Comparative Evaluation of Parallelization Strategies for Evolutionary and Stochastic Heuristics

Sadiq M. Sait Syed Sanaullah Ali Mustafa Zaidi Mustafa I. Ali
College of Computer Sciences & Engineering
King Fahd University of Petroleum & Minerals
Dhahran, Saudi Arabia
{sadiq,sanaulla,alizaidi,mustafa}@ccse.kfupm.edu.sa

ABSTRACT

In this paper we present an evaluation of selected parallel strategies for Simulated Annealing and Simulated Evolution, identifying the impact of various issues on the effectiveness of parallelization. Issues under consideration are the characteristics of these algorithms, the problem instance, and the implementation environment. Observations are presented regarding the impact of parallel strategies on runtime and achievable solution quality. Effective parallel algorithm design choices are identified, along with pitfalls to avoid. We further attempt to generalize our assessments to other heuristics.

Categories and Subject Descriptors

D.1.3 [**Programming Techniques**]: Concurrent Programming—*Parallel programming*

General Terms

Algorithms, Performance, Experimentation, Theory

Keywords

Metaheuristics, Parallel Processing, Parallel Algorithms, Combinatorial Optimization, Simulated Annealing, Simulated Evolution

1. INTRODUCTION

Parallelization of iterative heuristics aims to solve large problems and traverse larger search spaces in a reasonable amount of time [2]. The goals of parallelization can be to achieve either lower runtimes for same quality solutions or higher quality solutions in limited time. Achieving these goals requires proper partitioning of the problem for uniform distribution of computationally intensive tasks, while respecting the underlying implementation environment. The tractability of this is largely dependent on both the parallelizability of the cost computation and perturbation functions. But most importantly, achieving good quality solutions requires a thorough and intelligent traversal of a complex search space. Thus, for a parallelization strategy to be truly effective, its interaction with the 'intelligence' of the heuristic must be considered, as it directly affects the final solution quality obtainable, and also indirectly the runtime because of the time required for convergence. This is one of the key observations that we discuss in this paper. The goal is to present a qualitative analysis of the results of parallelization strategies for two heuristics, namely Simulated Annealing (SA) and Simulated Evolution (SimE), for the multi-objective standard cell placement problem [4]. To this end, we apply the preferred parallelization schemes identified in literature for a distributed memory (DM) environment. Based on our results, we present an evaluation of each strategy with a special focus on its interaction with the algorithmic intelligence. Finally, we mention how our assessments are applicable to evaluating parallelization strategies for other stochastic heuristics.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright is held by the author/owner.
GECCO'05, June 25–29, 2005, Washington, DC, USA.
ACM 1-59593-010-8/05/0006.

2. PARALLELIZATION STRATEGIES

Parallel SA (PSA): For SA [5] we use a parallel search approach using asynchronous multiple Markov chains (AMMC) strategy mainly because it involves minimal communications [1]. We experiment with two versions of AMMC, varying the work done by each processing element (PE). Strategy 1 (S1) aims to enhance quality, while strategy 2 (S2) attempts to achieve linear speedup, while retaining quality.

Distributed SimE (DSE): SimE [5] parallelization is carried out by partitioning workload among available PEs [3]. Several rows of the placement are assigned to individual PEs in an alternating fashion. Besides the original row allocation pattern (FRA), we have also experimented with random row allocation (RRA).

3. EXPERIMENTS AND ANALYSIS

In our experiments with PSA (Table 1), solution qualities from S1 always exceed those of the serial algorithm, though speedups are sub-linear. The results of S2 show a roughly 10% quality drop compared with S1 but the speedups are almost linear. Thus, of the two goals of parallelization, we are only successful in attaining the first i.e. quality improvement for fixed runtime. The DSE failed to fulfill both goals. As PEs are increased, degradation in achievable quality is

Table 1: Results of parallel SA for strategies 1 & 2 and distributed SimE for RRA & FRA. Due to space limitation only results for up to 5 processors (6 for PSA including master processor) are shown. Due to quality degradation in DSE with increase in processors, the target serial quality chosen for DSE was that achieved by p=5 for RRA. Quality fixed column indicates the % of target quality achieved by FRA for p=5.

Circuit Name	# of Cells	Solution Quality	Runtimes for Serial SA	Runtimes for Parallel SA Strategy 1 (sec)				Solution Quality (S2)	Runtimes for Parallel SA Strategy 2 (sec)			
				p=3	p=4	p=5	p=6		p=3	p=4	p=5	p=6
s1238	540	0.69947	212	183.91	130.32	127.55	117.12	0.630573	58.03	39.21	26.31	22.31
s1488	667	0.65038	275	151.46	118.44	112.59	98.94	0.582884	42.67	25.59	18.77	16.61
s1494	661	0.64792	214	131.4	116.27	101.89	98.13	0.591114	51.11	30.79	22.32	15.82
s3330	1961	0.79344	2137	1875.52	1658.6	1572.94	1419.51	0.720665	528.38	403.31	351.27	307.42
Circuit Name	# of Cells	Runtimes for Serial SimE		Runtimes for Random Row Distribution (sec)				% Sol. Quality Fixed	Runtimes for Fixed Row Distribution (sec)			
				p=2	p=3	p=4	p=5		p=2	p=3	p=4	p=5
s1238	540	16.5		9.24	9.29	6.12	3.14	95.80%	17.83	8.47	11.3	5.71
s1494	667	67		17.4	6.15	4.88	5.89	82.30%	2.77	1.85	1.76	4.34
s1488	661	60.23		24.6	7.78	3.72	3.02	96.60%	22	4.89	5.1	16
s3330	1961	UH		678.02	115	108.5	49.14	33.80%	316	215	4.6	3.4

observed. The solution qualities achieved using RRA are always superior to FRA, but fall well short of the quality achieved by the serial SimE. We now present an evaluation of these parallelization strategies highlighting the effect of various important factors.

Cost-Computation Function Complexity: The multiobjective cell placement problem was selected because it is computation intensive and difficult to distribute. Since the cost computation cannot be subdivided among processes, the amount of work done in cost computation per PE per iteration in DSE is not reduced with increase in PEs, resulting in poor run time scalability. Parallel search for SA, however, is immune to this problem as each thread undertakes an independent search.

Parallelization Environment: The use of barrier synchronization in DSE, coupled with linear increase in communication with increasing PEs causes a bottleneck. This effect is particularly important for a DM environment. In contrast, the asynchronous communication model in PSA minimizes the impact of communication.

Solution Perturbation and Selection: The solution perturbation and next-solution selection operations are where the intelligence of virtually all stochastic heuristics lies. Parallel search [2] is a well known technique for enhancing the robustness of heuristics, giving improved qualities with some runtime reduction. This is evident from the results of S1. However, to achieve linear speed-ups, some reduction in work with each PE proportional to number of PEs is required. The intelligence of SA lies in its cooling schedule. This explains why our attempts at achieving speed-ups just by dividing the iterations in Metropolis by number of PEs achieves good speed-ups but a drop in final quality achieved as this results in a less thorough parallel search of the neighboring solution space. In SimE, both evaluation and selection of each cell may be carried out independently of the other cells. However, the allocation step requires that cells be allocated in a certain order, thereby introducing a sequential dependence. Since parallelization of SimE is achieved by dividing the solution between PEs, movement of individual cells across the solution during the allocation step is restricted due to the lack of global placement view for each PE. This results in a division of the heuristic intelligence and consequently has an adverse effect on achievable solution quality. The effects of restrictive cell movement can be alleviated by using a better row allocation pattern that allows the cell to move to their intended best locations and as quickly as possible. Use of a pattern that facilitates a variety of combination among the rows sounds intuitively better. However, even our attempt to improve cell movement by using RRA falls short of improving solution qualities sufficiently.

4. CONCLUSION AND FUTURE WORK

The choices of effective parallelization strategies (flexibility) are governed by the amount of inherent/true parallelism available in the problem instance and its cost functions. Achieving near-linear speedups while maintaining solution qualities appear to be conflicting objectives, particularly for algorithms with such a strong sequential dependence such as SA and SimE. Our experiments have shown that this effect is primarily due to the division of intelligence that occurs when speedup is emphasized. In order to achieve solution qualities similar to or better than serial algorithms while providing near linear speedup, a parallelization strategy for any heuristic must be qualified in terms of any limitations it imposes on the algorithm's intelligence and effort must be made to counter these limitations. We are currently working on a comprehensive analysis/survey along these lines of various parallelization schemes for other heuristic algorithms and a variety of problem instances.

5. ACKNOWLEDGMENT

The authors would like to thank King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, for support under project code # COE/CELL PLACE/263.

6. REFERENCES

[1] J. A. Chandy, S. Kim, B. Ramkumar, S. Parkes, and P. Banerjee. An Evaluation of Parallel Simulated Annealing Strategies with Application to Standard Cell Placement. *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, 16 No.4, April 1997.

[2] V.-D. Cung, S. L. Martins, C. C. Riberio, and C. Roucairol. Strategies for the Parallel Implementation of Metaheuristics. *Essays and Surveys in Metaheuristics*, pages 263–308, Kluwer 2001.

[3] S. M. Sait, M. I. Ali, and A. M. Zaidi. Multiobjective VLSI Standard Cell Placement using Distributed Simulated Evolution Algorithm. *Proceedings of the Intl. Symp. On Circuits & Systems (ISCAS)*, May 2005.

[4] S. M. Sait, M. R. Minhas, and J. A. Khan. Performance and Low-Power Driven VLSI Standard Cell Placement using Tabu Search. *Proceedings of the 2002 Congress on Evolutionary Computation*, 1:372–377, May 2002.

[5] S. M. Sait and H. Youssef. Iterative Computer Algorithms and their Application to Engineering. *IEEE Computer Society Press*, December 1999.