Local Search and Modal Logic

Maarten C. Stol[⋆] and Maarten de Rijke[⋆⋆]

Institute for Logic, Language and Computation (ILLC)
Faculty of Science, University of Amsterdam
Plantage Muidergracht 24, 1018 TV Amsterdam
Email: {mstol, mdr}@science.uva.nl
URL: http://www.science.uva.nl/~{mstol, mdr}

Abstract. Local search techniques have widespread use for solving propositional satisfiability problems. We investigate the use of adaptive local search techniques for model generation problems for modal logics; we focus on the modal logic **S5**. A local search algorithm extended with an adaptive heuristic is presented and tested on an ensemble of randomly generated problem instances. We briefly discuss the limitations of using local search for other NP-complete modal logics.

1 Introduction

There is a clear need for automated reasoning methods for modal and modal-like logics [1]. As many reasoning tasks for such logics are NP-, PSPACE-, or even EXPTIME-hard, there is usually no single algorithm which can be developed, implemented and then used as a black box that performs well on all inputs. Instead, we need a *variety* of qualitatively different algorithms and heuristics for solving reasoning tasks, each of which may perform well on only a limited class of inputs. Combined, such solvers should cover as many inputs as possible.

There is a broad spectrum of methods for solving satisfiability problems for propositional logic [5]. Unfortunately, this diversity is largely absent for modal and modal-like logics, where most algorithms for satisfiability checking and model generation are either resolution-based (usually through translations into first-order logic) or tableau-based. The aim of this paper is to help broaden the spectrum of methods for modal logic by developing an algorithm for model generation based on local search [8].

The rest of the paper is organized as follows. In Section 2 we recall basic facts about local search and modal logic. We then develop a local search algorithm for the modal logic **S5**; we discuss the representation of candidate solutions and adaptive mechanisms in some detail. We conclude by discussing local search for other NP-complete modal logics and for modal logics beyond NP.

[⋆] Supported by the Netherlands Organization for Scientific Research (NWO) under project number 612.069.006.

[⋆⋆] Supported by the Spinoza project 'Logic in Action' and by grants from the Netherlands Organization for Scientific Research (NWO), under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, and 220-80-001.

2 Background

Given a problem, each instance is associated with a finite set of candidate solutions. Every candidate has a cost, with each candidate a neighborhood is associated, and the goal is to find a solution with minimal cost. E.g., in propositional satisfiability (SAT), the set of candidates for an instance consists of all assignments of Boolean values to the variables. Local search algorithms start from an initial candidate and repeatedly replace the current candidate by a neighboring one of lower cost until such improvements are no longer possible. At this point we have a *locally optimal* candidate, and the algorithm terminates [8].

To obtain a *global* optimum, additional techniques are required. To name just two, [4] presents a local search algorithm WGSAT that uses an adaption scheme to change the neighborhood structure at runtime to avoid local optima. Another example are the Saw-ing evolutionary algorithms [3]. We will present a comparable adaption scheme for modal model generation.

We assume that the reader is familiar with the basics of propositional modal logic [2]. Models are structures of the form $(W, R_1, \ldots, R_n, V)$, where W is a non-empty set (the domain), each R_i is a binary relation on W (which may be required to satisfy additional properties), and V is a valuation, i.e., a function assigning subsets of W to proposition letters. We use the standard semantics: $M, w \models \Diamond_i \phi$ if there exists $v \in W$ with $R_i wv$ and $M, v \models \phi$.

Different modal logics may be obtained by imposing properties on the structure of models. The *model generation problem* for a modal logic **L** consists in finding, for a given formula ϕ, a structure $(W, R_1, \ldots, R_n)$ with the correct properties for **L** and building a model on $(W, R_1, \ldots, R_n)$ by defining a valuation V such that one of the states satisfies ϕ.

How can we use local search methods for model generation for modal logic? Modal models have propositional and modal aspects to them. We know how to use local search for propositional model generation—but how can we handle the modal aspects by means of local search? To simplify matters, we start by developing local search methods for the modal logic **S5**, whose models have a very regular structure. In **S5**, every formula ϕ can be rewritten to a formula ϕ^*, in **S5**CNF, such that ϕ^* has the following properties: (i) No two modal operators occur nested; that is, ϕ^* is of modal depth at most 1. (ii) Every path from the root to a leaf in the **S5**CNF formula tree passes through the connectives in the order of $\wedge$, $\vee$, $\Diamond/\Box$, $\wedge$, $\vee$, $\neg$, before it reaches a variable; some connectives may be skipped but the order is fixed. (iii) ϕ^* is **S5**-satisfiable if and only if ϕ is **S5**-satisfiable. (iv) The length of ϕ^* is polynomial in the length of ϕ.

Basically, a formula in **S5**CNF is a large conjunction of clauses, as in propositional CNF, but now the literals in such clauses can be modal formulas. Every literal of type $\Box$ or $\Diamond$ has a propositional CNF subformula. This similarity to CNF is necessary in order to adapt propositional local search to **S5**.

Our local search method for modal model generation works by generating a representation of a model, using a local search method and constraint weights that are adapted at runtime to avoid stagnation at a suboptimal solution. In

```
procedure main:
  input: formula $\phi$ in S5CNF.
  parse($\phi$)
  initialize
  while no solution found and generations < maximum
    mutate(M)   % create $\lambda$ neighboring candidate solutions $M_1, \ldots, M_\lambda$
    M := select($M_1, \ldots, M_\lambda$) % select best candidate from the sampled neighbors
    if update period expired adapt(W) end if
    if $M_{0,0} < B_{0,0}$, B := M end if  % replace if improvement occurred
    generation := generation +1
  end while
end main
```

Table 1. The main procedure, showing an initialization stage and the iteration of procedures **mutate**, **select**, and **adapt**.

Table 1 we give a high-level overview of our algorithm; in Sections 3 and 4 we provide further details of its core aspects.

3 Representing Candidate Solutions

We use mosaics to represent modal models in such a way that local search can act on them. Mosaics were introduced by [10] and have since been used in modal logic by e.g., [9]. Briefly, a mosaic is a small part of a model. By imposing the right saturation conditions on sets of mosaics, the existence of an **S5**-model becomes equivalent to the existence of a finite **S5**-saturated set of mosaics.

More formally now, let ϕ be a modal formula and let $Cl(\phi)$ be the closure under single negations of the set of subformulas of ϕ, i.e., the set of subformulas of ϕ such that whenever $\psi \in Cl(\phi)$ for some non-negated subformula ψ of ϕ then $\neg\psi \in Cl(\phi)$. Also, for each $\neg\Box\psi \in Cl(\phi)$ we have $\Diamond\neg\psi \in Cl(\phi)$. A *mosaic* μ is a subset of $Cl(\phi)$; it is *coherent* if it satisfies the following four constraints: (i) $\bot \notin \mu$; (ii) for every negated subformula $\neg\psi \in Cl(\phi)$: $\neg\psi \in \mu$ iff $\psi \notin \mu$; (iii) for every conjunction $\psi_1 \wedge \psi_2 \in Cl(\phi)$: $\psi_1 \wedge \psi_2 \in \mu$ iff $\psi_1 \in \mu$ and $\psi_2 \in \mu$; and (iv) for every $\Box\phi \in Cl(\phi)$: $\neg\Box\psi \in \mu$ iff $\Diamond\neg\psi \in \mu$.

Definition 1 (S5-Saturation). An **S5**-*saturated* set of mosaics is a set of mosaics $M = \{\mu_1, \ldots, \mu_m\}$ such that each mosaic $\mu_i \in M$ is coherent and the following constraints are satisfied:

- For every $\mu_i \in M$ and for every $\Diamond\psi \in Cl(\phi)$, if $\psi \in \mu_i$ then $\Diamond\psi \in \mu_i$.
- For every $\mu_i \in M$ and for every $\Diamond\psi \in Cl(\phi)$, if $\Diamond\psi \in \mu_i$ then there is some $\mu_j \in M$ such that $\{\psi\} \cup \{\theta \mid \Box\theta \in \mu_i\} \cup \{\Diamond\theta \mid \theta \in \mu_i\} \subseteq \mu_j$ *and* for every $\Diamond\chi \in Cl(\phi)$, $\{\Diamond\chi \mid \Diamond\chi \in \mu_j\} \subseteq \mu_i$.

If there exists an **S5**-saturated set of mosaics for ϕ, then ϕ is satisfiable in some reflexive, transitive and symmetric model. Conversely, if ϕ is satisfiable in a reflexive, transitive and symmetric model, then there exist an **S5**-saturated set of mosaics for ϕ of size at most 1+ the number of $\Diamond$'s in ϕ. As an immediate corollary we have that an **S5**-saturated set of mosaics corresponds to a model in which the accessibility relation is universal.

It may seem natural to represent a mosaic μ by a bit string $\mathtt{b}_\mu$ of length $|Cl(\phi)|$ in such a way that $\mathtt{b}_\mu(i) = 1$ if the i-th subformula of ϕ is in μ, and $\mathtt{b}_\mu(i) = 0$ if it is not. Instead of using the values 0 and 1 to represent set membership, however, a real number can be used to represent a *degree of constraint violation.* The value 0 represents no violation, any value > 2 represents the presence of some constraint violation, and larger values are seen as less desirable by the heuristic. A propositional formula ψ in the variables $p_1, \ldots, p_n$ is changed into a function $\tilde{\psi}$ of variables $\tilde{p}_1, \ldots, \tilde{p}_n$ taking values 0 for *true*, and 2 for *false.* E.g., the formula $(p_1 \vee p_2 \vee \neg p_3) \wedge (\neg p_1 \vee \neg p_2)$ becomes $\widetilde{p_1} \cdot \widetilde{p_2} \cdot (2 - \widetilde{p_3}) + (2 - \widetilde{p_1}) \cdot (2 - \widetilde{p_2})$.

Definition 2 (Candidate Solution). A *candidate solution* is a matrix $\mathsf{M}_{i,j}$ of non-negative real numbers holding the constraint violation values of the elements ψ_j in $Cl(\phi)$ in each mosaic μ_i in a given set of mosaics. We say $i \in sub(j)$ if the i-th element of $Cl(\phi)$ is a subformula of the j-th. Also, there is a matrix $\mathsf{W}_{i,j}$ of positive real numbers containing the weights of the elements ψ_j in $Cl(\phi)$ in each mosaic μ_i in the set. The row M_0 corresponds to the state in the model where ϕ is to be satisfied. If ϕ is true in M_0, i.e., if the constraint violation value of ϕ in M_0 is 0, then the candidate is an actual solution.

4 Evaluation and Adaption

Evaluation of candidate solutions is at the heart of all heuristic methods for automated problem solving. It is the key component of our algorithm that distinguishes it from a random walk in the space of candidate solutions. Every candidate is assigned a numerical value, its *fitness.* In propositional logic, the fitness of a candidate is determined by the assignment of truth values to the variables. For us, a candidate is determined by the assignments *in each state of the model.* Hence, we evaluate the fitness of a candidate by computing the truth value of propositional subformulas in every single mosaic, and by interpreting the modal subformulas as quantifying over the full set of mosaics:

$$\mathsf{M}_{i,j} = \begin{cases} 2 - \mathsf{M}_{i,sub(j)} & \text{if } type(j) = \neg; \\ \prod_h \mathsf{M}_{i,h} & \text{if } type(j) = \vee, \quad h \in sub(j); \\ \sum_h \mathsf{W}_{i,h} \cdot \mathsf{M}_{i,h} & \text{if } type(j) = \wedge, \quad h \in sub(j). \end{cases} \tag{1}$$

To evaluate **S5**-saturation we need a way to deal with the modal information present in the candidate and assign appropriate values to the entries of type $\Diamond$ and $\Box$ in M_0. Equation (2) reflects the quantificational properties of $\Diamond$ and $\Box$:

$$\mathsf{M}_{0,j} = \begin{cases} \min_k \{\mathsf{W}_{k,sub(j)} \cdot \mathsf{M}_{k,sub(j)}\}, & \text{if } type(j) = \Diamond; \\ \max_k \{\mathsf{W}_{k,sub(j)} \cdot \mathsf{M}_{k,sub(j)}\}, & \text{if } type(j) = \Box. \end{cases} \tag{2}$$

The maximum and minimum are taken over all mosaics μ_k in the set. Our algorithm searches for the right assignments to variables in the separate states and tries to minimize the amount of constraint violation for ϕ in μ_0 measured by these sums. The search ends when $\mathsf{M}_{0,0} = 0$. What is new here, is the interpretation of $\Diamond$ and $\Box$ as minima and maxima over the full set of mosaics.

Equations (1) and (2) enable the heuristic evaluation of candidate solutions. Moreover, the use of the matrix W allows us to adapt the very heuristic itself to the results of the search. If an entry in W has a large value, the corresponding subformula has a large influence on the overall fitness of the candidate. One avoids local optima by increasing the weights of subformulas that need to be satisfied but have remained violated for some period during the search.

Two questions arise when extending adaption schemes of propositional logic to modal logics. First, where do we assign weights in the formula graph? We have opted to assign weights to all subformulas corresponding to clauses, whether propositional or modal. Second, how do we adapt weights so as to enhance the search behavior? If a modal literal in M_0 needs to be satisfied, the search should try to satisfy the propositional constraints that the modal literal demands of the set of mosaics. Therefore, the constraints of such modal literals in M_0 need to be propagated to the rest of the mosaics, and the algorithm needs some control over this propagation. This control is provided by letting the adaption mechanism increase the weights of the propositional constraints that have the *lowest* violation value for a given modal literal. In this way, the search can perform both the modal and the propositional reasoning task simultaneously.

5 Preliminary Tests

We have implemented our algorithm and tested it on a large number of randomly generated **S5**CNF formulas. Our first aim was to compare the algorithm's search behavior with that of a random walk in order to test wether the adaptive scheme helps the search escape from local minima. The test formulas come from an ensemble whose parameters include the number of variables, modal clauses, propositional clauses under a $\Diamond$ or $\Box$, etc.

Using a random formula generator, we proceeded in two stages. First, we tried to find regions in the input space that contain hard (satisfiable) problems; we aimed to construct a well parameterized set (see [7]) that contains modal problems that are not trivially satisfiable. Secondly, in the interesting regions we generated a new set of problems. On these problems we examined the impact of weight adaption. Using a paired sample t-test we found that adaption does significantly improve search length on the problems in our test set. Our future tests will involve measuring other performance aspects such as success rate as well as comparisons with other model generation methods for **S5**.

6 Discussion: Beyond S5 and NP

We have seen the application of adaptive local search to model generation for the modal logic **S5**. There were two key factors to the success of our use of local search: the very simple structure of **S5**-models and the poly-sized model property (meaning that every satisfiable formula can be satisfied on a model whose size is bounded by a polynomial of the input). To which extent do these factors hamper or facilitate the extension of our work to our modal logics?

Let's look at other logics with the poly-size model property first. Recall that **S4.3** is the logic of reflexive transitive non-branching frames, and $\mathbf{K}_n\mathbf{Alt}_1$ is the logic of partial functions; both are NP-complete [2]. For **S5** and $\mathbf{K}_n\mathbf{Alt}_1$ we only need to consider a *single* frame when trying to generate a model for a given formula; that is, for **S5** and $\mathbf{K}_n\mathbf{Alt}_1$ the search for the correct frame (W, R) is easy and the model generator is mainly concerned with finding the correct valuation V. This *unique frame property* fails for **S4.3**. As a consequence, if we want to generalize Equations (1) and (2) to **S4.3**, the minima and maxima for $\Diamond$ and $\Box$ have to be taken over suitable subsets of the set of mosaics, which leads to a substantial complication of the search procedure.

What if we give up the poly-size model property, and move to logics like **K** and **S4**? Such logics are at the focus of our ongoing work. Model generation for these logics requires a mixture of efficient representations and implicit ways of manipulating models. Bounding the size of candidate solutions and slowly incrementing their sizes during the search is essential so as to avoid the use of very large structures whenever possible. The strategy advocated by [6] is particularly relevant here; it solves modal satisfiability problems by alternating propositional and modal steps. In the propositional step one views modal formulas as propositional formulas with "modal atoms." Once a propositional model has been found, the "modal atoms" are unpacked by peeling off a single $\Diamond$ or $\Box$, and then doing a purely propositional step again. We propose to use local search for the propositional step, and are exploring ways of using local search for the intermediate modal steps.

References

1. C. Areces, E. Franconi, R. Goré, M. de Rijke, and H. Schlingloff. Use your logic. *Logic Journal of the IGPL*, 8:231–237, 2000.
2. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* CUP, 2001.
3. A. E. Eiben and J. van Hemert. Saw-ing EAs: Adapting the fitness function for solving constraint problems. In D. Corne et al., editor, *New Methods in Optimisation, Advanced Topics in Computing*, pages 389–402. McGraw-Hill, 1999.
4. J. Frank. Weighting for Godot, learning heuristics for GSAT. In *Proceedings AAAI-96*, pages 338–343, 1996.
5. I. Gent, H. van Maaren, and T. Walsh, editors. *SAT2000.* IOS Press, 2000.
6. E. Giunchiglia and R. Sebastiani. Building decision procedures for modal logics from propositional decision procedures. In *CADE-13*, 1996.
7. I. Horrocks, P. Patel-Schneider, and R. Sebastiani. An analysis of empirical testing for modal decision procedures. *Logic Journal of the IGPL*, 8(3):293–323, 2000.
8. D.S. Johnson, C.H. Papadimitrou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37:79–100, 1988.
9. M. Marx, Sz. Mikulás, and M. Reynolds. Temporal mosaics. Division of business, information technology and law research working paper, Murdoch University, 1999.
10. I. Németi. Decidable versions of first order logic and cylindric-relativized set algebras. In *Logic Colloquium '92*, pages 171–241, 1995.