Efficient Reasoning with Labelled Formula Translations of K_m

Jens Happe, School of Computing Science, Simon Fraser University
e-mail: jhappe@cs.sfu.ca

Simon Fraser University Centre for systems science

SFU RESEARCH MATTERS

Problem Description

Foundations

Modal Logic $\mathbf{K}_m$: Propositional logic augmented with unary operators for possibility ($\langle R_i \rangle$), and necessity ($[R_i]$).
Decidable, PSPACE-complete (with EXP-time complete extensions, see Future Work).
Equivalent to Description Logic $\mathcal{ALC}$, which is:

- Applied in real-world knowledge bases, e.g. GALEN [Rector, Nowlan, & Glowinski 1993].
- Foundation for semantic web representation Language: DAML+OIL [Klein et al. 2003]
- Suitable for a wide range of modelling and verification tasks (UML)
- Fast reasoners exist [Haarslev & Möller 2003]: FaCT, DLP, RACER
- Experimental prototypes: ModProf[Happe 2001a], *SAT[Giunchiglia & Tacchella, 2000]

Semantic of $\mathbf{K}_m$: given by means of Kripke models (see Example to the right).
Most implementations use a refinement of the tableau method.

Why Reasoning is intractable

(1) Diamond operators can result in exponentially growing numbers of worlds:

$$\Phi_1 = \{\langle R_i \rangle p_1, \langle R_i \rangle \langle R_i \rangle p_2, [R_i] \langle R_i \rangle \neg p_2, [R_i][R_i] \langle R_i \rangle p_3, [R_i] [R_i] \langle R_i \rangle \neg p_3, \ldots\}$$

(2) Disjunctions within necessities make matters much worse:

$$\Phi_2 = \Phi_1 \cup \{[R_i] \ldots [R_i]\, p_1 \vee p_2 \vee p_3 \vee \ldots\}$$

☛ involves branching over exponentially many worlds.
This has been identified as the major bottleneck in real-world applications as well [Horrocks & Tobies 2000].

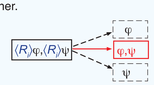
Existing Approaches

1. Caching and Model Merging [Horrocks 2003]

Assume a set of formulas Φ has been previously encountered and found satisfiable, and a new set $\Psi \subseteq \Phi$ is encountered.
Then Ψ can be declared satisfiable without any further inspection.
Conversely, if Φ has been previously found unsatisfiable, and $\Phi \subseteq \Psi$, then Ψ can be declared unsatisfiable without any further inspection.
☛ An efficient implementation of this is [Giunchiglia & Tacchella, 2000]. A more elaborate version of caching has been presented in [Happe 2001b].
Conceptually, caching produces non-tree Kripke models by grouping nodes together.

Model Merging also utilizes caching:

- Assume a node has two associated formulas $\langle R \rangle \varphi$ and $\langle R \rangle \psi$, and models for both φ and ψ exist, and their variable assignments are compatible.
  Then the worlds for φ and ψ can be merged into one successor world.



Caching and model merging fail when formulas are mutually inconsistent, as in the examples Φ_1,Φ_2 above.

2. Labelling [Beckert & Goré 2001]

While branching on a formula, assign labels to the subformula branched upon, capturing the modalities. Example:

$$[R_i][R_j]\langle R_k \rangle p \;\rightarrow\; 1.i.x.j.y.k.c.p.$$

Ensures completeness by duplicating ν-formulas, and by classifying variables as universal or free.
Decides satisfiability, but does not return satisfying models.
Our formalism is based on these ideas, but is more powerful and returns satisfying models.

3. Functional Translation [Ohlbach et al. 2001]

Formulas get translated into a decidable fragment of FOL. The translated formula is solved using a first-order theorem prover. Example:

$$\langle R_i \rangle p_1 \wedge [R_i]\langle R_j \rangle p_2 \wedge [R_i](\neg p_1 \vee [R_i] \neg p_2) \;\rightarrow\; p_1(c_i),\, p_2(x_i d),\, \neg p_1(y_i) \vee \neg p_2(y_i z_i)$$

Relies on the efficiency of FOL Provers; no specific optimization for modal formulas.
Today's most efficient FOL provers use resolution, not tableaux.
Dead-end predicates needed to cope with the trivial solution of $[R_i]\bot$.
Overall efficiency comparable to optimized tableau-based provers.

4. Model Evolution Calculus [Baumgartner & Tinelli 2003]

A similar method to Labelled Formulas, but for FOL.
Provides a mechanism similar to labels with exceptions, but less powerful.
Thus, Functional Translation plus Model Evolution Calculus cannot simulate our approach.

Solution

$\Phi = \{[R_1]\,(p_1 \vee q_1),\ [R_1]\,(p_2 \vee q_2),\ \langle R_1 \rangle(\neg p_1 \wedge \neg q_2),\ \langle R_1 \rangle(\neg q_1 \wedge \neg p_2)\}$

original Formula

Complexity: linear

$*_1\,(p_1 \vee q_1),\ *_1\,(p_2 \vee q_2),\ c_1\,(\neg p_1 \wedge \neg q_2),\ c'_1\,(\neg q_1 \wedge \neg p_2)$

labelled Form

Complexity: length∗depth

$*_1\,(p_1 \vee q_1),\ *_1\,(p_2 \vee q_2),\ c_1\,\neg p_1,\ c_1\,\neg q_2,\ c'_1\,\neg q_1,\ c'_1\,\neg p_2$

labelled Form (And/Or NF)

Branch on $*_1\,(p_1 \vee q_1)$: $*_1\,p_1$
→ Clashes with: $c_1\,\neg p_1$
Create exception: $(*_1,\{c_R\})\ p_1$
Re-branch on $c_1\,(p_1 \vee q_1)$: $c_1\,q_1$

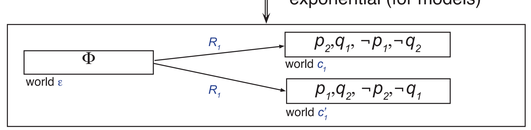
Branch on $*_1\,(p_2 \vee q_2)$: $*_1\,p_2$
→ Clashes with: $c'_1\,\neg p_2$
Create exception: $(*_1,\{c'_1\})\,p_2$
Re-branch on $c'_1\,(p_2 \vee q_2)$: $c'_1\,q_2$

Complexity: PSPACE-complete (for sat.) exponential (for models)

$(*_1,\{c_1\})\ p_1,\ c_1\,q_1,\ (*_1,\{c'_1\})\,p_2,\ c'_1\,q_2,\ c_1\,\neg p_1,\ c_1\,\neg q_2,\ c'_1\,\neg q_1,\ c'_1\,\neg p_2$

labelled Model (assertions)

Complexity: co-NP complete (for consistency) exponential (for models)



Kripke Model

Labels are strings of constants $c_i, c'_j, \ldots$ from countable domains D_i, and anonymous variables (wildcards) $*_i$, for each relation R_i.
The empty label is denoted ε.
Labels replace chains of modal operators in front of a formula and any of its subformulas as follows:

- $[R_i]$ is replaced by $*_i$.
- Each $\langle R_i \rangle$ is replaced by a new constant from D_i.

A wildcard $*_i$ can be instantiated by any constant from domain D_i, or by itself.
Intuitively, a label with wildcards represents the set of all its ground instances within a given context.
In this example: $*_1$ represents $\{c_1, c'_1\}$.
The mgu (σ,σ') of two labels σ and σ' is the most general common instance of σ and σ'.

In labelled formulas, labels distribute over conjunctions.
☛ Thus, $c_i\,(\neg p_1 \wedge \neg q_2)$ is equivalent to $c_i\,\neg p_1 \wedge c_i\,\neg q_2$.
Notice: In $\mathbf{K}_m$, this is not true: $\langle R_i \rangle(\neg p_1 \wedge \neg q_2)$ and $\langle R_i \rangle \neg p_1 \wedge \langle R_i \rangle \neg q_2$ are not equivalent.
☛ A formula can be converted to And/Or Normal Form,by:

- pulling labels inside conjunctions
- flattening nested conjunctions and disjunctions.

Just as in the tableau method, we seek to eliminate disjunctions by branching.
However, instead of ground labels, we branch universally over the label which precedes the disjunction (here: $*_1$).
(This would be unsound, were it not for our method of repairing clashes, see below.)

Two labelled literals form a clash, if their labels have common ground instances (clash witnesses).
Likewise, a formula $\sigma\,\bot$ forms a clash, if σ has a ground instance.
Clashes indicate elementary inconsistencies.

A label with exceptions is denoted as (σ,Σ), where σ is a simple label, and Σ is a set of instances of σ (exceptions).
Intuitively, (σ,Σ) represents the set of all ground instances of σ which are not (also) instances of any exception in Σ.
Notice: $(*_1,\{c_1\})\ p_1$ and $c_1\,\neg p_1$ do not form a clash. Hence, the former inconsistency has been repaired.

An assertion is an atomic formula ($p, \neg p$, T, or $\bot$) with a label (possibly with exceptions).
For a set M of assertions and a label σ, we define $M|_\sigma = \{\sigma'' \; a \mid \sigma\sigma'' \; a \in M$, and σ instantiates $\sigma'\}$.
A set M of assertions is a model of a formula F, if:

- M is clash-free.
- M satisfies F ($M \models F$), defined recursively as:
  $M \models T$, $M \models \bot$.
  $M \models p$ $(M \models \neg p)$, if $\varepsilon\, p \in M$ $(\varepsilon\, \neg p \in M)$.
  $M \models F \wedge G$, if $M \models F$ and $M \models G$.
  $M \models F \vee G$, if $M \models F$ or $M \models G$.
  $M \models \sigma\, F$, if σ exists in M (as a prefix of some label in M), and for a subset M' of M, $M'|_\sigma \models F$, and for all exceptions σ' in any of the labels in M', (σ,σ') does not exist, or $M \models (\sigma,\sigma')\, F$.

How to convert a consistent set M of assertions into a Kripke model $K = (W,R,V)$:

- Define W as the set of all ground instances within the context of M.
- Define $\sigma\, R_i\, \sigma c_i$, whenever σ and σc_i are in M, and $c_i \in D_i$
- Define $\sigma \in V(p)$, iff σ instantiates a label σ_p so that $\sigma_p p \in M$.

Status of Work

Theory

- Theoretical foundations are completely specified.
- The algorithm is proved sound and complete.
- Theoretical reasoning complexity specified. (See annotations on the example.)
- Remaining open question: Can resolution p-simulate reasoning with labelled formulas?

Implementation and Experimentation

An implementation of a model finder for labelled formulas is in progress.
☛ Goal: Provide experimental evidence for the practical usefulness/superiority of our approach.

Two classes of benchmark problems available:

- randomly-generated formulas (translations of QBF formulas of varying hardness)
  are challenging, but often "artificially" so.
  Some families of hard random formulas may become trivial if certain optimization techniques are enabled.
  Optimizations can be tailoured (to some extent) to the problem class.
- problems arising from real-world applications
  much more interesting.
  Nowadays, problem size is no longer an issue. (Large problems available.)
  However, they are few in number and often contain extra constructs used in DL.
  "CPU-time performance" often determined by pragmatic choices (compiler and hardware optimization, memory caching), as only very large problems are challenging.

Future Work

Global axioms

- Global axioms are standard in DL knowledge bases.
  A Kripke model satisfies a global axiom φ, if $K,w \models \varphi$ for all worlds w.
- Expressible by a slight extension of our formalism, as $*^* \varphi$.
- Problem: Need to use loops to ensure finiteness.
  What is the equivalent of loops in models of labelled formulas?

☛ equivalence classes of labels.

Absorption [Horrocks & Tobies 2000] is a technique for Description Logic with axioms.

- Our formalism is an alternative to absorption.
- But, can absorption be adapted to to formulas without axioms?

Other optimizations

- Dependency-directed Backtracking (Backjumping) is already built into the algorithm.
- We can still do model caching, but is it useful?
- Exploiting variable and role invariances?

References

[Baumgartner & Tinelli 2003] Baumgartner, P., and Tinelli, C. *The model evolution calculus.* In Baader, F., ed., Proceedings of the 19th International Conference on Automated Deduction, CADE-19 (Miami, Florida, USA), number 2741 in Lecture Notes in Artificial Intelligence, 350–364. Springer. 2003.

[Beckert & Goré 2001] Beckert, B., and Goré, R. *Free variable tableaux for propositional modal logics.* Studia Logica 69:59–96. 2001.

[Giunchiglia & Tacchella, 2000] E. Giunchiglia and A. Tacchella. *A subset-matching size-bounded cache for satisfiability of modal logics.* In Proceedings International Conference Tableaux 2000, pages 237–251. Springer. 2000.

[Haarslev & Möller 2003] Haarslev, V., and Möller, R. *Description logic systems.* In Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds., *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press. 427–449. 2003.

[Happe 2001a] Happe, J. *The ModProf theorem prover.* In Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2001), number 2083 in Lecture Notes in Artificial Intelligence, 459–463. Springer. 2001a.

[Happe 2001b] Happe, J. *A subsumption-aided caching technique.* In *Issues in the Design and Experimental Evaluation of Systems for Modal and Temporal Logics* (IJCAR 2001 Workshop), Technical Report DII 14/01, 49–57. Dipartimento di Ingegneria dell'Informazione, Unversit'a degli Studi di Siena, Siena, Italy. 2001b.

[Horrocks & Tobies 2000] Horrocks, I., and Tobies, S. *Reasoning with axioms: Theory and practice.* In Proc. of the 7th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2000), 285–296. 2000.

[Horrocks 2003] Horrocks, I. *Implementation and Optimization Techniques.* In Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; Patel-Schneider, P. F., eds., *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press. 306–346. 2003.

[Klein et al. 2003] Klein, M.; Broekstra, J.; Fensel, D.; van Harmelen, F.; and Horrocks, I. *Ontologies and schema languages on the web.* In Fensel, D.; Hendler, J.; Lieberman, H.; and Wahlster, W., eds., *Spinning the Semantic Web: Bringing the World Wide Web to its full potential.* MIT Press. 2003.

[Ohlbach et al. 2001] Ohlbach, H.; Nonnengart, A.; de Rijke, M.; and Gabbay, D. *Encoding two-valued nonclassical logics into classical logic.* In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning,* volume II. Elsevier Science. chapter 21, 1403–1486. 2001.

[Rector, Nowlan, & Glowinski 1993] Rector, A.; Nowlan, W.; and Glowinski, A. *Goals for concept representation in the Galen project.* In 17th annual Symposium on Computer Applications in Medical Care, Washington, USA, SCAMC 93, 414–418. 1993.