Machine-independent evaluation of theorem-proving strategies

(Position paper)

Maria Paola Bonacina *

Department of Computer Science
The University of Iowa
Iowa City, IA 52242-1419, USA
bonacina@cs.uiowa.edu

The evaluation of theorem-proving strategies has been done traditionally in an empirical manner: a strategy is implemented in a theorem prover, the prover is applied to a number of theorems, and the running times are reported and compared with those of other systems. In recent years, a growing effort has been devoted to make the evaluation of theorem provers more systematic. The need for a standard collection of theorem-proving problems (e.g., the TPTP library [9]) and a standard set of empirical measures has been recognized (e.g., [8]).

While benchmarking of theorem provers is necessary, and the progress in the methodology of empirical evaluation is important for the field, the problem of *strategy evaluation* remains open. A theorem prover is made of many components in addition to the strategy, including data structures, indexing techniques and service algorithms such as those for unification or term replacement. The performance of a theorem prover depends on all these components and the overall engineering of the system. It is very difficult to establish quantitatively how different features contribute to the observed performance. Therefore, empirical evaluation is evaluation of theorem-proving systems, not theorem-proving strategies. The goal of evaluating strategies independent of implementation requires the development of a theory of *"strategy analysis,"* comparable to algorithm analysis, and with potentially similar beneficial consequences, not only for theorem proving, but also for logic programming and all applications of deduction.

The idea of "strategy analysis" is new. Most of the work on search in artificial intelligence concentrates on the design of heuristics (e.g., [5]). Most of the research in complexity related to theorem proving studies the complexity of propositional proofs as part of the quest for $NP \neq co\text{–}NP$ (e.g., see [10] for a survey), or works with complexity measures based on the Herbrand theorem to determine lower bounds for sets of clauses, not upper bounds for strategies (e.g., [2, 4, 7]). In resolution theorem proving, the classical source for the modelling of search is [3], which was not concerned with evaluating the complexity of the strategies.

The primary objective of strategy analysis is to study the complexity of *searching for a proof.* An approach to this problem was proposed in [6]. It applies classical techniques from algorithm analysis to derive worst-case upper bounds

* Supported in part by the National Science Foundation with grant CCR-94-08667.

on the total size of the search space of various theorem-proving strategies in propositional Horn logic. This approach assumes that the search space is *finite*, so that presently it is not known how it could be extended to first-order logic.

An approach that aims at *infinite search spaces* was presented in [1]. The first component of a methodology for strategy analysis is a *model of the search space* and *search process* (i.e., the application of the search plan to the search space). The second step is to define *measures of complexity* in such a model, so that strategies can be evaluated and compared in terms of such measures. The third step is to carry out the analysis. In [1], the *model of the search space* is a *marked search graph*. The search graph represents all possible inferences. Unlike expansion inferences, however, contraction inferences, such as simplification or subsumption, cannot be represented by a static graph. Expansion inferences *visit* the graph. Contraction inferences *visit and modify* the graph by deletions, which cannot be represented prior to the search. Therefore, the search graph is enriched with a *marking*. The marking is used to represent dynamically the *search process*, including the selections by the search plan, the expansion steps and the contraction steps.

This model of the search space supports measures of complexity of search for infinite search spaces. The classical notion of complexity is complexity of a computation which is guaranteed to halt, and therefore deals with finite objects. For derivations that may not halt, the approach of [1] is to observe that while a strategy operates with a finite amount of data, this data represents a portion of the infinite space that the strategy is searching. In order to capture the complexity of the search, the analysis needs to involve both the *present* and the *future* of the derivation. In terms of the search graph, the present is the part explored so far, and the future is the unexplored part. The analysis needs to study how the inferences selected by the strategy affect both.

Complexity measures for this purpose are obtained by defining notions of *ancestor-graph* and *dynamic distance*, which replace the conventional notions of path and path-length. Unlike the latter, the distance is dynamic, because it depends on the actions of the strategy. For instance, if present clauses are deleted by contraction, clauses in the future may become *unreachable*. Given this notion of distance, it is possible to define the *bounded search space* with bound j as the space of clauses reachable within distance j. The infinite search space is viewed as an infinite succession (for all j) of bounded search spaces. Since the bounded search spaces are finite, they can be compared. Furthermore, they are characterized as *multisets of clauses*, so that it is possible to compare them with the multiset extension of a *well-founded ordering on clauses*. Thus, the notion of bounded search space allows us to handle infinite search spaces finitely.

In [1] this framework is applied to compare two generic strategies with the same expansion rules, the same search plan, but different contraction rules. It is shown that contraction reduces the bounded search spaces, and therefore the search complexity. Then it is proved that the strategy with more contraction induces a higher reduction of search complexity. This is the first analysis that translates the empirical observations of the efficacy of contraction in theorem proving into a formal result of reduction of complexity.

At present, this type of analysis is not asymptotic. Possible directions for future research include exploring approaches to make the analysis asymptotic, comparing strategies with different search plans, extending the framework from sequential to parallel search, and from forward-reasoning to backward-reasoning.

References

1. M. P. Bonacina and J. Hsiang. On the representation of dynamic search spaces in theorem proving. In C.-S. Yang, editor, *Proc. of the Int. Computer Symposium*, pages 85–94, 1996. Abstract presented at the *Logic Colloquium 1996*, and full version available as Tech. Rep. 95-05, Dept. of Comp. Sci., Univ. of Iowa.
2. J. Goubault. The complexity of resource-bounded first-order classical logic. In *Proc. of the 11th STACS*, volume 775 of *LNCS*, pages 59–70. Springer Verlag, 1994.
3. R. Kowalski. Search strategies for theorem proving. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 5, pages 181–201. Edinburgh University Press, 1969.
4. V.-P. Orevkov. Lower bounds for increasing complexity of derivations after cut elimination. *J. of Soviet Mathematics*, pages 2337–2350, 1982.
5. J. Pearl. *Heuristics – Intelligent search strategies for computer problem solving.* Addison Wesley, Reading, MA, 1984.
6. D. A. Plaisted. The search efficiency of theorem proving strategies. In A. Bundy, editor, *Proc. of CADE-12*, volume 814 of *LNAI*, pages 57–71. Springer Verlag, 1994. Full version available as Tech. Rep. MPI I-94-233.
7. R. Statman. Lower bounds on Herbrand theorems. In *Proc. AMS 75*, pages 104–107, 1979.
8. G. Sutcliffe and C. Suttner. The design of the CADE-13 ATP system competition. Technical Report 95/15 and AR-95-05, Dept. of Comp. Sci., JCU, Australia and Inst. für Informatik, TU München, Germany, 1996.
9. G. Sutcliffe, C. Suttner, and T. Yemenis. The TPTP problem library. In A. Bundy, editor, *Proc. of CADE-12*, volume 814 of *LNAI*, pages 252–266. Springer Verlag, 1994.
10. A. Urquhart. The complexity of propositional proofs. *Bull. of Symbolic Logic*, 1:425–467, 1995.