

# Solving Graded/Probabilistic Modal Logic via Linear Inequalities (System Description)

William Snell, Dirk Pattinson, and Florian Widmann

Dept. of Computing, Imperial College London

**Abstract.** We present the experience gained from implementing a new decision procedure for both graded and probabilistic modal logic. While our approach uses standard tableaux for propositional connectives, modal rules are given by linear constraints on the arguments of operators. The implementation uses binary decision diagrams for propositional connectives and a linear programming library for the modal rules. We compare our implementation, for graded modal logic, with other tools, showing average performance. Due to lack of other implementations, no comparison is provided for probabilistic modal logic, the main new feature of our implementation.

## Introduction

Both graded modal logic [9] and probabilistic modal logic [8] extend (classical) propositional logic by modal connectives that constrain the number of successor states in a Kripke frame or the probability of events in a Markov chain.

Graded modalities are used in description logic to represent cardinality restrictions in (extensions of) the logic  $\mathcal{ALCQ}$  [1], which is supported by a large number of tools such as Pellet [19], Fact++ [20], RACER [11] or Hermit [15]. Probabilities are not used in description logic (and not supported by description logic reasoners), but most formalisms for describing the behaviour of probabilistic systems [16, 5] rely on probabilistic modal logic in some form or other. While dedicated model checkers such as Prism [12] support the analysis of probabilistic systems, there is currently no tool support for *reasoning* in probabilistic (modal) logics.

This paper is a first step towards filling this gap: we present the experience gained from implementing a satisfiability checker that supports both graded and probabilistic modal logic in the same framework. The calculus that we implement uses standard tableau rules together with linear inequalities to describe the arithmetic or probabilistic constraints on successor states/events. The distinguishing feature of this calculus (see [18] for the dual treatment of validity) is that it does *not* try to construct a model. Instead, it produces linear constraints whose (un-)solvability guarantees the *existence* of a model. The calculus proceeds in two steps: in the first step, propositional connectives are eliminated in a standard way, leading to conjunctive clauses over modal literals. The modal rules applied subsequently are given in terms of *linear inequalities* over the arguments of the operators in the premise. Every solution (together with a side condition) represents a modal rule. The structural similarity of the rules for graded and probabilistic modal logic allows us to treat both logics almost identically.

The implementation of this calculus naturally reflects both steps. We represent modal formulae using binary decision diagrams [2] where modal atoms are represented by (BDD) variables. Propositional tableau rules then correspond to analysing all satisfying assignments of the corresponding BDD. In a second step, we calculate the set of all modal rules applicable to a set of modal literals where solvability of an associated linear programming problem is equivalent to validity of the corresponding tableau rule. Compared with other tools, our implementation of graded modal logic shows average performance, but allows for the treatment of probabilistic modalities in the same framework, something that is not currently supported by other tools.

To eliminate artefacts such as the particular choice of blocking or caching techniques used, we focus on the satisfiability problem of both logics over empty TBoxes. We describe syntax and semantics of both graded and probabilistic modal logic and briefly introduce the underlying calculus. We then discuss implementation details and present a brief comparison with other tools.

*Related Work.* Linear inequalities have been used for graded modal logic in [7] as an addition to a tableau calculus where they are used (along with the usual completion rules) to detect inconsistency in ABoxes. Our approach, in contrast, uses linear inequalities directly to determine valid completion steps (tableau rules). A proof-of-concept implementation demonstrating compositionality of modal logics, containing graded and probabilistic modal logic as building blocks, was presented in [3] using a brute-force approach that was orders of magnitude slower than other tools.

## 1 Syntax, Semantics and Tableau Calculus

Both graded modal logic and probabilistic modal logic are extensions of propositional logic by unary modal operators, given by the grammar

$$\mathcal{L} \ni \phi, \psi ::= p \mid \phi \wedge \psi \mid \neg\phi \mid \langle x \rangle \phi$$

where  $p \in \mathcal{V}$  is a propositional variable and where  $x \in \mathbb{N}$  in the case of graded modal logic and  $x \in [0, 1] \cap \mathbb{Q}$  for probabilistic modal logic.

For graded modal logic, we read  $\langle n \rangle \phi$  as ‘ $\phi$  holds in *more than*  $n$  successor worlds, and the probabilistic operator  $\langle p \rangle \phi$  stipulates that ‘ $\phi$  holds with probability  $\geq p$  in the next state’. We interpret graded modal logic over *multigraph frames* [4]  $(W, \sigma, \pi)$  where  $W$  is a set of worlds,  $\sigma : W \rightarrow \mathcal{B}(W)$  assigns a finite multiset (a ‘bag’, formally a function  $W \rightarrow \mathbb{N}$  with finite support) of elements of  $W$  to every world  $w$  and  $\pi : W \rightarrow \mathcal{P}(\mathcal{V})$  is a valuation of the propositional variables. Truth of a formula at a point is defined in the standard way, with the clause on the left below for the modal operators.

$$w \models \langle n \rangle \phi \iff \sum_{w' \models \phi} \sigma(w)(w') > n \qquad w \models \langle p \rangle \phi \iff \sum_{w' \models \phi} \sigma(w)(w') \geq p$$

Similarly, probabilistic modal logic is interpreted over discrete Markov chains  $(W, \sigma, \pi)$  where  $W$  and  $\pi$  are as above, and  $\sigma$  assigns a finitely supported probability distribution (formally a function  $\mu : W \rightarrow [0, 1]$  with finite support such that  $\sum_{w \in W} \mu(w) = 1$ ) to every world  $w$ . For modal operators, truth is given by the clause on the right above, together with the standard clauses for the remaining propositional connectives.

While graded modal logic is usually interpreted over Kripke frames (rather than multigraphs), it is easy to see that both semantics give rise to the same satisfiability problem. We obtain truth-preserving translations by considering a Kripke frame as a multigraph where every edge has multiplicity one, and multigraphs can be converted to Kripke frames by inserting the appropriate number of copies of each successor state. Similarly, the interpretation of probabilistic modal logic over non-discrete Markov chains induces the same satisfiability problem as the semantics presented here.

Our focus in this paper is the *satisfiability problem* for graded and probabilistic modal logic, and as usual, a formula  $\phi \in \mathcal{L}$  is *satisfiable* iff there exists a model  $(W, \sigma, \pi)$  and a world  $w \in W$  such that  $w \models \phi$ .

Our implementation is based on the following characterisation of satisfiability in terms of a tableau calculus over *tableau sequents*, that is, finite sets of formulae that we read conjunctively. We have the standard rules for propositional connectives

$$\frac{\Gamma, p, \neg p}{\Gamma, A \wedge B} \quad \frac{\Gamma, A \wedge B}{\Gamma, A, B} \quad \frac{\Gamma, \neg(A \wedge B)}{\Gamma, \neg A \quad \Gamma, \neg B} \quad \frac{\Gamma, \neg \neg A}{\Gamma, A} \quad \frac{\langle x_0 \rangle p_0, \dots, \langle x_n \rangle p_n, \neg \langle y_0 \rangle q_0, \dots, \neg \langle y_m \rangle q_m}{\sum_{i=0}^n r_i p_i - \sum_{j=0}^m s_j q_j > k}$$

together with all substitution instances of the rule on the right for graded and probabilistic modal logic, subject to (different) side conditions, depending on whether we deal with graded or probabilistic modal logic. For graded modal logic, we have  $k = 0$  and require the side condition on the left below

$$\sum_{1 \leq i \leq n} r_i(x_i + 1) - \sum_{1 \leq j \leq m} s_j y_j \geq 1 \quad \sum_{1 \leq i \leq n} r_i x_i - \sum_{1 \leq j \leq m} s_j y_j \triangleright k$$

while for probabilistic modal logic, we have  $k \in \mathbb{Z}$  and require the side condition on the right above, where  $\triangleright = \geq$  if  $m > 0$  and  $\triangleright = =$  if  $m = 0$ . For both logics,  $r_i, s_j \in \mathbb{N} \setminus \{0\}$ .

The linear inequality in the rule conclusions is a compact shorthand, and denotes a (possibly empty) set of conclusions. More precisely, we associate to each function  $v : \{p_1, \dots, p_n, q_1, \dots, q_m\} \rightarrow \{0, 1\}$  a propositional sequent  $s(v)$  containing  $p_i$  if  $v(p_i) = 1$  and  $\neg p_i$  if  $s(p_i) = 0$ , analogously for  $q_j$ . The inequality  $\sum_{i=0}^n r_i p_i - \sum_{j=0}^m s_j q_j > k$  then stands for the set of conclusions containing all  $s(v)$  for which  $\sum_{i=0}^n r_i v(p_i) - \sum_{j=0}^m s_j v(q_j) > k$ . That is, the conclusion encodes the set of binary valuations, seen as a tableau sequent, for which the inequality holds.

The calculus above is sound and complete for both graded and probabilistic modal logic [18, 14]: a sequent  $\Gamma$  is (conjunctively) unsatisfiable iff there exists a closed tableau with root  $\Gamma$ . In particular, *every* solution of the side condition in terms of weights  $r_j$  and  $s_j$  induces a modal rule, but one can establish a polynomial bound on the weights retaining completeness of the calculus [18, Lemma 6.16].

## 2 Implementation Details

We describe the representation of formulae in terms of binary decision diagrams, the representation of tableau rules and the reasoning algorithm.

**Representation of Formulae.** We represent formulae of both graded and probabilistic modal logic using binary decision diagrams [?], where propositional variables and modal atoms (that is, formulae of the form  $\langle x \rangle \phi$ ) are represented using BDD variables. A tableau sequent is represented by the conjunction of the formulae it contains. This allows us to effectively delegate propositional reasoning to binary decision diagrams: every satisfying valuation of (the BDD representing) a tableau sequent  $\Gamma$  corresponds to a leaf of a propositional tableau with root  $\Gamma$ . We maintain a look-up table to ensure that multiple occurrences of the same propositional variable are represented by the same BDD variable. In particular, modal atoms that have propositionally equivalent arguments are presented by the same BDD variable. We note two consequences.

**Deep Congruence.** We call two formulae *congruence equivalent* if their equivalence can be established by propositional reasoning and the congruence rule  $\phi \leftrightarrow \psi / \heartsuit \phi \leftrightarrow \heartsuit \psi$  where  $\heartsuit$  is a modal operator. As modal atoms with propositionally equivalent arguments are represented by the *same* BDD variable, congruence-equivalent formulae are represented by the same BDD.

**Semantic Branching.** The implementation of propositional tableau rules by means of computing satisfying assignments of the associated BDD implies an implicit use of semantic branching. This is a consequence of computing satisfying assignments using BDDs, where the set of satisfying assignments corresponds to the set of paths from the root node to the leaf representing logical truth. For example, the set of satisfying assignments of the (BDD encoding of the) formula  $p_0 \vee p_1 \vee p_2$  will be  $p_0$ ,  $\neg p_0 \wedge p_1$  and  $\neg p_0 \wedge \neg p_1 \wedge p_2$  in the variable order  $p_0 < p_1 < p_2$ .

While BDDs conveniently relieve us of the task of implementing the rules for propositional reasoning, one obvious disadvantage of using BDDs as a 'black box' is the impossibility of implementing other optimisations, in particular backjumping [13].

**Application of Modal Rules.** Given a sequent  $\Gamma$  consisting of possibly negated modal atoms and propositional variables, we compute the set of modal rules applicable to  $\Gamma$  on the fly. While every solution of the side condition gives rise to an instance of the rule, we can compute a polynomial bound on the size of the search space: Lemma 17.1b in [17] guarantees that all possible rules are generated if we limit the weights to non-negative integers of (binary) size  $\leq 6n^3w$  where  $w$  is the size of the side condition and  $n$  is the number of modal literals. As numbers are represented in binary, the search space to be explored is of size  $2^{6n^4w}$  (as every of the  $n$  coefficients may vary between 0 and  $2^{6n^3w}$ ). This bound is exponential unlike the doubly exponential bound  $2^{2^n}$  on the number of possible rules applicable to a sequent containing  $n$  modal literals. In practice, the doubly exponential bound  $2^{2^n}$  will be below the size of the search space computed above for sequents containing less than approximately 34 modal literals, which appears to be enough to cover all practical applications. We therefore implement rule generation using a doubly exponential algorithm. Given a sequent  $\Gamma$  consisting of  $n$  positive modal atoms of the form  $\langle x \rangle p_i$  with arguments  $p_1, \dots, p_n$  and  $m$  negated modal atoms  $\neg \langle x \rangle q_j$  with arguments  $q_1, \dots, q_m$ , we iteratively check for all propositional formulae  $\phi$  in variables  $p_1, \dots, p_n, q_1, \dots, q_m$  whether  $\Gamma / \text{dnf}(\phi)$  is a valid rule by encoding both the side condition and the conclusion  $\phi$  into a system of linear inequalities. (Here  $\text{dnf}(\phi)$  denotes the disjunctive normal form of  $\phi$  in the form of a set of tableau sequents.) In more detail,  $\Gamma / \text{dnf}(\phi)$  is a rule of graded / probabilistic modal logic if the system of

linear inequalities consisting of the side condition together with the inequalities

$$\begin{cases} \sum_{i=1}^n r_i v(p_i) - \sum_{j=1}^m s_j v(q_j) > k & \text{if } \phi \text{ evaluates to } \top \text{ under } v \\ \sum_{i=1}^n r_i v(p_i) - \sum_{j=1}^m s_j v(q_j) \leq k & \text{otherwise,} \end{cases}$$

where  $v$  ranges over all valuations  $\{p_1, \dots, p_n, q_1, \dots, q_m\} \rightarrow \{0, 1\}$ , has a solution. We use an external library [?] to determine whether systems of linear inequalities have a solution. This (doubly exponential) search space can be pruned significantly.

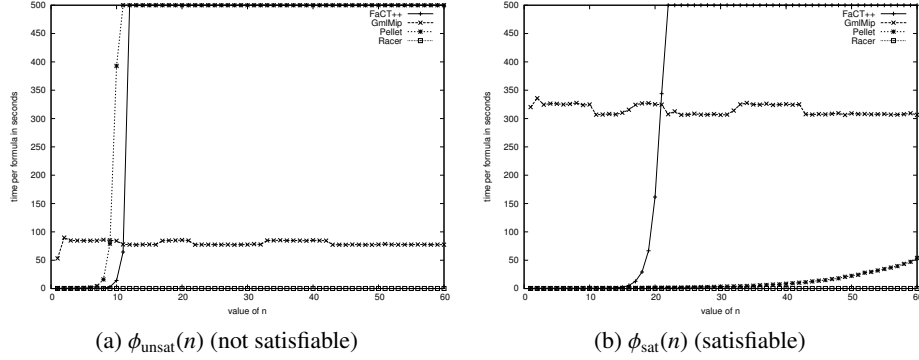
*Rule Subsumption.* We say that a modal rule  $\Gamma/\text{dnf}(\phi)$  *subsumes* the rule  $\Gamma/\text{dnf}(\psi)$  if  $\phi \rightarrow \psi$  is propositional tautology. Intuitively, the conclusion  $\phi$  is ‘harder’ to satisfy, so that any application of  $\Gamma/\text{dnf}(\psi)$  in a closed tableau can be replaced by an application of  $\Gamma/\text{dnf}(\phi)$ . In other words, omitting the rule  $\Gamma/\text{dnf}(\psi)$  does not jeopardise completeness of the calculus. Given a sequent  $\Gamma$  consisting of positive atoms  $\langle x \rangle p_i$  with arguments  $p_1, \dots, p_n$  and negative modal atoms  $\neg \langle x \rangle q_j$  with arguments  $q_1, \dots, q_m$ , we arrange the set of propositional formulae in variables  $\{p_1, \dots, p_n, q_1, \dots, q_m\}$  in a directed, rooted, acyclic graph where the edge relation represents subsumption. Let  $\Theta = \{v_1, \dots, v_{2^{n+m}}\}$  be the set of all possible valuations of the propositional variables present. A node in the subsumption graph contains a subset  $N \subseteq \Theta$  of valuations that represent the propositional formula  $\phi_N = \bigvee_{v \in \Theta} \bigwedge_{v(a)=1} a \wedge \bigwedge_{v(a)=0} \neg a$  where  $a$  ranges over the set  $\{p_1, \dots, p_n, q_1, \dots, q_m\}$  of propositional variables. Two (different) nodes  $N$  and  $M$  are related by a direct edge  $N \rightarrow M$  if  $M = N \cup \{v_j\}$  and  $j > \max\{1 \leq i \leq 2^{n+m} \mid v_i \in N\}$ , and the root of the subsumption graph is the empty set. This arrangement guarantees that a rule with conclusion  $\phi_N$  subsumes all rules with conclusions  $\phi_M$  provided that  $N$  is a (direct or indirect) ancestor of  $M$ . We generate rule conclusions using *breadth first* search through the subsumption tree, pruning all children of a node  $N$  as soon as it has been established that  $\Gamma/\text{dnf}(N)$  is a valid rule instance.

*Reasoning Algorithm.* Completeness of the tableau calculus allows us to reduce satisfiability of the root formula to the non-existence of a closed tableau. We check for the existence of a closed tableau using standard depth-first search. At the moment, we have not implemented any optimisations such global caching or unsat caching [6, 10].

### 3 Experimental Evaluation

As mentioned already, we are not aware of any other reasoner that supports probabilistic modal logic. We therefore restrict ourselves to graded modal logic in this section. We do not claim to conducting a thorough system comparison, but merely want to gauge whether the approach described in the previous section is feasible. In particular, we only consider satisfiability over the empty TBox to factor out artefacts like the particular choice of caching or blocking techniques used and concentrate on formulae whose main ‘complexity’ is due to graded modalities. All formulae, and our tool, are available at <http://www.doc.ic.ac.uk/~dirk/Software/GmlMiP/>.

We compare our prover (GmlMiP) with Fact++ [20], Pellet [19], and RACER [11]. The benchmarks were run on a 64 bit Linux system with an Intel Core i5-650 CPU and a memory limit of 4 GB, and out of memory errors were treated as timeouts.



**Fig. 1.** Increasing the cardinality

**Table 1.** Randomly generated formulae

No. of timeouts	n = 10	20	30	40	50	60	70	80	90	100
Fact	1	0	2	4	4	5	3	6	6	8
GmlMip	0	1	5	20	23	40	33	55	61	72
Pellet	3	13	31	30	41	46	42	60	57	71
Racer	0	0	0	0	0	0	0	0	0	0

The first two sets of benchmarks show how the provers are affected by increasing numbers in cardinality constraints. The formulae are of the form

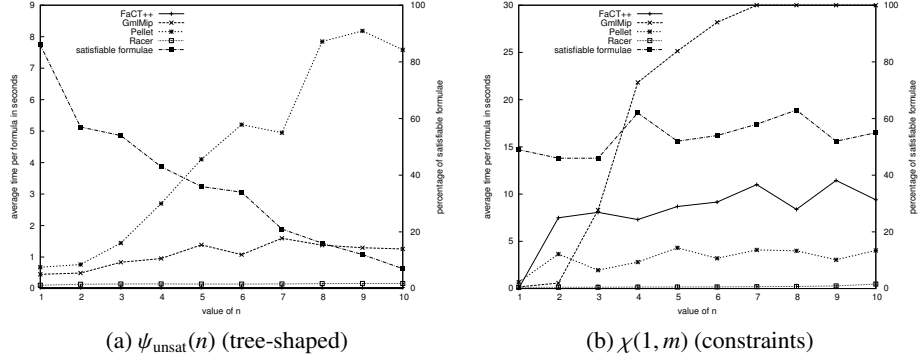
$$\phi_x(n) = \langle n-1 \rangle \neg p_1 \wedge \langle n-1 \rangle p_1 \wedge \neg \langle n \rangle p_0 \wedge \neg \langle m_x \rangle \neg p_0$$

where  $x \in \{\text{unsat}, \text{sat}\}$  and  $m_{\text{unsat}} = n-1$  and  $m_{\text{sat}} = n$ . Figure 1a shows the results for  $\phi_{\text{unsat}}(n)$  for  $1 \leq n \leq 60$  and a timeout of 500 seconds. Since Fact++ and Pellet appear to treat number restrictions naively, they fail for larger instances, whereas GmlMip and Racer are not affected by the cardinality. In the satisfiable case, see Fig. 1b, GmlMip takes longer because all modal rule applications have to be constructed, whereas Pellet behaves much better, presumably because it can find a model fast.

As a sanity check, we also used randomly generated formulae without any structure. Despite not being overly informative, it is useful for testing provers for discrepancies. (Indeed, one bug in Fact++ was discovered which has been fixed prior to our benchmarks.) The size of these formulae, denoted by  $n$ , is the number of symbols, counting 1 for cardinalities. The formulae were created by randomly choosing a connective with equal probability and recursively creating the subformulae. Cardinalities were chosen randomly between 0 and 99 inclusive. We tested 100 formulae for each size with a timeout of 30 seconds. Table 1 shows the number of timeouts relative to the size ( $n$ ) of formulae, as this is more informative than average runtime.

The next benchmark formulae enforce tree shaped models and are of the form

$$\psi(0) = \top \quad \text{and} \quad \psi(n) = \langle c_1^n - 1 \rangle \top \wedge \neg \langle c_2^n \rangle \top \wedge \neg \langle 0 \rangle \neg \psi(n-1)$$



**Fig. 2.** Tree-shaped models and simple constraint problems

**Table 2.**  $\chi(2, 2)$  (constraints)

	Fact	GmlMip	Pellet	Racer
Number of timeouts	15	0	7	0
Average time per formula	123s	79s	80s	<1s

where  $1 \leq c_1^n, c_2^n \leq 100$  are drawn randomly so that  $c_1^n \leq c_2^n$  with probability 0.8. For each  $1 \leq n \leq 10$  we generated 100 formulae and the average time (per  $n$ ) is plotted in Fig. 2a. The timeout was 30 seconds which was only exceeded by Pellet occasionally. The figure also shows the percentage of satisfiable formulae which naturally drops as the depth of the trees increases.

The last two sets of benchmarks mimic simple constraint problems of the form

$$\chi(n, m) = \bigwedge_{i=1}^n \langle c_i \rangle (\text{wish}_i \wedge \text{disj}_i) \wedge \bigwedge_{j=1}^m \neg \langle d_j \rangle r_j$$

where  $0 \leq c_i, d_j < 100$  are drawn randomly and  $\text{wish}_i$  is a simple randomly generated propositional formula in the variables  $r_j$  and  $\text{disj}_i$  is a (fixed) propositional formula not containing  $r_j$  such that  $\text{disj}_i \wedge \text{disj}_{i'}$  is unsatisfiable for  $i \neq i'$ : if two children want ice cream ( $r_0$ ) and three want ice cream or an apple ( $r_1$ ), and you have two ice creams and three apples, we get the formula  $\langle 1 \rangle (r_0 \wedge q) \wedge \langle 2 \rangle ((r_0 \vee r_1) \wedge \neg q) \wedge \neg \langle 2 \rangle r_0 \wedge \neg \langle 3 \rangle r_1$ . Figure 2b shows the results for  $n = 1$  fixed. For each  $m$  we tested 100 formulae, again with a timeout of 30 seconds. Unfortunately, GmlMip cannot compete for large  $m$  as the performance of the linear solver degrades when the number of modal formulae in a sequent becomes too big. We also tested 30 formulae of the form  $\chi(2, 2)$  with a timeout of 300 seconds and the results are given in Table 2.

## 4 Conclusion

While our implementation of graded modal logic was not quantitatively better in comparison to other tools, our implementation also supports probabilistic modal logic. The

experimental results indicate that the method itself can be made competitive. The structural similarity between graded and probabilistic modal logic insinuates that this also applies to probabilistic modal logic, and our tests with probabilistic modal logic formulae show roughly equivalent performance (comparing formula size).

## References

1. F. Baader and W. Nutt. Basic description logics. In F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors, *Description Logic Handbook*, pages 43–95. Cambridge University Press, 2003.
2. R. E. Bryant. Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams. *ACM Computing Surveys*, 24(3):293–317, 1992.
3. G. Calin, R. Myers, D. Pattinson, and L. Schröder. COLOSS: The coalgebraic logic satisfiability solver. In *Proc. Methods for Modalities 5 (2007)*, volume 231 of *ENTCS*, pages 41–54, 2009.
4. G. D’Agostino and A. Visser. Finality regained: A coalgebraic study of Scott-sets and multisets. *Arch. Math. Logic*, 41:267–298, 2002.
5. J. Desharnais, A. Edalat, and P. Panangaden. Bisimulation for labelled markov processes. *Inf. Comput.*, 179(2):163–193, 2002.
6. F. Donini and F. Massacci. EXPTIME tableaux for ALC. *Artif. Intell.*, 124(1):87–138, 2000.
7. J. Faddoul and V. Haarslev. Algebraic tableau reasoning for the description logic SHOQ. *J. Applied Logic*, 8(4):334–355, 2010.
8. R. Fagin and J. Halpern. Reasoning about knowledge and probability. *J. ACM*, 41:340–367, 1994.
9. K. Fine. In so many possible worlds. *Notre Dame J. Formal Logic*, 13:516–520, 1972.
10. R. Goré and L. Nguyen. EXPTIME tableaux for ALC using sound global caching. In D. Calvanese, E. Franconi, V. Haarslev, D. Lembo, B. Motik, A.-Y. Turhan, and S. Tessaris, editors, *Proc. Description Logics 2007*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.
11. V. Haarslev and R. Miller. RACER system description. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Proc. IJCAR 2001*, volume 2083 of *LNCS*, pages 701–705. Springer, 2001.
12. A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In H. Hermanns and J. Palsberg, editors, *Proc. TACAS 2006*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.
13. I. Horrocks and P. Patel-Schneider. Optimising description logic subsumption. *J. Logic Comput.*, 9:267–293, 1999.
14. C. Kupke and D. Pattinson. On modal logics of linear inequalities. In V. Goranko and V. Shehtman, editors, *Proc. AiML 2010*. College Publications, 2010.
15. B. Motik, R. Shearer, and I. Horrocks. Hypertableau Reasoning for Description Logics. *Journal of Artificial Intelligence Research*, 36:165–228, 2009.
16. A. Parma and R. Segala. Logical characterizations of bisimulations for discrete probabilistic systems. In H. Seidl, editor, *Proc. FoSSaCS 2007*, volume 4423 of *LNCS*, pages 287–301. Springer, 2007.
17. A. Schrijver. *Theory of linear and integer programming*. Wiley Interscience, 1986.
18. L. Schröder and D. Pattinson. PSPACE bounds for rank-1 modal logics. *ACM Transactions on Computational Logics*, 10(2), 2009.
19. E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 2006. to appear.
20. D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. IJCAR 2006*, volume 4130 of *LNAI*, pages 292–297. Springer, 2006.