mathfrak{M}, v \models \phi$. This can either be a primitive or a defined operator. (For example, if we were working in linear temporal logic we could define it by $P\phi \vee \phi \vee F\phi$.) All that matters is that we have it at our disposal.

Zo: Are you assuming anything else about $\mathcal{L}^T$ and $\mathcal{L}^B$ apart from the fact that they're modal languages?

Zi: Well, I'll assume that they're countable and classical. Moreover, to keep the proof as simple and clean as possible, I'll assume that they're one-dimensional: that is, formulas are evaluated at single states.

OK, let's begin. First we need to define three ingredients: the layered language $\mathcal{L}^T(\mathcal{L}^B)$, the models for this language and the combined logic $\mathbf{L}^T(\mathbf{L}^B)$. Now, the formulas of the combined language $\mathcal{L}^T(\mathcal{L}^B)$ are built up from $\mathcal{L}^T$ atoms and $\mathcal{L}^B$-formulas-viewed-as-atoms of $\mathcal{L}^T$. Note that I haven't added any modalities for exploiting the refinement relation between models for $\mathcal{L}^T$ and models for $\mathcal{L}^B$; we're working solely with the modalities of the two original languages. I'll assume that $\mathcal{L}^T$ and $\mathcal{L}^B$ have nothing in common except the boolean connectives.

As for the models, they are refinement trios of the form $(\mathfrak{M}, z, \mathsf{K})$ consisting of a single $\mathcal{L}^T$-model $\mathfrak{M}$ as the left continent and a class K of *pointed* $\mathcal{L}^B$-models as the right continent. By a pointed $\mathcal{L}^B$-model I simply mean a pair $(\mathfrak{N}, v)$ where $\mathfrak{N}$ is an ordinary $\mathcal{L}^B$-model, and v is a node in this model. As for the bridge, this is an injective function $z : |\mathfrak{M}| \to \mathsf{K}$, where $|\mathfrak{M}|$ denotes the domain of $\mathfrak{M}$. I will usually write $\mathfrak{N}_w, v_w$ for $z(w)$.

To interpret $\mathcal{L}^T(\mathcal{L})^B$-formulas on such refinement trios, let me first introduce some notation: for # an n-ary modal operator in $\mathcal{L}^T$, let *Table*$_\#(x, X_1, \ldots, X_n)$ be its table; that is, the recipe for computing the truth value of the formula $\#(\phi_1, \ldots, \phi_n)$. Here X_i stands for the set of states satisfying ϕ_i.

Given a refinement trio $(\mathfrak{M}, z, \mathsf{K})$, then for all nodes w in $\mathfrak{M}$ we define $(\mathfrak{M}, z, \mathsf{K}), w \models \phi$ as follows. First, for all $\phi \in \mathcal{L}^B$ we define:

$$(\mathfrak{M}, z, \mathsf{K}), w \models \phi \text{ iff } \mathfrak{N}_w, v_w \models \phi.$$

(Recall that $\mathfrak{N}_w, v_w$ is just $z(w)$.) Next, for all $\phi \in \mathcal{L}^T$:

$$\begin{aligned}
(\mathfrak{M}, z, \mathsf{K}), w \models p &\text{ iff } \mathfrak{M}, w \models p\\
(\mathfrak{M}, z, \mathsf{K}), w \models \neg\phi &\text{ iff } (\mathfrak{M}, z, \mathsf{K}), w \not\models \phi\\
(\mathfrak{M}, z, \mathsf{K}), w \models \phi \wedge \psi &\text{ iff } (\mathfrak{M}, z, \mathsf{K}), w \models \phi \text{ and } (\mathfrak{M}, z, \mathsf{K}), w \models \psi\\
(\mathfrak{M}, z, \mathsf{K}), w \models \#(\phi_1, \ldots, \phi_n) &\text{ iff } \textit{Table}_\#(w, [\![\phi_1]\!], \ldots, [\![\phi_n]\!]),
\end{aligned}$$

where $[\![\phi_i]\!] = \{u \mid (\mathfrak{M}, z, \mathsf{K}), u \models \phi_i\}$.

Zo: Let me think. OK, the last four clauses essentially say that $\mathcal{L}^T$-formulas are evaluated in the left continent $\mathfrak{M}$ in exactly the way we would expect – after all, $\mathfrak{M}$ is just a model for $\mathcal{L}^T$. So the only thing that is new is the first clause. This tells us that whenever we hit an $\mathcal{L}^B$-formula ϕ while evaluating at a point w, we should zoom into the pointed model in the right continent that's associated with w, and start evaluating ϕ at the designated point. That's OK too; $\mathfrak{N}_w$ is just a model for $\mathcal{L}^B$.

Zi: Exactly! Now, the combined logic $\mathbf{L}^T(\mathbf{L}^B)$ is defined as follows. Its axioms are those of $\mathbf{L}^T$. Its inference rules are those of $\mathbf{L}^T$ plus the following rule that allows one to lift validities from the bottom logic $\mathbf{L}^B$ to the combined one:

(Lift) For every $\mathcal{L}^B$-formula ϕ, if $\vdash_{\mathbf{L}^B} \phi$ then $\vdash_{\mathbf{L}^T(\mathbf{L}^B)} \phi$.

Zo: I can certainly see why $\mathbf{L}^T(\mathbf{L}^B)$ is sound for the combined models. But I think I can almost see how you could prove the logic to be complete as well. Simply take an $\mathcal{L}^T(\mathcal{L}^B)$-formula ϕ that is consistent in $\mathbf{L}^T(\mathbf{L}^B)$. If you could view ϕ as a (consistent) $\mathcal{L}^T$-formula, you could first use the completeness of $\mathbf{L}^T$ to find an $\mathcal{L}^T$-model for ϕ-viewed-as-an-$\mathcal{L}^T$-formula. That would give us half the desired model; the next step would be to make use of the completeness result for $\mathcal{L}^B$ to build the rest. Do you see what I'm trying to get at? First of all you 'hide' the $\mathcal{L}^B$-parts of the formula, and build a model for the $\mathcal{L}^T$ component; then you 'unpack' the $\mathcal{L}^B$-information and build the rest of the model.

Zi: That's *exactly* the idea behind the completeness proof! To make it work, first define a function *hide* : $\mathcal{L}^T(\mathcal{L}^B) \longrightarrow \mathcal{L}^T$ as follows. Let $q_0, q_1, \ldots$ be a collection of new proposition symbols, and let $\psi_0, \psi_1, \ldots$ be an enumeration of the $\mathcal{L}^B$-formulas. Now define

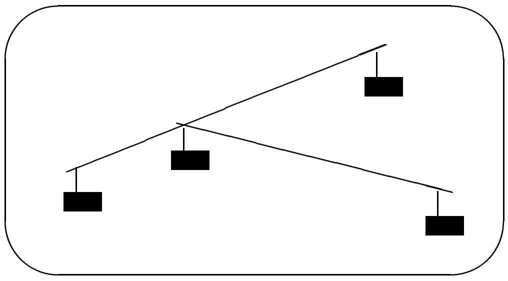
$$\begin{aligned}
\mathit{hide}(\psi_i) &= q_i\\
\mathit{hide}(p) &= p\\
\mathit{hide}(\neg\phi) &= \neg\mathit{hide}(\phi)\\
\mathit{hide}(\phi\wedge\psi) &= \mathit{hide}(\phi)\wedge\mathit{hide}(\psi)\\
\mathit{hide}(\#(\phi_1,\ldots,\phi_n)) &= \#(\mathit{hide}(\phi_1),\ldots,\mathit{hide}(\phi_n)).
\end{aligned}$$

The first thing we need to do is show that hiding preserves consistency. That's the content of the following lemma:

LEMMA A. *If ϕ is consistent in $\mathbf{L}^T(\mathbf{L}^B)$, then hide$(\phi)$ is consistent in $\mathbf{L}^T$.*

To prove this, one can argue by contraposition. If *hide*(ϕ) is inconsistent, this means $\vdash_{\mathbf{L}^T}$ *hide*$(\phi) \to \bot$. As all axioms and inference rules of $\mathbf{L}^T$ are also in $\mathbf{L}^T(\mathbf{L}^B)$, this proof can be mimicked to get a proof of $\phi \to \bot$ in $\mathbf{L}^T(\mathbf{L}^B)$, and this proves the lemma.

Zo: Alright, so from the $\mathbf{L}^T(\mathbf{L}^B)$-consistency of ϕ and the completeness of $\mathbf{L}^T$, we get an $\mathcal{L}^T$-model for *hide*(ϕ). But this model contains hidden $\mathcal{L}^B$-information – how do we extract it?



An $\mathcal{L}^T$-model with hidden $\mathcal{L}^B$-information

Zi: Indeed, that's the next issue we need to address. As we want to find $\mathcal{L}^B$-models for the hidden $\mathcal{L}^B$-information, we need to be sure that there are no hidden inconsistencies waiting to appear when we start the unpacking process. Let's check this out.

First, let's identify the information that's actually hidden. Let ψ be a formula in $\mathcal{L}^T(\mathcal{L}^B)$. Then

$$\mathit{Alien}(\psi) := \left\{ \begin{array}{c} \text{maximal subformulas of } \psi \text{ whose main} \\ \text{connective is not in } \mathcal{L}^T \end{array} \right\}.$$

We also need to define the closure of $\mathit{Alien}(\psi)$ under single negations:

$$\mathit{Alien}^{\neg}(\psi) := \mathit{Alien}(\psi) \cup \{\neg\chi \mid \chi \in \mathit{Alien}(\psi)\}.$$

Now, let $\mathfrak{M}$ be a model for $\mathcal{L}^T$, let w be a state in $\mathfrak{M}$, and define

$$\mathit{Unpack}_{\psi}(w) := \{\chi \in \mathit{Alien}^{\neg}(\psi) \mid \mathfrak{M}, w \models \mathit{hide}(\chi)\}.$$

That is: $\mathit{Unpack}_{\psi}(w)$ contains the relevant $\mathcal{L}^B$-information hidden at the state w in the $\mathcal{L}^T$-model $\mathfrak{M}$.

To rephrase our earlier requirement, what we need to prove is that $\mathit{Unpack}_{\psi}(w)$ is $\mathbf{L}^B$-consistent whenever ψ is $\mathbf{L}^T(\mathbf{L}^B)$-consistent. To achieve this, we look at all the problematic subsets of $\mathit{Alien}^{\neg}(\psi)$, and using the random access operator E we can make sure that they occur nowhere:

$$\begin{aligned} \mathit{Problem}(\psi) &:= \left\{ \bigwedge \Gamma \mid \Gamma \subseteq \mathit{Alien}^{\neg}(\psi) \text{ and } \Gamma \vdash_{\mathbf{L}^B} \bot \right\} \\ \mathit{No\text{-}Problem}(\psi) &:= \bigwedge \{\neg E\chi \mid \chi \in \mathit{Problem}(\psi)\} \\ \mathit{OK}(\psi) &:= \psi \wedge \mathit{No\text{-}Problem}(\psi). \end{aligned}$$

Zo: I get it. The key point is the use of the random access operator E in the definition of $\mathit{No\text{-}Problem}(\psi)$ to explicitly rule out all possible contradictions.

Zi: Indeed – and that's the only thing I use it for. Now, let's establish a few more lemmas and finish the completeness proof.

LEMMA B. $\vdash_{\mathbf{L}^T(\mathbf{L}^B)}$ *No-Problem*(ψ).

LEMMA C. $\vdash_{\mathbf{L}^T(\mathbf{L}^B)}$ *OK*$(\psi) \leftrightarrow \psi$.

LEMMA D. *Let $\mathfrak{M}$ be a model for hide(OK(ψ)); that is, assume that for some w in $\mathfrak{M}$, $\mathfrak{M}, w \models$ hide(OK(ψ)). If ψ is $\mathbf{L}^T(\mathbf{L}^B)$-consistent, then Unpack$_\psi(v)$ is $\mathbf{L}^B$-consistent for any v in $\mathfrak{M}$.*

To see that Lemma B holds, note that by propositional calculus

$$\vdash_{\mathbf{L}^T(\mathbf{L}^B)} \bigwedge\{\neg\chi \mid \chi \in \textit{Problem}(\psi)\}.$$

But then we can deduce

$$\vdash_{\mathbf{L}^T(\mathbf{L}^B)} \neg E \neg \bigwedge\{\neg\chi \mid \chi \in \textit{Problem}(\psi)\}.$$

(This step is just an application of the modal necessitation rule for E, and as $\mathbf{L}^T$ is a complete logic, this rule must be derivable.) Using the de Morgan laws and the fact that E commutes over disjunctions, it is easy to see that this formula is equivalent to *No-Problem*(ψ). Thus Lemma B is proved. Lemma C is an immediate consequence of Lemma B.

Zo: I see. Let me try to prove Lemma D. Assume that *Unpack*$_\psi(u)$ is $\mathbf{L}^B$-inconsistent for some u. That is: there are $\chi_1, \ldots, \chi_n \in$ *Unpack*$_\psi(u)$ such that $\vdash_{\mathbf{L}^B} \chi_1 \wedge \ldots \wedge \chi_n \to \bot$. It follows that $(\bigwedge_i \chi_i) \in$ *Problem*(ψ), and so $\neg E\, (\bigwedge_i \chi_i)$ is one of the conjuncts in *No-Problem*(ψ). Now, by assumption we have that

$$\begin{aligned}
\mathfrak{M}, w \models \textit{hide}(\textit{OK}(\psi)) &\Rightarrow \mathfrak{M}, w \models \textit{hide}(\textit{No-Problem}(\psi))\\
&\Rightarrow \mathfrak{M}, w \models \textit{hide}(\neg E \textstyle\bigwedge_i \chi_i)\\
&\Rightarrow \mathfrak{M}, w \models \neg E\, \textit{hide}(\textstyle\bigwedge_i \chi_i)\\
&\Rightarrow \mathfrak{M}, v \models \neg \textstyle\bigwedge_i \textit{hide}(\chi_i), \text{ for all } v \text{ in } \mathfrak{M}.
\end{aligned}$$

On the other hand, as $\chi_i \in$ *Unpack*$_\psi(u)$, $\mathfrak{M}, u \models \bigwedge_i$ *hide*(χ_i), and we have a contradiction. This proves Lemma D.

Zi: Quite right! And with these lemmas at our disposal, we are ready to prove the completeness of the combined logic $\mathbf{L}^T(\mathbf{L}^B)$. To be precise, I want to prove:

THEOREM E. *Assume that $\mathbf{L}^T$ and $\mathbf{L}^B$ are complete in $\mathcal{L}^T$ and $\mathcal{L}^B$, respectively. Let ϕ be a formula in $\mathcal{L}^T(\mathcal{L}^B)$. Then ϕ is valid on all $\mathcal{L}^T(\mathcal{L}^B)$-models iff it is* *provable in the combined logic* $\mathbf{L}^T(\mathbf{L}^B)$.

To prove the completeness half of this theorem (that is, the right to left direction), let's assume that ϕ is an $\mathbf{L}^T(\mathbf{L}^B)$-consistent formula in the combined language. By Lemma C, $OK(\phi)$ is $\mathbf{L}^T(\mathbf{L}^B)$-consistent as well, and by Lemma A $hide(OK(\phi))$ must be $\mathbf{L}^T$-consistent. The completeness of $\mathbf{L}^T$ gives us an $\mathcal{L}^T$-model $\mathfrak{M}$ and a state w in $\mathfrak{M}$ such that $\mathfrak{M}, w \models hide(OK(\phi))$.

Fix any state u in $\mathfrak{M}$. By Lemma D it follows that $Unpack_\phi(u)$ is $\mathbf{L}^B$-consistent for all u. Now use the finiteness of $Unpack_\phi(u)$ and the completeness of $\mathbf{L}^B$ to find an $\mathcal{L}^B$-model $(\mathfrak{N}_u, v_u)$ for $Unpack_\phi(u)$. Define

$$\begin{aligned} \mathsf{K} &:= \{(\mathfrak{N}_u, v_u) \mid u \in \mathfrak{M}\} \\ z(u) &:= (\mathfrak{N}_u, v_u) \text{ for } u \in \mathfrak{M}. \end{aligned}$$

Then, by an easy induction we find that for every u in $\mathfrak{M}$ and every subformula ψ of $OK(\phi)$

$$\mathfrak{M}, u \models hide(\psi) \text{ iff } (\mathfrak{M}, z, \mathsf{K}), u \models \psi.$$

So, from $\mathfrak{M}, w \models hide(OK(\phi))$ we get $(\mathfrak{M}, z, \mathsf{K}),\ w \models OK(\phi)$ and hence $(\mathfrak{M}, z, \mathsf{K}),\ w \models \phi$, as required. Thus we have proved the desired completeness result.

Zo: Let me try and summarize. First you hide the $\mathcal{L}^B$-information in ϕ, and appeal to the completeness of $\mathbf{L}^T$ to get an $\mathcal{L}^T$-model for $hide(\phi)$. Then you unpack the hidden information, and give it suitable models. This step uses the completeness of $\mathcal{L}^B$, and we can be sure that nothing goes wrong here because our use of the random access operator guarantees that there are no $\mathbf{L}^B$-inconsistencies lurking somewhere. 'Hide and unpack' is really a very simple idea!

Zi: Yes, indeed – and one can use it to give transfer results for other bridges besides state refinement. For example, transfer results for bisimilar trios are easily obtained, though there is more bookkeeping to do. But the general idea is natural. Simply iterate the above procedure: move back and forth between the structures, all the time hiding and unpacking information.

3. Day 3: Questions

Zo: I've enjoyed our discussion. Let's suppose that you're right, and it really is important to develop ways of thinking about rich ontologies and their logics. OK, so where do we go from here? I can see that standard techniques – for example Henkin models and filtrations – can sometimes be applied in richer settings. And maybe 'hiding and unpacking' will turn out to be a useful technique . . .

Zi: I think it will. It underlies the results of Fine and Schurz (1991) and Kracht and Wolter (1991) concerning transfer of completeness and decidability of modal logics to their independent joins, and the Finger and Gabbay (1992, 1996) completeness results for layered and fibered languages.

Zo: It also seems that we know some of the risks involved. For example, the LFG example shows that undecidability lurks just around the corner. And I guess there's often going to be increases in computational complexity . . .

Zi: Quite. Spaan (1993) and Hemaspaandra (1994, 1996) give a good map of the area.

Zo: OK. So we already know a reasonable amount. But once again: where do we go from here? I can see a real danger. Your examples are very nice, but your technical remarks strike me as a bit piecemeal. You run the risk of endless *ad-hoc* investigations, starting completely anew each time a new application comes along.

Zi: I think that's a fair comment. What's required is a more serious classification of trios. At the moment I only distinguish between refinements, classifications and full interaction. That's not good enough. We need to know what kind of bridges there are, and what kind of interactions between structures allow a transfer of properties of the component logics to the trios. From the examples we've looked at we know that completeness transfers if we consider refinement trios. We also know that decidability does not transfer to LFG style classification trios. But where is the boundary between transfer and non-transfer? And does completeness always transfer in the case of uni-directional bridges?

To answer such questions a more abstract approach may be fruitful. There's some examples of such work. For example Seligman (1996) takes an algebraic perspective on logical combination; interestingly, the "hide and unpack" idea emerges as a natural construction. Kurucz et al. (1996) use category theory as a framework for investigating combined logics. Given the way category theory focuses on the links between mathematical structures, this may turn out to be the natural home for further investigations. I think 2-categories and fibrations may prove especially useful. Similar categorical approaches have already proved useful in abstract studies of modularity; see Goguen and Burstall (1984).

The ideal, I think, is to pursue both concrete and abstract investigations. This will ensure that the guiding intuitions are preserved, and that we don't run the risk of missing the wood for the trees.

* * *

Zo: There is an obvious weakness of the story you have been telling so far. You have acted as if combined ontologies are lifeless, *static* entities. This ignores the fact that for many applications it is precisely the *dynamic* aspects of combined ontologies that are of interest – for example, how they are to be built, extended, or otherwise amended over time.

Zi: I couldn't agree more. Let's consider a concrete example: the Tree Adjoining Grammars (TAGs) of Joshi et al. (1975). TAG analyses are essentially dynamic; sentences are viewed as the result of merging trees together. To gain something of the flavour of TAG in action, consider the operation known as *adjunction*. Let τ be a tree with an internal node labeled by the non-terminal symbol A. Let ρ be an auxiliary tree with root and foot node labeled by the same non-terminal symbol A. The tree τ' that results by adjoining ρ at the A-labeled node in τ is formed by removing the subtree of τ rooted at this node, inserting ρ in its place, and then reinserting the subtree at the foot node of ρ. Perhaps the most important thing to notice is the role played by the node labeled A. We began with an initial structure (namely τ) with a designated node (namely that labeled A); we then performed a computation step; and this created a larger structure with a new designated node, the site for further creation.

Of course, all this *could* be described statically. But to do so may do violence to the underlying intuitions. We need analyses which cope with the growth of structures rather than merely treating them as completed objects.

Zo: And do you have such an analysis?

Zi: No, but I do have some suggestions. Firstly, underlying the TAG example is a notion of substitution of structure, much as we discussed earlier. Secondly, dynamics is not alien to logic. For example, formal proof construction can be naturally thought of as information processing, and recent work in substructural logics (influenced by Girard's (1987) treatment of Linear Logic) addresses such issues as the management of limited resources over time.

Moreover, there are a number of logical systems that draw on computational metaphors such as "information growth", "transition sequences" and "updating". The classic example is the Kripkean analysis of intuitionistic logic; more recent examples include propositional dynamic logic (Pratt, 1976; Harel, 1984), action algebras (Pratt, 1991) Peirce algebras (Brink et al., 1994; de Rijke, 1995)), the various "arrow logics" (van Benthem, 1993; Marx et al., 1996; Vakarelov, 1996; Venema, 1996; de Rijke, 1992) and evolving algebras (Gurevich, 1991).

As yet no consensus has emerged, but such systems have already provided interesting analyses of programming languages (Börger, 1990), theory change (Gärdenfors, 1988; de Rijke, 1994), anaphora and presupposition in natural language (Groenendijk and Stokhof, 1991; Beaver, 1994), and various syntactic formalisms (Moshier and Rounds, 1987; Johnson and Moss, 1994). What will ultimately emerge is unclear – but I do think it's both interesting and important to get to grips with dynamic issues.

Look, let's wrap things up here. As far as I'm concerned, the most important point is the following. Logic is increasingly being influenced by applications – and not just the traditional applications in philosophy and mathematics. Instead, new interdisciplinary work in such areas as Computer Science, Artificial Intelligence and Theoretical and Computational Linguistics are becoming the focus of attention.

This broadening of the scope of applied logic brings new responsibilities. It forces the logician to take ontological diversity seriously, and to consider how rich systems evolve over time.

Acknowledgment

Part of this research was carried out while Patrick Blackburn was at the Philosophy Department of Utrecht University, and Maarten de Rijke at CWI, Amsterdam. During this period both authors were supported by the Netherlands Organisation for Scientific Research (NWO), project NF 102/62-356.

References

Beaver, D., 1994, "What comes first in dynamic semantics?," PhD Thesis, University of Edinburgh.

van Benthem, J., 1991, *Language in Action*, Amsterdam: North-Holland.

Blackburn, P., Gardent, C., and Meyer-Viol, W., 1993, "Talking about trees," pp. 21–29 in *Proceedings of the 6th Conference of the European Chapter of the Association of Computational Linguistics*, Utrecht.

Blackburn, P., Gardent, C., and de Rijke, M., 1996, "Rich ontologies for tense and aspect," in *Logic, Language and Computation*, J. Seligman and D. Westerståhl, eds., Stanford: CSLI Publications. To appear.

Blackburn, P. and Meyer-Viol, W., 1996, "Modal logic and model theoretic syntax," in *Advances in Intensional Logic*, M. de Rijke, ed., Dordrecht: Kluwer. To appear.

Blackburn, P., Meyer-Viol, W., and de Rijke, M., 1996, "A proof system for finite trees," in *Computer Science Logic '95*, Berlin: Springer-Verlag. To appear.

Blackburn, P. and de Rijke, M., 1996, Logics of communicating structures. In preparation.

Börger, E., 1990, "A logical operational semantics for full Prolog, Part 1." pp. 36–64 in *Proceedings CSL '89*, E. Börger et al., eds., LNCS 440, Berlin: Springer-Verlag.

Brink, C., Britz, K., and Schmidt, R., 1994, "Peirce algebras," *Formal Aspects of Computing* **6**, 339–358.

Fine, K. and Schurz, G., 1991, "Transfer theorems for stratified multi-modal logics," in *Proceedings of the Arthur Prior Memorial Conference.* To appear.

Finger, M. and Gabbay, D.M., 1992, "Adding a temporal dimension to a logic system," *Journal of Logic, Language and Information* **1**, 203–233.

Finger, M. and Gabbay, D.M., 1996, "Combining temporal logic systems," *Notre Dame Journal of Formal Logic.* To appear.

Gabbay, D.M., 1994, "Fibered semantics," manuscript, London: Imperial College.

Gärdenfors, P., 1988, *Knowledge in Flux*, Cambridge, MA: MIT Press.

Gazdar, G., Klein, E., Pullum, G., and Sag, I., 1985, *Generalised Phrase Structure Grammar*, Oxford: Basil Blackwell.

Girard, J.-Y., 1987, "Linear logic," *Theoretical Computer Science* **50**, 1–102.

van Glabbeek, R., 1990, "Comparative concurrency semantics and refinement of actions," PhD Thesis, Vrije Universiteit Amsterdam.

Goguen J.A. and Burstall, R.M., 1984, "Some fundamental algebraic tools of the semantics of computation (I)," *Theoretical Computer Science* **31**, 175–209.

Groenendijk, J. and Stokhof, M., 1991, "Dynamic predicate logic," *Linguistics and Philosophy* **14**, 39–100.

Gurevich, Y., 1991, "Evolving algebras; a tutorial introduction," *Bulletin EATCS* **43**, 264–284.

Harel, D., 1984, "Dynamic logic," pp. 497–604 in *Handbook of Philosophical Logic 2*, D. Gabbay and F. Guenthner, eds., Dordrecht: Reidel.

Hemaspaandra, E., 1994, "Complexity transfer for modal logic," pp. 164–173 in *Proceedings 9th LICS*, Paris.

Hemaspaandra, E., 1996, “The price of universality,” *Notre Dame Journal of Formal Logic*. To appear.

Johnson, D., and Moss, L., 1994, “Grammar formalisms viewed as evolving algebras,” *Linguistics and Philosophy* **17**, 537–560.

Joshi, A., Levy, L., and Takahashi, M., 1975, “Tree adjunct grammars,” *Journal of Computer and Systems Sciences* **10**, 136–163.

Kaplan, R. and Bresnan, J., 1982, “Lexical-functional grammar: a formal system for grammatical representation,” pp. 173–281 in *The Mental Representation of Grammatical Relations*, J. Bresnan, ed., Cambridge, MA: MIT Press.

Kasper, R. and Rounds, W., 1990, “The logic of unification in grammar,” *Linguistics and Philosophy* **13**, 33–58.

Kracht, M. and Wolter, F., 1991, “Properties of independently axiomatisable bimodal logics,” *Journal of Symbolic Logic* **56**, 1469–1485.

Kurucz, A., Jánossy, A., and Eiben, G., 1996, “Combining algebraizable logics,” *Notre Dame Journal of Formal Logic*. To appear.

Marx, M., Németi, I., Sain, I., and Mikulás, S., 1996, “Investigations in arrow logics,” in *Arrow Logics and Multi-modal Logics*, M. Marx, L. Polos, and M. Masuch, eds., Studies in Logic, Language and Information. Stanford: CSLI Publications. To appear.

Moens, M., and Steedman, M., 1988, “Temporal ontology and temporal reference,” *Computational Linguistics* **14**, 15–28.

Moshier, D., and Rounds, W., 1987, “A logic for partially specified data structures,” pp. 156–167 in *Proceedings of the 14th ACM Symposium on Principles of Programming Languages*, Munich.

Pratt, V., 1976, “Semantical considerations on Floyd–Hoare logic,” pp. 109–121 in *Proceedings 17th IEEE Symposium on the Foundations of Computer Science.*

Pratt, V., 1991, “Action logic and pure induction,” pp. 79–120 in *Logics in AI*, J. van Eijck, ed., LNAI 478, Berlin: Springer.

de Rijke, M., 1992, A system of dynamic modal logic, Report #CSLI-92-170, Stanford University. To appear in *Journal of Philosophical Logic.*

de Rijke, M., 1994, “Meeting some neighbours,” pp. 170–195 in *Logic and Information Flow*, J. van Eijck and A. Visser, eds., Cambridge, MA: MIT Press.

de Rijke, M., 1995, “The logic of Peirce algebras,” *Journal of Logic, Language and Information* **4**, 227–250.

Seligman, J., 1990, “Perspectives: A relativistic approach to the theory of information,” PhD Thesis, University of Edinburgh.

Seligman, J., 1996, Combining logics with ease. In preparation.

Spaan, E., 1993, “Complexity of modal logics,” PhD Thesis, ILLC, University of Amsterdam.

Vakarelov, D., 1996, “Modal arrow logic,” in *Advances in Intensional Logic*, M. de Rijke, ed., Dordrecht: Kluwer. To appear.

Venema, Y., 1996, “A crash course in arrow logic,” in *Arrow Logics and Multi-modal Logics*, M. Marx, L. Polos and M. Masuch, eds., Studies in Logic, Language and Information. CSLI Publications, Stanford. To appear.