Logical Engineering with Instance-Based Methods

Peter Baumgartner

NICTA and
Australian National University
Canberra, Australia
Peter.Baumgartner@nicta.com.au

1 Instance Based Methods

The term "instance based methods" (IMs) refers to a certain family of methods for first-order logic theorem proving. IMs share the principle of carrying out proof search by maintaining a set of instances of input clauses and analyzing it for satisfiability until completion. IMs are conceptually essentially different to well established methods like resolution or free-variable analytic tableaux. (See [Pla94] for a comparison of various calculi and strategies, including an instance based method.) Also, IMs exhibit a search space and termination behaviour (in the satisfiable case) different from those methods, which makes them attractive from a practical point of view as a complementary method.

The idea behind IMs is already present in a rudimentary way in the work by Davis, Putnam, Logemann and Loveland, and others, in the early sixties of the last century [DP60, DLL62b, DLL62a, Dav63, CDHM64]. The contemporary stream of research on IMs was initiated with the Lee and Plaisted's Hyperlinking calculus [PL90, LP92] Since then, other methods have been developed by Plaisted and his coworkers [CP94, PZ97, PZ00]; Billon's disconnection calculus [Bil96] was picked up by Letz and Stenz and has been significantly developed further since then [LS01, LS02, SL04, LS07]. New methods have been described in [Bau98, BEF99], by Hooker [HRCS02], and by Ganzinger and Korovin [GK03, GK04, GK06]. See [JW07] for a thorough comparison of some of these IMs.

The author introduced a first-order version of the propositional DPLL procedure, FDPLL [Bau00, Bau02], which is now subsumed by the Model Evolution (ME) calculus [BT03, BT05, BFT06b]. The model representation formalism in the ME calculus has been studied in [FP05].

Some quite sophisticated implementations are available, two of which, Darwin [BFT06a] and DCTP [Ste02], regularly participate in the CASC competition.

Open research questions concern, for example, better understanding of theoretical properties and comparison with other methods, implementation techniques, extensions for reasoning modulo fixed background theories, variants for deciding more fragments of first-order logic than currently known, and adaption for specific applications, as outlined in the following section.

F. Pfenning (Ed.): CADE 2007, LNAI 4603, pp. 404–409, 2007.

2 Logical Engineering

The term *logical engineering* is meant to refer to the strategy to exploit the properties of the logical calculus or proof procedure at hand to solve a given problem or class of problems. This may involve translating the problem into a suitable logical language and also specific tailoring the calculus or tuning the proof procedure.

For example, in the context of IMs, the ME calculus when applied to *range-restricted* [MB88] clause sets can be used as a bottom-up model-generation method similar to hyper tableau [BFN96] or hyper resolution (with splitting). This setup has applications, e.g., for reasoning in modal logics (in conjunction with certain translations into first-order clause logic) [GHS02, SH05, e.g.]. Any such application is thus amenable to Model Evolution, too.

An interesting property of *all* (known) instance based methods is that they provide natural decision procedures for the Bernays-Schönfinkel fragment of first-order logic, or, more precisely, function-free clause logic (FFCL).[1] In contrast, most other first-order methods, including free-variable tableau and resolution methods cannot be used as decision procedures for FFCL.[2] This suggests to capitalize on this distinguishing property of IMs and to investigate reduction of application problems into FFCL.

For instance, the optimized functional translation of modal logics [OS97] leads directly to FFCL [Sch99]. Many benchmark problems obtained this way are contained in the TPTP problem library [SS98], and implementations of instance based methods consistently score very well on them. In the description logic context, [MSS04] show how to translate the expressive description logic $\mathcal{SHIQ}$(D) to FFCL (with a different motivation, though).

Another "generic" application area is *finite model computation*. Most approaches for finite model computation essentially work by stepwise reduction to formulas in propositional logic [Sla92, McC94, ZZ95, CS03, e.g.].[3] In [BF$^+$07] we have shown how the MACE-style model computation paradigm can be rooted in the Model Evolution calculus instead of a SAT solver, which can lead to space advantages.

In a similar spirit, IMs might be usable within a SMT architecture (satisfiability modulo theories, see [RT06] for a recent overview). For instance, one could design a DPLL(T) solver [Tin02,NOT06] based on the ME calculus, which would thus be equipped with a first-order variant of a DPLL solver instead of a propositional one. The then available native first-order reasoning capabilities could turn out to be useful for various purposes, such as guiding the search for a candidate model by adding (redundant) theory axioms to the clause set, solving problems over an extended background theory, or going beyond quantifier-free problems. The latter will typically require an heuristic approach, though, as no complete

[1] i.e., first-order clause logic without function symbols of arity > 0.

[2] Not counting approaches that are based on reduction to ground clause logic. See [BS06] for recent improvements of that approach.

[3] But see [FL96,BT98,Pel03b,Pel03a,BS06,dNM06, e.g.] for direct first-order methods; [FLHT01] is a more general overview.

calculus can exist already for rather simple decidable arithmetic theories when extended with free predicate symbols [MHM98].

Other (potential) applications of the FFCL fragment and IMs lie within the *constraint programming* area. Perhaps IMs are not the preferred choice as solvers for search problems (in NP), as this is the domain of the traditional constraint programming paradigm. More appropriate seems the application to, e.g., model expansion problems [MTHM06] (with NEXPTIME combined query/data complexity), which can be reduced to FFCL in a way similar to finite model computation mentioned above. Another application is to reason *about* constraint specifications for the purpose to prove “interesting” properties, like functional dependencies and symmetries [CM05, CM04]. Quite often, the resulting proof obligations lie within FFCL.

References

[Bau98] Baumgartner, P.: Hyper Tableaux — The Next Generation. In: de Swart, H. (ed.) TABLEAUX 1998. LNCS (LNAI), vol. 1397, pp. 60–76. Springer, Heidelberg (1998)

[Bau00] Baumgartner, P.: FDPLL – A First-Order Davis-Putnam-Logeman-Loveland Procedure. In: McAllester, D. (ed.) Automated Deduction - CADE-17. LNCS, vol. 1831, pp. 200–219. Springer, Heidelberg (2000)

[Bau02] Baumgartner, P.: A First-Order Logic Davis-Putnam-Logemann-Loveland Procedure. In: Lakemeyer, G., Nebel, B. (eds.) AI in the new Millenium, Morgan Kaufmann, Seattle (2002)

[BEF99] Baumgartner, P., Eisinger, N., Furbach, U.: A confluent connection calculus. In: Ganzinger, H. (ed.) Automated Deduction - CADE-16. LNCS (LNAI), vol. 1632, pp. 329–343. Springer, Heidelberg (1999)

[BF+07] Baumgartner, P., Fuchs, A., de Nivelle, H., Tinelli, C.: Computing finite models by reduction to function-free clause logic. Journal of Applied Logic (to appear)

[BFN96] Baumgartner, P., Furbach, U., Niemelä, I.: Hyper Tableaux. In: Orłowska, E., Alferes, J.J., Moniz Pereira, L. (eds.) JELIA 1996. LNCS, vol. 1126, Springer, Heidelberg (1996)

[BFT06a] Baumgartner, P., Fuchs, A., Tinelli, C.: Implementing the model evolution calculus. International Journal of Artificial Intelligence Tools 15(1), 21–52 (2006)

[BFT06b] Baumgartner, P., Fuchs, A., Tinelli, C.: Lemma learning in the model evolution calculus. In: Hermann, M., Voronkov, A. (eds.) LPAR 2006. LNCS (LNAI), vol. 4246, pp. 572–586. Springer, Heidelberg (2006)

[Bil96] Billon, J.-P.: The Disconnection Method. In: Miglioli, P., Moscato, U., Ornaghi, M., Mundici, D. (eds.) TABLEAUX 1996. LNCS, vol. 1071, pp. 110–126. Springer, Heidelberg (1996)

[BS06] Baumgartner, P., Schmidt, R.: Blocking and other enhancements for bottom-up model generation methods. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, Springer, Heidelberg (2006)

[BT98] Bry, F., Torge, S.: A Deduction Method Complete for Refutation and Finite Satisfiability. In: Proc. 6th European Workshop on Logics in AI (JELIA). LNCS (LNAI), Springer, Heidelberg (1998)

[BT03] Baumgartner, P., Tinelli, C.: The Model Evolution Calculus. In: Baader, F. (ed.) Automated Deduction – CADE-19. LNCS (LNAI), vol. 2741, pp. 350–364. Springer, Heidelberg (2003)

[BT05] Baumgartner, P., Tinelli, C.: The model evolution calculus with equality. In: Nieuwenhuis [Nie05], pp. 392–408

[Bun94] Bundy, A. (ed.): Automated Deduction - CADE-12. LNCS, vol. 814. Springer, Heidelberg (1994)

[CDHM64] Chinlund, T.J., Davis, M., Hinman, P.G., McIlroy, M.D.: Theorem-Proving by Matching. Technical report, Bell Laboratories (1964)

[CM04] Cadoli, M., Mancini, T.: Exploiting functional dependencies in declarative problem specifications. In: Alferes, J.J., Leite, J.A. (eds.) JELIA 2004. LNCS (LNAI), vol. 3229, pp. 628–640. Springer, Heidelberg (2004)

[CM05] Cadoli, M., Mancini, T.: Using a theorem prover for reasoning on constraint problems. In: AI*IA, pp. 38–49 (2005)

[CP94] Chu, H., Plaisted, D.A.: Semantically Guided First-Order Theorem Proving using Hyper-Linking. In: Bundy [Bun94], pp. 192–206

[CS03] Claessen, K., Sörensson, N.: New techniques that improve mace-style finite model building. In: Baader, F. (ed.) Automated Deduction – CADE-19. LNCS (LNAI), vol. 2741, Springer, Heidelberg (2003)

[Dav63] Davis, M.: Eliminating the irrelevant from mechanical proofs. In: Proceedings of Symposia in Applied Amthematics – Experimental Arithmetic, High Speed Computing and Mathematics, vol. XV, pp. 15–30. American Mathematical Society (1963)

[DLL62a] Davis, M., Logemann, G., Loveland, D.: A machine program for theorem proving. Communications of the ACM 5(7) (1962)

[DLL62b] Davis, M., Logemann, G., Loveland, D.: A machine program for theorem proving. Communications of the ACM 5(7), 394–397 (1962)

[dNM06] de Nivelle, H., Meng, J.: Geometric resolution: A proof procedure based on finite model search. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, Springer, Heidelberg (2006)

[DP60] Davis, M., Putnam, H.: A computing procedure for quantification theory. Journal of the ACM 7(3), 201–215 (1960)

[EF02] Egly, U., Fermüller, C.G. (eds.): TABLEAUX 2002. LNCS (LNAI), vol. 2381. Springer, Heidelberg (2002)

[FL96] Fermüller, C., Leitsch, A.: Hyperresolution and automated model building. J. Log. Comput. 6(2), 173–203 (1996)

[FLHT01] Fermüller, C.G., Leitsch, A., Hustadt, U., Tammet, T.: Resolution decision procedures. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning (ch. 25), 25th edn., vol. II, pp. 1791–1849. Elsevier, North-Holland (2001)

[FP05] Fermüller, C.G., Pichler, R.: Model representation via contexts and implicit generalizations. In: Nieuwenhuis [Nie05], pp. 409–423

[GHS02] Georgieva, L., Hustadt, U., Schmidt, R.A.: Hyperresolution for guarded formulae. Journal of Symbolic Computation (2002)

[GK03] Ganzinger, H., Korovin, K.: New directions in instance-based theorem proving. In: LICS - Logics in Computer Science (2003)

[GK04] Ganzinger, H., Korovin, K.: Integrating equational reasoning into instantiation-based theorem proving. In: Marcinkowski, J., Tarlecki, A. (eds.) CSL 2004. LNCS, vol. 3210, pp. 71–84. Springer, Heidelberg (2004)

[GK06] Ganzinger, H., Korovin, K.: Theory Instantiation. In: Hermann, M., Voronkov, A. (eds.) LPAR 2006. LNCS (LNAI), vol. 4246, Springer, Heidelberg (2006)

[HRCS02] Hooker, J.N., Rago, G., Chandru, V., Shrivastava, A.: Partial Instantiation Methods for Inference in First Order Logic. Journal of Automated Reasoning 28(4), 371–396 (2002)

[JW07] Jacobs, S., Waldmann, U.: Comparing instance generation methods for automated reasoning. J. Autom. Reason. 38(1-3), 57–78 (2007)

[LP92] Lee, S.-J., Plaisted, D.: Eliminating Duplicates with the Hyper-Linking Strategy. Journal of Automated Reasoning 9, 25–42 (1992)

[LS01] Letz, R., Stenz, G.: Proof and Model Generation with Disconnection Tableaux. In: Nieuwenhuis, R., Voronkov, A. (eds.) LPAR 2001. LNCS (LNAI), vol. 2250, Springer, Heidelberg (2001)

[LS02] Letz, R., Stenz, G.: Integration of Equality Reasoning into the Disconnection Calculus. In: Egly and Fermüller [EF02], pp. 176–190

[LS07] Letz, R., Stenz, G.: The disconnection tableau calculus. J. Autom. Reason. 38(1-3), 79–126 (2007)

[MB88] Manthey, R., Bry, F.: SATCHMO: a theorem prover implemented in Prolog. In: Lusk, E., Overbeek, R. (eds.) CADE-9. LNCS, vol. 310, pp. 415–434. Springer, Heidelberg (1988)

[McC94] McCune, W.: A davis-putnam program and its application to finite first-order model search: Qusigroup existence problems. Technical report, Argonne National Laboratory (1994)

[MHM98] Mesnard, F., Hoarau, S., Maillard, A.: CLP(χ) for automatically proving program properties. J. Log. Program 37(1-3), 77–93 (1998)

[MSS04] Motik, B., Sattler, U., Studer, R.: Query answering for owl-dl with rules. In: Dubois, D., Welty, C.A., Williams, M.-A. (eds.) International Semantic Web Conference, pp. 549–563. AAAI Press, Stanford (2004)

[MTHM06] Mitchell, D., Ternovska, E., Hach, F., Mohebali, R.: Model expansion as a framework for modelling and solving search problems. Technical Report TR 2006-24 (December 2006)

[Nie05] Nieuwenhuis, R. (ed.): Automated Deduction – CADE-20. LNCS (LNAI), vol. 3632. Springer, Heidelberg (2005)

[NOT06] Nieuwenhuis, R., Oliveras, A., Tinelli, C.: Solving SAT and SAT Modulo Theories: from an Abstract Davis-Putnam-Logemann-Loveland Procedure to DPLL(T). Journal of the ACM 53(6), 937–977 (2006)

[OS97] Ohlbach, H.J., Schmidt, R.A.: Functional translation and second-order frame properties of modal logics. Journal of Logic and Computation 7(5), 581–603 (1997)

[Pel03a] Peltier, N.: A calculus combining resolution and enumeration for building finite models. Journal of Symbolic Computation 36(1-2), 49–77 (2003)

[Pel03b] Peltier, N.: A more efficient tableaux procedure for simultaneous search for refutations and finite models. In: Mayer, M.C., Pirri, F. (eds.) TABLEAUX 2003. LNCS, vol. 2796, pp. 181–195. Springer, Heidelberg (2003)

[PL90] Plaisted, D.A., Lee, S.-J.: Inference by clause matching. In: Ras, Z.W., Zemankova, M. (eds.) Intelligent Systems: State of the Art and Future Directions, pp. 200–235. Ellis Horwood, New York (1990)

[Pla94] Plaisted, D.: The Search Efficiency of Theorem Proving Strategies. In: Bundy [Bun94]

[PZ97] Plaisted, D.A., Zhu, Y.: Ordered Semantic Hyper Linking. In: Proceedings of Fourteenth National Conference on Artificial Intelligence (AAAI-97) (1997)

[PZ00] Plaisted, D.A., Zhu, Y.: Ordered Semantic Hyper Linking. Journal of Automated Reasoning 25(3), 167–217 (2000)

[RT06] Ranise, S., Tinelli, C.: Satisfiability modulo theories. Trends and Controversies - IEEE Intelligent Systems Magazine 21(6), 71–81 (2006)

[Sch99] Schmidt, R.A.: Decidability by resolution for propositional modal logics. J. Autom. Reason. 22(4), 379–396 (1999)

[SH05] Schmidt, R.A., Hustadt, U.: The axiomatic translation principle for modal logic. ACM Transactions on Computational Logic (to appear)

[SL04] Stenz, G., Letz, R.: Generlized handling of variables in disconnection tableaux. In: Basin, D., Rusinowitch, M. (eds.) IJCAR 2004. LNCS (LNAI), vol. 3097, pp. 289–306. Springer, Heidelberg (2004)

[Sla92] Slaney, J.: Finder (finite domain enumerator): Notes and guide. Technical Report TR-ARP-1/92, Australian National University, Automated Reasoning Project, Canberra (1992)

[SS98] Sutcliffe, G., Suttner, C.B.: The TPTP Problem Library: CNF Release v1.2.1. Journal of Automated Reasoning 21(2), 177–203 (1998)

[Ste02] Stenz, G.: DCTP 1.2 - System Abstract. In: Egly and Fermüller [BF02], pp. 335–340

[Tin02] Tinelli, C.: A DPLL-based calculus for ground satisfiability modulo theories. In: Flesca, S., Greco, S., Leone, N., Ianni, G. (eds.) JELIA 2002. LNCS (LNAI), vol. 2424, Springer, Heidelberg (2002)

[ZZ95] Zhang, J., Zhang, H.: Sem: a system for enumerating models. In: IJCAI-95 — Proceedings of the 14^{th} International Joint Conference on Artificial Intelligence, Montreal, pp. 298–303 (1995)