MIXED PARALLEL EXECUTION OF ALGORITHMS FOR SATISFIABILITY PROBLEM

Kairong Zhang and Masahiro Nagamatu
Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, Kitakyushu, Japan

Abstract: LPPH has been proposed to solve the satisfiability problem (SAT). In order to solve the SAT more efficiently, a parallel execution has been proposed. Experimental results show that higher speedup ratio is obtained by using this parallel execution of the LPPH. In this paper, we propose a method of mixed parallel execution of several algorithms for the SAT. "Mixed" means the parallel execution of the LPPH and local search algorithms. In the experiments, we used the LPPH with attenuation coefficient generating function and the GSAT. Results of experiments show mixing these two algorithms yield excellent performance.

Key words: Satisfiability problem; parallel execution; neural network; Lagrangian method

1. INTRODUCTION

We have proposed a parallel execution of the LPPH[1,2,3], and experimental results show that higher speedup ratio is obtained by using this parallel execution of the LPPH. In this paper, we propose a mixed parallel execution of the LPPH and local search algorithms. We did experiments in which the LPPH with attenuation coefficient generating function[4] and the GSAT[5] are used. Results of the experiments show the mixed parallel execution can be more efficient than the parallel execution of the LPPH only or the parallel execution of the GSAT only for many problems.

2. PARALLEL EXECUTION OF LPPH

We have proposed a parallel execution of the LPPH: (1) Prepare plural neural networks of the LPPH. (2) Start the LPPHs simultaneously from different initial points from each other. (3) When any of the LPPHs finds a solution, halt all LPPHs and return the solution. It is very easy to realize this parallel execution of the LPPH by hardware. Only we have to do is preparing plural neural networks. The total system is very simple and executable at high-speed.

Suppose that a parallel execution of p LPPHs is done. Let t_j be the execution time of jth LPPH for finding a solution. Then, $T_p = \min\{t_j \mid 1 \le j \le p\}$ is the execution time of the parallel execution. We will call T_p and pT_p, "execution time" and "total execution time", respectively. Sometimes we will use "number of updates" instead of "execution time", because these are proportional when the problem is fixed. The result of experiment is shown in Fig.1. The horizontal axis indicates the number of LPPHs, and the vertical axis indicates the speedup ratio, namely $E(T_p)/E(T_1)$, where E means the average. In this experiment, parallel execution of p (p=1, 2, ... , 50) neural networks of LPPH is used. Randomly generated 3-SAT problems are used in this experiments. They are exp-r300 (300 variables and 1275 clauses), exp-r200 (200 variables and 860 clauses), exp-r100 (100 variables and 430 clauses), and exp-r50 (50 variables and 215 clauses). From Fig.1, it is shown that higher speedup ratio is obtained. This is remarkable for large and difficult problems, e.g. exp-r300.

3. MIXED PARALLEL EXECUTION OF LPPH AND GSAT

The local search algorithms find solutions from randomly selected initial points. Therefore they can be used in our parallel execution. In this paper, we propose a mixed parallel execution of local search algorithms and the LPPH as follows:

(1) Prepare plural solvers of the SAT, such as the LPPH and local search algorithms. These solvers may be neural networks, electronic circuits or software processes executed on computers.

(2) Start the solvers simultaneously from different initial points.

(3) When any of the solvers finds a solution, halt all solvers and return the solution.

The mixed parallel execution also has the same advantage with the parallel execution of the LPPH, and also do not need any communication overhead.

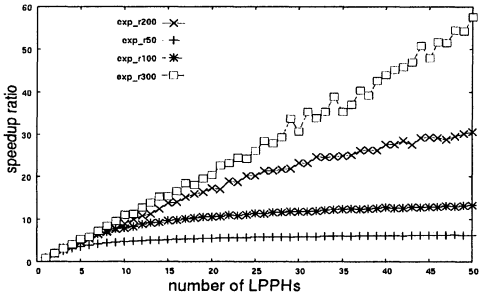

Figure1. The speedup ratio for random 3-SAT

4. EXPERIMENT

We did experiments which use the LPPH with attenuation coefficient generating function and the GSAT for the mixed parallel execution. Results of experiments are shown in Figure 2 and Figure 3. Figure 2 and Figure 3 are the results of a random 3-SAT of 200 variables and 860 clauses, and a 20-Queen problem, respectively. In these graphs, horizontal axes indicate the number of solvers and vertical axes indicate the average of the total CPU times of 200 trials. In these graphs, 3 types of parallel execution are compared. In the first type, all solvers are the LPPH with attenuation coefficient generating function. We call this type PA_{LPPH}. In the second type, all solvers are GSAT (PA_{GSAT}). In the third type, they are mixed into halves (PA_{mixed}). For many problems we have results similar to Figure2 and Figure 3. From these results, we can say PA_{LPPH} is more efficient than PA_{GSAT}, if p is small, and PA_{mixed} is at least as efficient as the better one of PA_{GSAT} and PA_{LPPH}. Furthermore, it is not so rare, to our surprise, that PA_{mixed} becomes more efficient than PA_{GSAT} and PA_{LPPH} as shown in Figure 3.

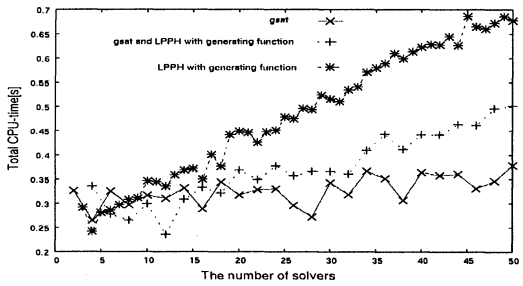

Figure 2. Comparison of several parallel execution methods for random 3-SAT of 200 variables and 860 clauses

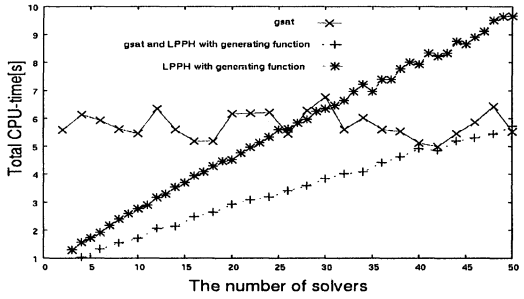

Figure3. Comparison of several parallel execution methods for 20 —queen

5. CONCLUSION

It is easy for the LPPH to achieve a parallel processing when neurons are implemented individually by electronic circuits or on different computers (neuron-level parallel processing). Many researches of neuron-level parallel processing had been done. In this case several kinds of communication overheads are needed, such as space overhead or time overhead.

We have proposed a parallel execution of the LPPH (we call this a network-level parallel processing). This method achieves the high speedup ratio and low communication overhead. In this paper, we extend this to the mixed parallel execution of the LPPH and the GSAT. The experimental results show that for many problems the mixed parallel execution is at least as efficient as the better one of the two non-mixed parallel executions. Furthermore, for some problems, the mixed parallel execution becomes more efficient than non-mixed parallel executions. The future work for us, is to study about the reason of the efficiency of mixed parallel execution.

REFERENCE

1. M. Nagamatu and T. Yannaru, "On the stability of Lagrange programming neural networks of satisfiability problems of propositional calculus", Neurocomputing, 13, 119-133, 1995.
2. M. Nagamatu and T. Yannaru, "Parallel state space search for SAT with Lagrange Programming Neural Network", proceedings of the fifth International Conference on Neural Information Processing. October, 1998
3. K. Zhang and M. Nagamatu, "Parallel execution of neural networks for solving SAT" (to appear).
4. K. Zhang and M. Nagamatu, "Solving SAT by execution of neural network with probabilistic attenuation coefficient generator" (to appear)

5. B.Selman, H. Levesque and D.Mitchell, "A new method for solving hard satisfiability problem," AAA-92, Proceedings Tenth National Conference on Artifical Intelligence, pp.440-6,1992