Scheduling Methods for Parallel Automated Theorem Proving*

Gernot Stenz\(^1\) and Andreas Wolf\(^2\)

\(^1\) Institut für Informatik der Technischen Universität München 
80290 München, Germany 
stenzg@in.tum.de

\(^2\) NorCom Information Technology AG 
Stefan-George-Ring 6 
81929 München, Germany 
avo@norcom.de

Abstract. One of the key issues in automated theorem proving is the search for optimal proof strategies. Since there is not one uniform strategy which works optimally on all proof tasks, one is faced with the difficult problem of selecting a good strategy for a given task. Strategy parallelism is a way of circumventing this strategy selection problem. However, the problem of selecting the parallel strategies and distributing the available resources remains. Therefore we have developed a method for strategy evaluation and selection based on training data. We present a theorem prover system which has been automated with respect to the entire process of theorem prover application including automatic data generation, automatic schedule selection and classical automated theorem proving. In the theorem prover e-SETHEO, we present an implementation of such a system that, for the first time, can handle the necessary problem domain adaption fully automatically and which is an improvement of the prover which solved the largest number of problems in the MIX division of the CADE-16 ATP competition. This is followed by some experimental data produced with this system. We address the problem of test set extraction and give an assessment of our work as well as a lookout to future research issues.

1 Introduction

Automated Theorem Proving (ATP) is the subfield of computer science dealing with the automatic verification of the validity of logical formulae. Attempting to prove the validity of such formulae automatically, particularly beyond simple textbook examples, typically results in a tremendously large search space. Such a search problem is usually solved by a uniform search procedure. In automated deduction, different search strategies may behave significantly different on a given problem. Unfortunately, in general, it cannot be decided in advance which

* This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Sonderforschungsbereich 342.
strategy is the best for a given problem. This motivates the competitive use of different strategies, especially when the available resources are restricted. In order to be successful with such an approach, the strategies must satisfy the following two conditions. **Sub-linearity:** Let $sol(s,t)$ denote the set of problems solved with a strategy $s$ in time $t$. Then, for a typical set of problems, the function $\frac{\text{sol}(s,t)}{t}$ must be sub-linear, i.e., with each additional time interval fewer new problems are solved. **Complementarity:** The competing strategies must be complementary w.r.t. a given problem set, i.e., the sets of problems solved in a certain time limit by two different strategies should differ significantly.

If both conditions are satisfied, then a competitive use of different strategies can be more successful than the best single strategy.

This paper is organized as follows. In Section 2 we give a brief overview on automated theorem proving and relate strategy parallelism with other parallelization methods in automated deduction. We briefly address problems like partitioning and strategy allocation. Furthermore, in this Section we give an outline of the design and the implementation of our strategy parallel theorem prover e-SETHEO. Then, in Section 3 we describe our data generation, strategy evaluation and selection principles. Section 4 presents some experimental results obtained with this system. The extraction of test sets and the effects of test set reduction on the prover performance are discussed in Section 5. This is followed by an outlook on future development and an assessment of our current work in Section 6 and, finally, a conclusion in Section 7.

2 Strategies and Strategy Parallelism; a Framework for a Strategy Parallel Prover

Many ways of organizing parallel computing have already been studied. However, many of these methods do not apply to automated theorem proving, since it is generally impossible to predict the size of each of the parallelized subproblems and it is therefore very hard to create an even workload distribution among the different agents. Here, we cite some of the successful examples.

A central concept in our work is the (search) strategy. For us, a strategy is one particular way of traversing the search space. From our practical point of view, a strategy is a calculus for automated theorem proving combined with a particular search method.

We are now looking for a way of efficiently combining and applying different strategies in parallel. An example can be found in the **nagging** concept [SS94a]. Dependent subtasks are sent by a master process to the naggers, which try to solve them and report on their success. The results are integrated into the main proof attempt. A combination of different strategies is used within the team-work concept of DISCOUNT [DKS97] for unit equality problems. These strategies periodically exchange intermediate results and work together evaluating these intermediate results and determining the further search strategies. Strategy selection techniques are applied even in systems with finite search spaces like