ParaGlueminisat

Seongsoo Moon
Graduate School of Information Science and Technology,
The University of Tokyo, Japan

Inaba Mary
Graduate School of Information Science and Technology,
The University of Tokyo, Japan

***Abstract*—We briefly introduce our solver, ParaGlueminisat, submitted to SAT-Race 2015. This solver is a parallel version of Glueminisat with several deterministic policies.**

I. Introduction

Diversification of search space has contributed to the rapid progress in SAT solving, and appears to be one of the most important keys in modern SAT solvers. It also plays an important role in portfolio-based parallel SAT solving. However, in portfolio solvers, maintenance of diversification among solvers is not that simple, especially for massively parallel machines. In this description we implements parallel version of glueminisat with several policies to diversify or intensify search space.

II. *Shuffling* of variables in order

In portfolio approaches, the maintenance of the diversification and intensification tradeoff is very important[2]. Diversification is easily acquired with random noise, but this will lead to a different search space, and intensification will deteriorate. Our goal is to obtain extra diversity while maintaining the intensity of search.

It is well known factor that small changes in CDCL solvers can affect their running time substantially[1]. To add small changes, we shuffle variables in order. *Shuffling* method have some merits. Firstly, this method does not destroy the structure of the problem, but this still changes the order of the varibles which carries the possibility of changing the running times. Therefore, *shuffling* method might produce extra diversity. Secondly, this is a simple pre-processing method, and can be easily applied to any solver especially in massively parallel environments.

III. *recursive reshuffling*

We designed *recursive reshuffling* for more active use of *shuffling* method. Several threads have a time limit, and reshuffle recursively when they reach that time limit. In our assumption, certain index orders would cause search to fall into desert search spaces and never find the solution. If we use *recursive reshuffling*, we can escape desert search spaces and might obtain extra diversity of search spaces. Details are as follows.

- *p*: Process number
- *MTL*: Maximum time limit for recursive search
- *MN*: Maximum number of recursive searches in time *MTL*
- *ET*: Extra time to trigger reshuffling process
- *tr*[*i*]: Time limit for recursive reshuffling (*i* + 1) times

$$tr[i] = \frac{MTL-(i+2)\times ET}{i+2}$$

- t_i: Time limit of *i*th thread for reshuffling

$$t_i = \begin{cases} tr[\frac{(MN-1)\times(i+1)}{i\%MN}], & \text{if } i \leq (p-4)\%MN \\ tr[i\%MN], & \text{otherwise} \end{cases}$$

IV. ParaGluminisat

We implemented the parallel solver ParaGlueminisat. Glueminisat is used as the base solver. In our program, we set policies based on LBD[3], *shuffling* method, *recursive reshuffling* and the use of the MiniSat feature. SAT problems are divided into satisfiable problems and unsatisfiable problems. If we mix solvers strong for satisfiable problems with solvers strong for unsatisfiable problems, this might lead to better results. MiniSat is still competitive on satisfiables problems and mixture of Glueminisat with MiniSat can be simply conducted by option because MiniSat is the base solver of Glueminisat.

We allocate Glueminisat to half workers and MiniSat to the other half workers. Workers of Glueminisat shares clauses under LBD = 2, while workers of MiniSat only shares unit clauses. When the number of cores exceed 24, *recursive reshuffling* is applied to some workers.

References

[1] Gilles Audemard, Laurent Simon. Experimenting with Small Changes in Conflict-Driven Clause Learning Algorithms. Principles and Practice of Constraint Programming, pp 630-634, 2008.
[2] Long Guo, Youssef Hamadi, Said Jabbour, Lakhdar Sais. The Diversification and Intensification in Parallel SAT Solving. Principles and Practice of Constraint Programming, pp 252-265 2010.
[3] Gilles Audemard, Laurent Simon. Predicting Learnt Clauses Quality in Modern SAT Solvers. In Proceedings of the 21st international jont conference on Artifical intelligence, 399-404, 2009.