Dynamic Backtracking for Modal Logics

Zhen Li
*University of Manchester
liz@cs.man.ac.uk

Renate Schmidt
†University of Manchester
schmidt@cs.man.ac.uk

Abstract

Backtracking algorithms can make a considerable difference on the performance of theorem provers. Dynamic backtracking uses elimination explanations to record information from earlier failures in order to find suitable decision points where to backtrack to. Unlike backjumping and conflict-directed backjumping, dynamic backtracking only undoes the steps relevant to the current contradiction. We have developed a dynamic backtracking algorithm for both modal clauses and general modal formulae and proved the correctness of our algorithms. The algorithms are implemented in the modal logic theorem prover MLTP. Performance analysis shows that dynamic backtracking has a clear advantage over backjumping and MLTP is reasonably efficient compared with other highly optimized provers.

Introduction Previous efforts in the area have shown that intelligent backtracking algorithms play a significant (if not the most important) role among all the optimization techniques for tableau theorem provers. Intelligent backtracking algorithms are improvements over traditional chronological backtracking. Chronological backtracking is not efficient because it does not remember the reason of its earlier failed attempts. As a consequence it may waste too much effort on the same failure. In other words, if the reasons of earlier failure can be used to guide the following proof search, the performance probably can be significantly improved. Better backtracking algorithms have been proposed and studied. Examples include backjumping (Gaschnig (1979)) and conflict-directed backjumping (Prosser (1993)).

Conflict-directed backjumping employs conflict sets to store information about previous failures. With the help of the conflict sets, conflict-directed backjumping can directly backtrack to the right decision point which is the true reason of the current failure without exploring irrelevant decision points. Although conflict-directed backjumping saves time by performing less backtracking, it undoes too much work unnecessarily during backtracking. Therefore Ginsberg proposed a new form of backtracking, called dynamic backtracking (Ginsberg and McAllester (1994)), to solve constraint satisfaction problems. Dynamic backtracking can be viewed as an improvement on conflict-directed backjumping. It has mechanisms not only to reduce the amount of backtracking performed but also to undo as little work as possible during backtracking.

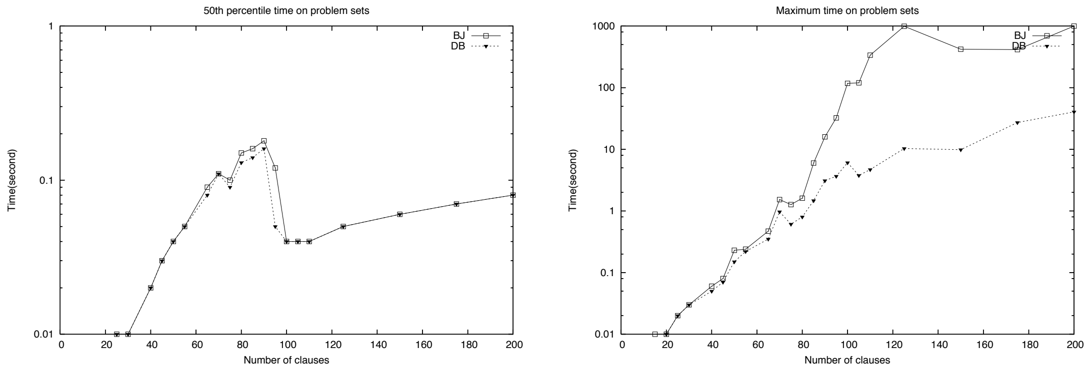
Dynamic backtracking Dynamic backtracking boasts two features that make it worth investigating. Firstly, it uses only polynomial space to record the information which is used to control backtracking. Secondly, it only performs backtrackings on the inference steps that are relevant to the current clash. Dynamic backtracking improves chronological backtracking by keeping more information. When a clash happens, there are two complementary literals and their parent disjunctions are called conflicts. Dynamic backtracking remembers the conflict information for each disjunct of a disjunctive formula and this information is passed to preceding disjunctions when backtracking. In dynamic backtracking this conflict information are called *elimination explanations*. The elimination explanation for a disjunct ϕ is in the form, $E_{\phi} = \{\psi_1, \psi_2, \psi_3\}$. The meaning is that ϕ cannot be true because the current choices of the disjunctions ψ_1, ψ_2 and ψ_3. In other words, to make ϕ true backtracking has to happen on at least one of these disjunctions ψ_1, ψ_2 and ψ_3.

Dynamic backtracking has been developed and studied for propositional clause sets, i.e. SAT problems. We have extended the dynamic backtracking algorithm to modal logic clause sets, but also general propositional formula sets and general modal logic formula sets.

Theorem 1. *Our algorithms with dynamic backtracking are sound, complete decision procedures for modal logic K.*

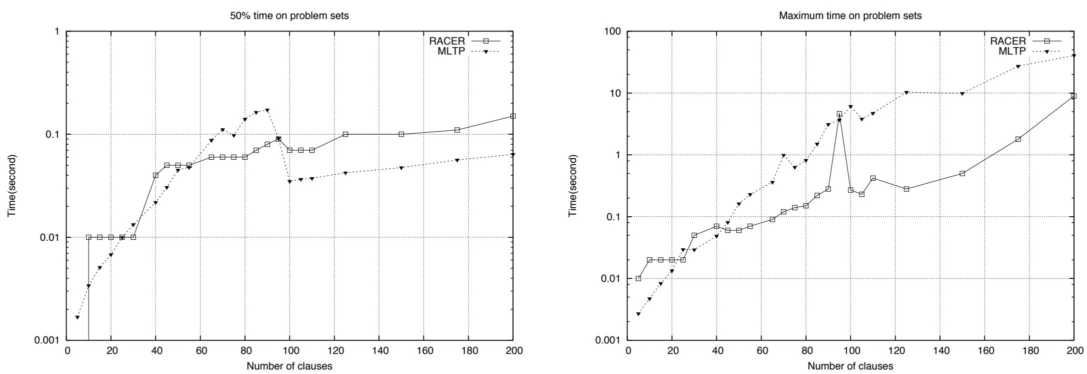
Implementation and evaluation We have implemented the algorithm in an automated theorem prover for modal logics, called MLTP. The system is written in C++. What distinguishes MLTP from the other currently available modal logic and description logic provers is that MLTP features a generic framework. In this framework users can specify or build their own strategies. They can create new optimization methods and make these methods work smoothly within the system. They can even add functions to handle special operators from a certain logic.

Empirical tests have been conducted to evaluate the performance of our different techniques including different backtracking techniques. The tests used randomly generated problems (Hustadt and Schmidt (1997)). Both dynamic backtracking and backjumping are implemented in MLTP. We compared the results of the prover using dynamic backtracking with results of the prover using backjumping. From Figure 1 we can see that in most of the cases dynamic backtracking is the better choice. We also compared the performance of our prover with RACER, a state-of-the-art description logic and modal logic prover (Haarslev and Möller (2001)). Figure 2 shows that the average performance of our system is comparable with RACER. However the right graph shows that for the most difficult problems MLTP is slower than RACER in a scale of 1 order. This is not surprising since MLTP has to sacrifice performance for generality.



a) 50th percentile CPU time   b) 100th percentile CPU time

Figure 1: Dynamic backtracking vs. Backjumping



a) 50th percentile CPU time   b) 100th percentile CPU time

Figure 2: MLTP vs. RACER

References

J. Gaschnig. *Performance Measurement and Analysis of Certain Search Algorithms*. PhD thesis, Carnegie-Mellon University, Pittsburgh, PA, 1979.

M. L. Ginsberg and D. A. McAllester. Gsat and dynamic backtracking. In P. Torasso, J. Doyle, and E. Sandewall, editors, *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, pages 226–237. Morgan Kaufmann, 1994.

V. Haarslev and R. Möller. RACER system description. In *Proc. IJCAR 2001*, volume 2083 of *LNAI*, pages 701–706. Springer, 2001.

U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logics. In M. Pollack, editor, *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, volume 1, pages 202–207. Morgan Kaufmann, August 1997.

Patrick Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9(3):268–299, 1993.