Optimised functional translation and resolution

Ullrich Hustadt[1], Renate A. Schmidt[1], and Christoph Weidenbach[2]

[1] Department of Computing, Manchester Metropolitan University,
Chester Street, Manchester M1 5GD, United Kingdom
{U.Hustadt, R.A.Schmidt}@doc.mmu.ac.uk
[2] Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany
weidenb@mpi-sb.mpg.de

Prover: We facilitate modal theorem proving in a first-order resolution calculus implemented in SPASS Version 0.77 [4]. SPASS uses ordered resolution and ordered factoring, it supports splitting and branch condensing (splitting amounts to case analysis while branch condensing resembles branch pruning in the Logics Workbench), it has an extensive set of reduction rules including tautology deletion, subsumption and condensing, and it supports dynamic sort theories by additional inference and reduction rules.

The translation we use is the *optimised functional translation* [2]. It maps normal propositional modal logics into a class of *path logics*. Path logics are clausal logics over the language of the monadic fragment of sorted first-order logic with a special binary function symbol for defining accessibility. Clauses of path logics are restricted in that only Skolem terms which are constants may occur and the prefix stability property holds. Ordinary resolution without any refinement strategies is a decision procedure for the path logics associated with $K(m)$ and $KT(m)$ [3]. Our decision procedure for $S4$ uses an a priori term depth bound.

Availability: SPASS and a routine for the translation of modal formulae are available from `http://www.mpi-sb.mpg.de/~hustadt/mdp`

Advantages of the prover: SPASS is a fast and sophisticated state-of-the-art first-order theorem prover. Ordered inference rules and splitting are of particular importance when treating satisfiable formulae, while unit propagation and branch condensing are important for benchmarks based on randomly generated modal formulae [1].

Advantages of translation approaches: In its most general form the translation approach can deal with any complete, finitely axiomatizable, normal modal logic. Moreover, any first-order theorem prover can be used, that is, we may substitute SPASS with another theorem prover (not necessarily a resolution theorem prover). The relational and optimised functional translation approach are refinements of the general translation approach towards efficient modal theorem proving. The optimised functional translation is applicable to many propositional modal logics, including K, KT, and S4 and their multi-modal versions, but notably also to some second-order modal logics, like KM [2]. A general result shows that any first-order resolution theorem prover (with condensing) provides a decision procedure for a variety of modal logics [3].

Hardware: Sun Ultra 1 Model 170E (167 MHz UltraSPARC processor, 512 KB second level cache), 192 MB main memory.

Results: On classes of provable formulae (in the first column), the combination of the optimised functional translation approach and SPASS has little difficulty. Notable exceptions are the classes *k_ph_p*, *kt_ph_p*, *s4_ph_p*, *k_branch_p* and *s4_ipc_p*. Observe that SPASS solves more formulae in *s4_branch_p* than in either *kt_branch_p* or *k_branch_p*. While for the basic modal logic, the classes of non-provable formulae (in the second column) are not harder than the classes of provable formulae, we see a noticeable difference between *kt_dum_n*, *kt_poly_n*, and *kt_t4p_n* and the corresponding classes of provable KT-formulae. However, the results are still acceptable. In contrast, for the classes *s4_45_n*, *s4_grz_n*, *s4_s5_n*, and *s4_t4p_n* of non-provable S4-formulae the performance is unsatisfactory. We attribute this to using superposition and not *E*-unification, and to enforcing termination by an explicit term depth bound instead of a loop check. For the classes *s4_45_n* and *s4_s5_n* there are trivial satisfiability checks on the clause level which are not implemented in SPASS.

k_branch_p	9	*k_branch_n*	9
k_d4_p	> 20	*k_d4_n*	18
k_dum_p	> 20	*k_dum_n*	> 20
k_grz_p	> 20	*k_grz_n*	> 20
k_lin_p	> 20	*k_lin_n*	> 20
k_path_p	20	*k_path_n*	20
k_ph_p	6	*k_ph_n*	9
k_poly_p	16	*k_poly_n*	17
k_t4p_p	> 20	*k_t4p_n*	19
kt_45_p	17	*kt_45_n*	6
kt_branch_p	13	*kt_branch_n*	9
kt_dum_p	17	*kt_dum_n*	9
kt_grz_p	> 20	*kt_grz_n*	> 20
kt_md_p	16	*kt_md_n*	20
kt_path_p	> 20	*kt_path_n*	16
kt_ph_p	5	*kt_ph_n*	12
kt_poly_p	16	*kt_poly_n*	3
kt_t4p_p	> 20	*kt_t4p_n*	7
s4_45_p	9	*s4_45_n*	0
s4_branch_p	> 20	*s4_branch_n*	4
s4_grz_p	14	*s4_grz_n*	0
s4_ipc_p	6	*s4_ipc_n*	> 20
s4_md_p	9	*s4_md_n*	10
s4_path_p	15	*s4_path_n*	> 20
s4_ph_p	5	*s4_ph_n*	5
s4_s5_p	> 20	*s4_s5_n*	1
s4_t4p_p	11	*s4_t4p_n*	0

Table 1: Performance indices

References

1. U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logics. In *Proc. IJCAI'97*, pages 202–207, 1997.
2. H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. Res. Report MPI-I-95-2-002, MPI f. Informatik, Saarbrücken, 1995.
3. R. A. Schmidt. Resolution is a decision procedure for many propositional modal logics: Extended abstract. To appear in *Proc. AiML'96*, 1997.
4. C. Weidenbach, B. Gaede, and G. Rock. SPASS & FLOTTER version 0.42. In *Proc. CADE-13*, LNAI 1104, pages 141–145, 1996.