Relaxing Synchronization in Distributed Simulated Annealing

Bruce M. McMillin
Missouri University of Science and Technology, ff@mst.edu

Chul-Eui Hong

The number of possible cases for evaluating the predicate of the branching is drastically reduced in these special cases. By the commutativity of multiplication (for Case I), by the commutativity of addition (for Case II), and by induction (for Case III) there are i possible cases of the predicate evaluation at ith iteration. This reduced number of possible cases of the predicate evaluation and the constant coefficients naturally simplify the precomputation to set up the binary tree representation and the path selection from the tree.

We parallelize these special cases after strip mining of $\sqrt{p}$ iterations at a time with p processors. By taking $i\, e_{i,u}$'s at the ith level from its binary tree representation where $u = 2^{k-1}\,(1 \leq k \leq i)$, the path selection for a partition can be done in approximately $\log p$ time with p processors. The precomputation for a partition can be done also in $O(\log p)$ time, because there are $\sqrt{p}$ first-order linear recurrences of size $\sqrt{p}$. So, these simple forms of first-order linear mixed recurrence loops can be parallelized with speedup proportional to $n/\log n$ with n^2 processors.

Using Parafrase [3], we have experimented our way of parallelizing the special cases of linear mixed recurrence loops with some basic nonnumerical algorithms. Nine out of fifteen algorithms experimented possess conditional cyclic loops (see [7] for the list of algorithms and the loop structures in them). For the nine algorithms with conditional cyclic loops, 80% of the conditional cyclic loops were the special cases of linear mixed recurrence loop. The average speedup improved to 12.36 from 5.04 for 32 processors (see [7] for the speedup for each individual algorithm).

VI. Conclusion

We have considered parallelizing conditional cyclic loops on a shared memory multiprocessor allowing concurrent reads. Based on a binary tree representation of a conditional cyclic loop, executing the loop requires precomputing all the possible values of predicate variables and selecting a single path from the root of the tree. Precomputing the predicate variables requires solving a set of recurrence relations and selecting a path from the tree requires solving a full-order nonlinear Boolean recurrence. Solving all the recurrences incurs a tremendous overhead resulting in little speedup gain of $O(\log p/\log\log p)$ from parallelizing the loop with p processors.

Fortunately, most conditional cyclic loops encountered in practice are of simpler forms, either postfix-IF or the three special cases reported. With simpler forms, the number of possible cases of predicate evaluations for conditional branching is reduced drastically to $O(n)$ from $O(2^n)$ in general form of conditional cyclic loops, where n is the loop bound. This naturally makes our way of parallelizing the loop more efficient, resulting $O(p/\log p)$ speedup with p processors.

Little benefit derived from the parallelization makes it impractical to parallelize conditional cyclic loops in general. Furthermore, precomputing all possible values of predicate variables in conditional branching may cause undesirable side effect of spurious arithmetic faults. We suggest not to resort solely on automatic program restructuring for significant gain of speedups, if mixed recurrence loops in general form are major loops of a program to be restructured.

References

[1] U. Banerjee and D. Gajski, "Fast evaluation of loops with IF statement," *IEEE Trans. Comput.*, vol. C-33, pp. 1030–1033, 1984.

[2] S. Chen, D. Kuck, and A. Sameh, "Practical band triangular system solvers," *ACM Trans. Math. Software*, vol. 4, no. 3, pp. 270–277, Sept. 1978.

[3] D. J. Kuck, A. Sameh, R. Cytron, A. Veidenbaum, C. Polychronopoulos, G. Lee, T. McDaniel, B. Leasure, C. Beckman, J. Davies, and C. P. Kruskal, "The effects of program restructuring, algorithm change, and architecture choice on program performance," in *Proc. 1984 Int. Conf. Parallel Processing*, Aug. 1984, pp. 129–138.

[4] D. J. Kuck, *The Structure of Computers and Computations, Vol. I.* New York: Wiley, 1978.

[5] R. H. Kuhn, "Optimization and interconnection complexity for: Parallel processors, single-stage networks, and decision trees," Ph.D. dissertation, Dep. Comput. Sci., Univ. Illinois, Urbana-Champaign, 1980.

[6] R. E. Ladner and M. J. Fischer, "Parallel prefix computation," *J. ACM.*, vol. 27, no. 4, pp. 831–848, Oct. 1980.

[7] G. Lee, C. Kruskal, and D. J. Kuck, "An empirical study of automatic restructuring of nonnumerical programs for parallel processors," *IEEE Trans. Comput.*, vol. C-34, pp. 927–933, 1985.

[8] I. Munro and M. Patterson, "Optimal algorithms for parallel polynomial evaluation," *J. Comput. Syst. Sci.*, vol. 7, 1973.

[9] A. Sameh and R. Brent, "Solving triangular systems on a parallel computer," *SIAM J. Numerical Anal.*, vol. 14, no. 6, pp. 1101–1113, Dec. 1977.

[10] R. A. Towle, "Control and data dependence for program transformations," Ph.D. dissertation, Dep. Comput. Sci., Univ. Illinois, Urbana-Champaign, 1976.

[11] M. J. Wolfe, "Optimizing super compilers for supercomputers," Ph.D. dissertation, Dep. Comput. Sci., Univ. Illinois, Urbana-Champaign, 1982.

Relaxing Synchronization in Distributed Simulated Annealing

Chul-Eui Hong and Bruce M. McMillin

***Abstract*—This paper presents a cost error measurement scheme and relaxed synchronization method, for simulated annealing on a distributed memory multicomputer, which predicts the amount of cost error that an algorithm will tolerate. An adaptive error control method is developed and implemented on an Intel iPSC/2.**

***Index Terms*—Optimization, partial synchrony, state maintenance, cost error, parallelism, algorithms.**

I. Introduction

The simulated annealing algorithm is based on the analogy between simulation of the annealing of solids and the problem of solving large combinatorial optimization problems [7]. Reference [9] proposes a Monte Carlo method, which simulates the evolution to thermal equilibrium of a solid for a fixed value of the temperature T. In implementation of simulated annealing, the initial temperature is set sufficiently high so that all moves are accepted. With a small perturbation of the current state space, we can reach a new state. Let ΔC be the difference of the energies (cost) of current state and new state, or *new cost-old cost.* The probability that a candidate move

Manuscript received December 21, 1992; revised March 21, 1994. This work was supported in part by the National Science Foundation under Grants MSS-9216479 and CDA-9222827, and, in part, from the Air Force Office of Scientific Research under Contracts F49620-92-J-0546 and F49620-93-1-0409.

C.-E. Hung is with the Electronics and Telecommunications Research Institute, Taejon 305-606, Korea.

B. McMillin is with the Department of Computer Science, University of Missouri-Rolla, Rolla, MO 65401 USA.

IEEE Log Number 9408139.

is accepted or rejected in simulated annealing is determined by *the Metropolis criterion*:

$$\text{Prob}\,[\Delta C\,\text{is accepted}] = \min\left(1, \exp\left(-\frac{\Delta C}{T}\right)\right). \tag{1-1}$$

Evaluation of the cost change (ΔC) is expensive due to the large number of state parameters that need to be evaluated. Parallelization of the annealing procedure is an attractive option. In particular, distributed memory multicomputers provide the best promise in massive performance speedup. A multicomputer consists of individual processors with local memory that communicate by message passing over an interconnection network. There is no shared memory available for maintenance of a global system state, thus we may have inconsistent state views among the processors unless expensive synchronization is performed before each evaluation of a cost change. Using inconsistent states to calculate the cost change may result in a *cost error.*

When the new cost is larger than the current cost, i.e., $\Delta C > 0$, this proposed move is called a *hill climbing move* in simulated annealing. Since simulated annealing randomly selects hill climbing moves, it can tolerate some cost errors. Under the proper conditions, annealing algorithms can evaluate the cost using old state information, and still converge to a reasonable solution. Thus, it is important to find an upper bound on the cost error at a particular temperature so that we can maximize the amount of independent work processors can do, and, thereby, increase the parallel speedup in the parallel implementation. So cost error tolerance plays a useful role in multiprocessing. When processors independently operate on different parts of the problem, they need not synchronously update other processors. A processor can save several changes, and then send a single block to the other processors.

Previous work on cost-error-tolerant schemes cannot measure the cost error correctly, and cost error has been tolerated empirically. In this paper, we define maximum bound of tolerable cost error as a function of the global update frequency. The method of this paper is applicable to any combinatorial optimization problems. However, the spatial decomposition (data parallel domain decomposition) is chosen as a model problem. In a data parallel domain decomposition, each processing node of the multicomputer receives a subdomain of the entire problem such that the workload is balanced among all processors. The specific model problem chosen here is the stock-cutting problem [8]. The stock-cutting problem is to allocate regular and/or irregular patterns onto a large stock sheet of finite dimensions in such a way that the resulting scrap will be minimized. The pattern is a physical object to be packed and cut out. A sample pattern placement is shown in Fig. 1. This problem is common to many applications in aerospace, shipbuilding, VLSI design, steel construction, shoe manufacturing, clothing and furniture manufacturing. The cost function corresponding to this problem is made up of an affinity relation between patterns, the distance from the origin, and overlap penalty between patterns. Consider the cost function C as being

$$C = -\alpha \sum \frac{a_{i,j}}{d_{i,j}} + \beta \sum d_{i_0} + \gamma \sum O_{i,j} \tag{1-2}$$

where α, β, and γ are positive real numbers that indicate the contribution of each of the components in the cost function. $a_{i,j}$ is the affinity relation, or strength of attraction, between pattern i and j. $d_{i,j}$ is the distance between pattern i and j. d_{i_0} represents the distance of pattern i from the origin. $O_{i,j}$ is the overlap between pattern i and j.

A data parallel decomposition of the stock cutting problem gives each node approximately the same number of patterns. Each node performs internal move, rotate and exchange operations as well as participating in moves between nodes. The distance from origin, $\Sigma\, d_{i_0}$, is calculated correctly without the global information of location of all object since the origin is fixed. However, in calculating the affinity relation between two patterns i and $j, \Sigma\,(a_{i,j}/d_{i,j})$, we need global information of the correct location of two patterns. Without this global information, calculation of the affinity relation, and, thus, the cost function, C, incurs a cost error.

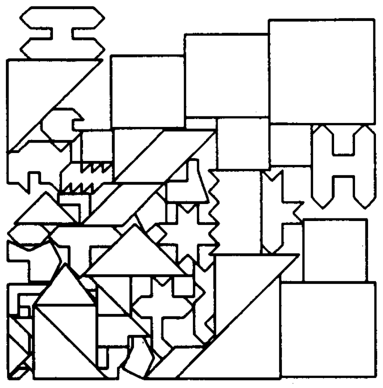
Fig. 1. A sample placement for 50 patterns.

Section II discusses previous work on cost-error-tolerant schemes. In Section III, we classify the error model by case study, and present our cost error measurement scheme and error-tolerant method. This cost-error-tolerant method is applied to the stock-cutting problem using an asynchronous parallel spatial decomposition simulated annealing algorithm. Finally, Section IV presents and discusses some experimental results.

II. Previous Work on Error Tolerance

[5] describes the characteristics of cost errors at different temperatures. The error in the cost function is defined to be the difference between the real change in cost from initial to final states and the estimated change in cost, which is equal to the sum of the changes in cost at each processor (ΔC_i for processor i).

Definition 2-1: The *cost error* (ΔE_{tr}) is defined as the difference between the actual (real) cost change and the estimated (measured) cost change.

$$\Delta E_{tr} = \Delta C_a - \Delta C_e = (C_{af} - C_{ai}) - \sum_{i=1}^{P} \Delta C_i \tag{2-1}$$

where ΔC_a is the actual cost change, ΔC_e is the estimated cost change. C_{af} is the actual final cost and C_{ai} is the actual initial cost. ΔC_i is the estimated cost change in processor i, and P is the total number of processors.

This cost error measurement scheme will be referred to as *the traditional error measurement scheme.* There are shortcomings in this traditional error measurement scheme. This method counts only the accepted moves because there is no way to calculate the actual cost without global information. The second problem is that when both the actual and the estimated cost change are negative, even though the acceptance of the move is correct, the difference in cost, $\Delta C_a - \Delta C_e$, is added to the total amount of cost error. Finally, the optimistic and the pessimistic errors are compensated. These three problems are corrected by the cost error measurement scheme proposed in the next section.

Definition 2-2: The *stream length* s is defined as the number of continuous moves before the global update where all local information is broadcast and updated.

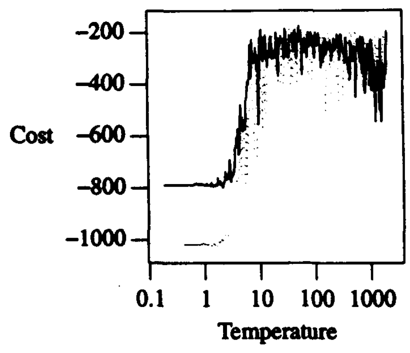

Fig. 2. Fluctuations of cost.

Previous methods, such as [1], [3], set the upper bound on the maximum permissible cost error empirically. For example, [3] suggests that cost errors which are much smaller than the temperature do not change the results of the algorithm. In statistical mechanics, all macroscopic properties of a material can be derived from the partition function z, which is defined as the sum of the Boltzmann factors over all possible states, $z = \Sigma_{i \in S} \exp(-[C(i)/T])$. With this method, the maximum stream length in a fixed temperature can be probabilistically predicted based on the expected magnitude of a cost error.

Definition 2-3: When the cost error is positive, the acceptance ratio is increased. This kind of error is called an *optimistic error* and this move an *optimistic move.* A *pessimistic error* occurs when the cost error is negative. This being the reverse case of an optimistic error, the acceptance ratio is decreased in the case of a pessimistic error.

[1] observed that the annealing curve is similar to the acceptance curve. If the acceptance ratio with cost error is held to within 5 percent of a normal distribution, a pessimistic cost error bound B_+ and an optimistic cost error bound B_- are approximated as follows:

$$\begin{aligned} B_+ &\leq -T \cdot \ln(1 - 0.05) \approx T/20 \\ B_- &\leq T \cdot \ln(1 + 0.05) \approx T/21. \end{aligned} \tag{2-2}$$

If the average cost error after a stream length is higher than $(T/21)$, the stream length is reduced to commensurate with that excess. If average cost error is lower than $(T/42)$, the stream length is increased slowly. A 5% deviation in composite acceptance is set experimentally to maintain convergence.

III. A New Error Tolerance Method

In this section, a new cost error measurement scheme is presented and the maximum cost error bound for a specific temperature is set based on the hill climbing nature of simulated annealing. Using the measured amount of cost error and the maximum cost error bound, an optimal stream length s is derived (Definition 2-2).

Definition 3-1: Hill climbing power(or depth) is the degree of accepting the hill climbing move.

The rationale for deriving this bound is based on the observation (Fig. 2) that as the stream length s increases, the hill climbing power decreases since the fluctuations in cost reduce in the error-present annealing process. The decreased hill climbing power can be compensated for by an increased additional Markov chain length. Since the cost error increases as the stream length increases, the optimal stream length and the additional Markov chain length are proportional to retain the same convergence of the sequential (error-free) annealing process.

In development of the theory, we need to determine the distribution of the move acceptance and of the erroneous move decision.

Theorem 3-1: The acceptance move decision is exponentially distributed with respect to the parameter $T > 0$.

$$\text{Prob}\,[\text{Move accepted with cost change } [0, \Delta C]] = 1 - e^{-(\Delta C/T)}$$

Proof: Define the continuous random variable X to be a function which associates a positive real number, the hill climbing cost change (ΔC) with each possible outcome of an accepted move decision. The probability of move acceptance is $e^{-}(\Delta C/T)$ when the cost change is $(\Delta C, \infty)$ from the Metropolis criterion (1-1). So the cumulative probability of move acceptance is $1 - e^{-}(\Delta C/T)$ when the cost change is $[0, \Delta C]$, which is the exponential cumulative distribution function.

$$\text{Prob}\,[X \leq \Delta C] = 1 - e^{-(\Delta C/T)} \qquad \text{from (1-1)}$$

So the continuous random variable X has an exponential distribution with respect to the parameter $T > 0$. □

Theorem 3-2: The erroneous move decision is exponentially distributed with respect to the parameter $T > 0$, given that the candidate move is accepted with a smaller cost change between the actual and the estimated cost changes.

$$\text{Prob}\,[\text{The erroneous move decision with cost error } [0, \Delta E]] = 1 - e^{-(\Delta E/T)}$$

Proof: By applying the excess life to the move acceptance and using the memoryless property of the exponential distribution, the erroneous move decision can be proved to be distributed exponentially. The detail of proof is in [4]. □

In the following, the move decisions are classified according to the actual cost change (ΔC_a) and the estimated cost change (ΔC_e) with corresponding probabilities of an erroneous move decision calculated from Theorem 3-2. Next, we refine traditional cost-error measurement (Definition 2-1) by correcting three problems.

A. Case-by-Case Study of Error Model

There are 4 possible cost change cases (*Case 3-1* through *Case 3-4*) each with 4 possible sub-cases, that is:

1) A move is accepted based on an estimated cost change, yet will be rejected based on an actual cost change.
2) A move is rejected based on an estimated cost change, yet will be accepted based on an actual cost change.
3) A move is accepted based on an estimated cost change and also based on an actual cost change.
4) A move is rejected based on an estimated cost change and also based on an actual cost change.

In subcases 1) and 2), an erroneous move decision occurs due to the cost error. However, in subcases 3) and 4), the move decision is correct regardless of the cost error measurement used.

Case 3-1: $\Delta C_e \geq \Delta C_a > 0$ (*Pessimistic move* from Definition 2-3).

Define one move error $\Delta E = \Delta C_e - \Delta C_a$, where $\Delta E \geq 0$

1) This subcase is not applicable because $\Delta C_e \geq \Delta C_a$.
2) The move is rejected with the estimated cost change ΔC_e, where the probability of a move rejection is $1 - \exp(-[\Delta C_e/T])$. However, this move can be accepted with

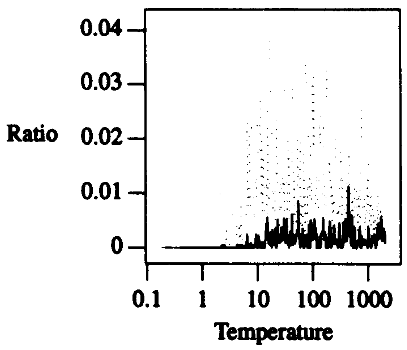

Fig. 3. The ratio of # of Case 3-3 to # of total moves in 4 processors.

TABLE I
PROBABILITY OF THE COST ERROR IN A HILL CLIMBING MOVE

	Pessimistic Move($\Delta C_e \geq \Delta C_a$)		Optimistic move($\Delta C_a \geq \Delta C_e$)	
Move	acc w/ ΔC_e	rej w/ ΔC_e	acc w/ ΔC_e	rej w/ ΔC_e
acc w/ ΔC_a	0	$e^{-\frac{\Delta C_e}{T}} \cdot \left(e^{\frac{\Delta E}{T}} - 1\right)$	0	-
rej w/ ΔC_a	-	0	$e^{-\frac{\Delta C_e}{T}} \cdot \left(1 - e^{-\frac{\Delta E}{T}}\right)$	0

the actual cost change ΔC_a with a probability P_1, where

$$\begin{aligned} P_1 &= \text{Prob}\,[\text{Move accepted with}\, \Delta C_a \cap \text{Move rejected with}\, \Delta C_e] \\ &= e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1) \qquad \text{from Theorem 3-2} \end{aligned}$$

Case 3-2: $\Delta C_a \geq \Delta C_e > 0$ (*Optimistic move* from Definition 2-3).

We define one move error $\Delta E = \Delta C_a - \Delta C_e$, where $\Delta E \geq 0$

1) This move will be accepted based on the estimated cost change but this move can be rejected with the actual cost change ΔC_a with a probability P_2, where

$$\begin{aligned} P_2 &= \text{Prob}\,[\text{move accepted with}\, \Delta C_e \cap \text{move rejected with}\, \Delta C_a] \\ &= e^{-(\Delta C_e/T)} \cdot (1 - e^{-(\Delta E/T)}) \qquad \text{from Theorem 3-2} \end{aligned}$$

2) This subcase is not applicable because $\Delta C_a \geq \Delta C_e$.

Case 3-3: $(\Delta C_e > 0 \cap \Delta C_a \leq 0) \cup (\Delta C_a > 0 \cup \Delta C_e \leq 0)$

In *Case 3-3*, the cost error is greater than the absolute value of the estimated cost change. Since computing these two probabilities is somewhat complex and the ratio of this case to the total attempted moves in the stock cutting problem is less than 4% (Fig. 3), the occurrence of these events is ignored. In VLSI placement, the maximum error bound is $T/21$ from (2-2), so we can guess that the erroneous decision is less than 5% even though the cost error is dependent on the specific problem.

Case 3-4: $\Delta C_e \leq 0 \cup \Delta C_a \leq 0$.

In *Case 3-4*, a move will be always accepted and there is no cost error, since the decision of move is correct, i.e. move is accepted also with the actual cost change ΔC_a.

The summary of the above cases are in Table I, where ΔE is the amount of the cost error.

B. Improved Error Measurement Scheme

In improving the cost error measurement scheme, the total amount of cost error is calculated throughout a given stream length, s, based on the probabilistic analysis of the previous section. ΔE is the cost error of any iteration i and $|\langle E \rangle|$ is the average cost error in one move. On the average, $|\langle E \rangle|$ is calculated from 1/2 the distance of the maximum spatial distance move a processor may make. This is a worst case assumption, since moves with smaller distances are accepted more frequently. Then the cost error ΔE at an iteration i can be represented as an average error ($|\langle E \rangle|$) times the acceptance ratio (α) times the number of iterations, i.

Lemma 3-1: The actual cost change (ΔC_a) is represented as the sum of the estimated cost change (ΔC_e) and the cost error during i iterations without global update.

$$\Delta C_a \equiv \Delta C_e \pm \Delta E = \Delta C_e \pm i \cdot \alpha \cdot |\langle E \rangle|.$$

Proof: The proof is obvious from the above statements □

Since a cost error occurs only with a positive cost change in this analysis, to calculate the cost error, it is necessary to compute a probability for the conditions of *Case 3-1* and *Case 3-2*.

Lemma 3-2: The probability of positive estimated cost change is

$$\text{Prob}\,[\Delta C_e > 0] = \frac{\sum\limits_{i \neq j} I_{\{\Delta C_e(i,j) > 0\}}(i)}{N(i)}, \qquad \text{for any state}\, i.$$

Proof: State j is any neighbor of state i. $\Delta C_e(i,j)$ is the estimated cost change of a move from state i to state j. $N(i)$ is the number of neighbor states from state i. The proof is obvious by using the specified indicator I. □

Since $N(i)$ and $\Delta C_e(i,j)$ are not known in advance, it is difficult to estimate the Prob $[\Delta C_e > 0]$. However, during the running of the algorithm, the estimated cost change can be calculated. So only when the estimated cost change is greater than zero, the total probability of cost error is counted.

We can now express the probability of an optimistic error and a pessimistic error in terms of the estimated cost change (ΔC_e) and cost error (ΔE).

Corollary 3-1: The probabilities of optimistic cost error, P_{opt}, and pessimistic cost error, P_{pes}, are respectively

$$\begin{aligned} P_{opt} &= \text{Prob}\,[\Delta C_a \geq \Delta C_e > 0] \cdot |e^{-(C_e/T)} \cdot (e^{-(\Delta E/T)} - 1)| \\ P_{pes} &= \text{Prob}\,[\Delta C_e \geq \Delta C_a > 0] \cdot |e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1)|. \end{aligned}$$

The probabilities of optimistic cost error (P_{opt}) and pessimistic cost error (P_{pes}) are in [0, 1] [4].

Theorem 3-3: Since a cost error occurs only in a positive cost change, the total probability of cost error, P_T, is given by

$$P_T \leq \text{Prob}\,[\Delta C_e > 0] \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1).$$

Proof:

$$\begin{aligned} P_T &= P_{pes} + P_{opt} \\ &\leq (\text{Prob}\,[\Delta C_e \geq \Delta C_a > 0] + \text{Prob}\,[\Delta C_a \geq \Delta C_e > 0]) \\ &\quad \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1). \\ &= \text{Prob}\,[\Delta C_e > 0] \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1). \end{aligned}$$

Now the cost error can be determined using the probability of cost error (P_T).

Theorem 3-4: The amount of cost error in the hill climbing move $(\Delta C_e > 0)$ is

$$E = \Delta C_e \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1).$$

Proof: Since $\text{Prob}\,[\Delta C_e > 0] = 1$, the probability of cost error in the hill climbing move is given by

$$\begin{aligned} E &= \Delta C_e \cdot P_T \\ &= \Delta C_e \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1), \quad \text{given that } \Delta C_e > 0 \\ &\qquad \text{from Theorem 3-3.} \end{aligned}$$ □

[2] suggested that some problems or algorithms [6] are more resistant to the cost error than others [11], [10]. This robustness to the cost error can be explained by Theorem 3-5.

Theorem 3-5: The total amount of the cost error (E) depends on the portion of the cost error $(\Delta E$ or $i \cdot \alpha \cdot |\langle E \rangle|)$ in the estimated cost (ΔC_e).

Proof: From Theorem 3-4, when the cost error, ΔE, has only a small portion of the estimated cost error, ΔC_e, i.e. $\Delta E \ll \Delta C_e$, the total probability of the cost error goes to 0.

$$\begin{aligned} E &\propto e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1) \\ &= e^{-(\Delta C_e - \Delta E/T)} - e^{-(\Delta C_e/T)}. \end{aligned}$$

Since $\Delta C_e > \Delta E > 0$ in the hill climbing move, the total amount of cost error (E) is alway positive. Therefore, robustness to the cost error depends on the portion of the cost error (ΔE) in the estimated cost (ΔC_e). □

It was shown that the traditional cost error measurement scheme (Definition 2-1) has three shortcomings. These shortcomings are now corrected as follows. First, the new cost error measurement method includes the cost error of the rejected moves because the probability of pessimistic error is added to the probability of the total cost error (Corollary 3-1 and Theorem 3-3). Second, this method does not include the cost error of the negative cost change moves because the move decision is always correct regardless of the cost error used (Lemma 3-2). Finally, there is no compensated cost error between the pessimistic and optimistic cost errors because this method adds to the probabilities of the pessimistic and optimistic cost error (Theorem 3-3).

C. Maximum Bound of Tolerable Error

In this section, the optimal stream length is derived for a fixed amount of cost error. Since a cost error is tolerated by hill climbing moves, a maximum bound on the cost error can be defined using a maximum bound on the hill climbing move.

Theorem 3-6 [12]: Let $d(s)$ be the maximum amount (or depth) of cost which can be hill-climbed at a given temperature T and stream length s. Then

$$e^{-(d(s)/T)} \geq \frac{1}{s} \Rightarrow d(s) \leq T \ln s.$$

This means that there is a possibility to choose $d(s)$ hill climbing moves in s moves [12]. The maximum hill climbing depth is a function of temperature and log of the stream length.

The error-present simulated annealing has a smaller hill climbing power than sequential simulated annealing, so the error-present algorithm is likely to be kept in a local minimum due to cost error (Fig. 2). Since the decreased hill climbing power is due to the cost error, the following theorem is derived.

Theorem 3-7: The hill climbing depth of the error-present algorithm (d_e) is less than that of the sequential algorithm (d_a) by at most the amount of error (E).

$$d_a \leq d_e + E$$

where d_a is the hill climbing depth of sequential simulated annealing for one hill climbing move, d_e is the hill climbing depth of the error-present algorithm for one hill climbing move, and E is the hill climbing error derived from Theorem 3-4.

Proof: A loss of hill climbing power is introduced only by pessimistic errors $(\Delta C_e > \Delta C_a > 0)$. Hill climbing power, probabilistically, is $d_a = \Delta C_a \cdot e^{-(\Delta C_a/T)}$ in the sequential annealing process and $d_e = \Delta C_e \cdot e^{-(\Delta C_e/T)}$ in the error-present annealing process. Using E (from Theorem 3-4) and pessimistic condition $(\Delta E = \Delta C_e - \Delta C_a$ where $\Delta C_e > \Delta C_a > 0)$, we have

$$\begin{aligned} d_a - (d_e + E) &= \Delta C_a \cdot e^{-(\Delta C_a/T)} - (\Delta C_e \cdot e^{-(\Delta C_e/T)} \\ &\quad + \Delta C_e \cdot e^{-(\Delta C_e/T)} \cdot (e^{\Delta E/T} - 1)) \\ &= e^{-(\Delta C_a/T)} \cdot (\Delta C_a - \Delta C_e) \leq 0 \\ &\qquad \text{from Lemma 3-1 and} \\ &\qquad \text{pessimistic condition.} \end{aligned}$$

So, $d_a \leq d_e + E$. □

Next, an extra stream length (u) is required for the decreased amount of hill climbing depth, $E(s)$, throughout the stream length s.

Corollary 3-2: The extra moves (u) to tolerate the cost error $E(s)$ is given by

$$e^{-[E(s)/T]} \geq \frac{1}{u}, \quad \text{so } u \geq e^{E(s)/T}. \qquad \text{from Theorem 3-6}$$

For a given temperature T, at least u moves have a hill climbing power $E(s)$; and with stream length s, there is a hill climbing power $d_e(s)$ in an error-present algorithm.

Corollary 3-3:

$$e^{-(d_e(s)/T)} \geq \frac{1}{s}. \qquad \text{from Theorem 3-6}$$

Now the stream length s_a can be calculated for the error-tolerable algorithm having a regular hill climbing depth $d_a(s)$. That is, in order to increase the hill climbing depth to match that of sequential simulated annealing, the stream length s_a is needed.

Theorem 3-8: When the total amount of cost error $(E(s))$ occurs during stream length s, $s_a = s \cdot u$ stream length is needed to tolerate the cost error.

Proof:

$$\begin{aligned} e^{-(d_a/T)} &\geq e^{-(d_e+E)/T} && \text{from Theorem 3-7} \\ e^{-(d_a(s)/T)} &\geq e^{-(d_e(s)+E(s))/T} = e^{-(d_e(s)/T)} \\ &\quad \cdot e^{-(E(s)/T)} && \text{from ergodicity} \\ &\geq \frac{1}{s} \cdot \frac{1}{u} && \text{from Corollary 3-2} \\ & && \text{and Corollary 3-3} \end{aligned}$$

So $s_a = s \cdot u$ stream length is needed for the error present algorithm to have the same hill climbing depth as the sequential annealing process in the stream length s. □

```
PROCEDURE ERROR-TOLERANT SIMULATED ANNEALING
begin
  INITIALIZE;
  k:= 0;
  repeat
    calculate $|<E>|$ in $T_k$       /* Average cost error in one accepted move */
    $E(s) = 0$       /* $E(s)$ is the total amount of cost error in a given stream length */
    repeat
      PERTURB(config. $i \rightarrow$ config. $j, \Delta C_{ij}$);
      if $\Delta c_{ij} \leq 0$ then accept
      else
        $E(s) = E(s) + \Delta c_{ij} \cdot e^{-\frac{\Delta c_{ij}}{T_k}} \cdot \left( e^{\frac{i \cdot \alpha \cdot |<E>|}{T_k}} - 1 \right)$
                /* $i$ is the $i$th iteration in the stream length */
                /* $\alpha$ is the acceptance ratio */
        if $\exp(-\Delta c_{ij}/T_k) >$ random [0,1) then accept;
      if accept then
        UPDATE(configuration j);
    until equilibrium is approached sufficiently closely;

    $T_{k+1} := f(T_k)$;
    k:= k+1;
    if ( $E(s) \leq T_k \cdot \log u$ )       /* $u = 1.1$ */
      increase stream length
    else
      decrease stream length
  until stop criterion == true (system is 'frozen');

end.
```

Fig. 4. The adaptive stream length simulated annealing algorithm.

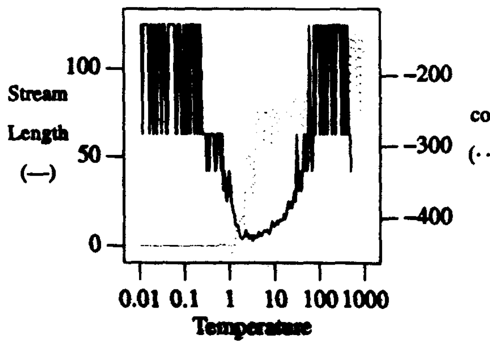

Fig. 5. Stream length versus annealing curve.

TABLE II
FINAL COST OF ADAPTIVE AND STATIC METHODS

	Mean	Std. Dev.	Worst	Best
Adp	-424.5	8.29	-414.4	-436.2
Static-10	-424.3	11.70	-402.9	-436.2

The next task is how to define the extra stream length factor u, considering the time for a global update. From Corollary 3-2,

$$E(s) \leq T \cdot \ln u, \quad \text{since}\, e^{-(E(s)/T)} \geq \frac{1}{u}. \tag{3-1}$$

In order to decrease the extra stream length factor u to improve solution quality, the maximum tolerable cost error $E(s)$ must be decreased as well. So, the extra stream length factor u and the maximum tolerable cost error $E(s)$ are proportional. That is, as the stream length increases, the cost error increases, so the extra stream length factor (u) must increase to keep the convergence.

For example, if a 10% increase of Markov chain length is allowed, i.e. $u = 1.1$, then the maximum error bound $E(s)$ can be calculated using (3-1). If the measured amount of cost error in a given stream length s is greater than the maximum bound cost error $E(s)$, the stream length will be decreased. If the measured amount of cost error in a given stream length s is less than the maximum bound cost error $E(s)$, the stream length will be increased. When the stream length is changed, the Markov chain length M is kept fixed, i.e. Markov chain length $M = \text{stream length}(s) \times$ # of global updates in a given temperature. The pseudocode for the adaptive stream length simulated annealing is in Fig. 4.

With respect to the methods of [1], [3] surveyed in Section II, the developed method adds the stream length by an extra stream length factor, u, to maintain the convergency of the sequential annealing algorithm. Even though [1] derives the maximum error bound empirically based on the acceptance distribution, and the equation of [1] (2-2) is same of that of the proposed method (3-1), this method derives it from the hill climbing power and proves the convergency by introducing the extra stream length factor, u. So this method is universally applicable and does not need previous runs to set a maximum error bound. Furthermore, this paper address a new cost error measurement scheme which corrects the problems of traditional error measurement schemes.

IV. EXPERIMENTAL RESULTS

The adaptive stream length algorithm (Fig. 4) was implemented on a 16 processor Intel iPSC/2 multicomputer. The target problem was the composite stock cutting problem (Section I), which was decomposed spatially along space of the stock sheet.

The parallel space-decomposition simulated annealing algorithm was implemented in 4 nodes. A total of 16 irregular patterns were used. The Markov chain length was 500. Comparing the stream length at each temperature with the annealing curve (Fig. 5), the stream length reduces to 2 in the critical region. The critical region is the middle-temperature region in the annealing curve. In this region the system is cooling down rapidly, so very slow cooling is required. However, the stream length increases to 125 far from the critical region. The stream length varies dynamically according to the annealing curve. This means the cost error has little affect on the annealing process away from the critical region, but affects it greatly in the critical region. This corresponds to the fact that the annealing process can proceed rapidly away from the critical region, but much more slowly in the critical region to keep still convergency property.

In Table II, Adp denotes the adaptive method, and Static-10 means that the stream length was fixed at 10. Since the average final cost starts to increase above the stream length 10, the stream length of 10 was selected for the static method. While the average final costs between the two methods was similar, the standard deviation of the adaptive method was smaller than that of the static method, as expected. The average stream length of the adaptive method is larger than that of the static method. Since the number of global updates was inversely proportional to the stream length, the average number of global updates was reduced 6.3 times in the adaptive method, compared to the static method. From the above data (Fig. 5, and Table II), the adaptive method adapts the stream length dynamically, with comparable final results.

As a second experiment, 16 different sets of patterns were implemented to observe the results of the adaptive method. The number of

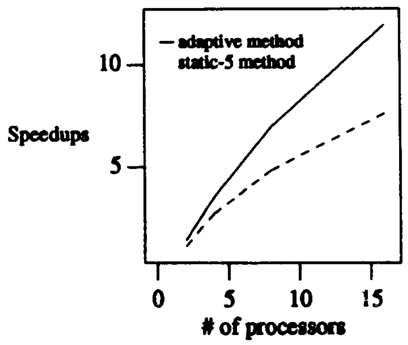

Fig. 6. Speedups of adaptive and static.

TABLE III
FINAL COST OF ADAPTIVE AND STATIC METHODS

node size		Mean	Std. Dev.
Seq.		583848	91797.9
2	Adp	587628	83453.7
	Static	588608	83138.0
4	Adp	578001	77181.2
	Static	575734	79882.2
8	Adp	574773	76009.6
	Static	575499	75534.0
16	Adp	589618	75242.4
	Static	591113	76073.4

patterns varied from 128 to 160. A cooling schedule was set such that the initial temperature was about 200 000, the temperature decrement ratio was 0.98 to 0.99, and the Markov chain length was 5000 to 20 000. Fig. 6 plots the adaptive method (adaptive) and the static method, where the stream length was fixed at 5 (static-5) in terms of their speedups with respect to the sequential annealing process. Since the adaptive method reduced the global update frequency, the adaptive method achieves a better speedup than the static method for a large number of processors.

Table III compares the final cost of the adaptive method with that of the static method. The stream length of the static method, for each node size, was chosen optimally in the range from 5 to 100. The mean final cost of the adaptive method was smaller than that of the static method over the entire processor range. If we let the cost deviation of the parallel implementation be defined as

$$\text{Cost Deviation} = \frac{\text{Cost of Parallel} - \text{Cost of Sequential}}{\text{Cost of Sequential}}$$

then the cost deviation of the parallel implementation using 16 nodes was less than 1% for the adaptive method and 1.2% for the static method. The standard deviation of the adaptive method was smaller than that of the static method for all node ranges. This corresponds to the previous experiments (Table II).

From the experimental results, the adaptive method is well suited for relaxing the frequency of the global updates, i.e., for increasing the stream length while maintaining the quality of the final results comparatively. Aside from the improved speedups, the adaptive method has an advantage over the static method since, in the latter, we cannot determine the optimal stream length *a priori.*

V. CONCLUSION

In this paper we developed a cost error measurement scheme and an adaptive error control method in terms of stream length based on a cost error analysis of hill climbing power. An adaptive error control algorithm was developed that varies the stream length as a function of the annealing schedule. Experimental results show that this method reduces the frequency of global state updates, thus, improving the parallel speedup, while still reaching the optimal configuration of the system. Additionally, the adaptive error control scheme chooses the optimal stream length dynamically rather than through the extensive experimentation required by a static stream length method.

REFERENCES

[1] P. Banerjee, M. H. Jones, and J. S. Sargent, "Parallel simulated annealing algorithm for cell placement on hypercube multiprocessors," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 1, Jan. 1990.
[2] M. Durand, "Parallel simulated annealing: Accuracy versus speed in placement," *IEEE Design and Test*, pp. 8–34, June 1989.
[3] L. Grover, "A new simulated annealing algorithm for standard cell placement," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 1986.
[4] C.-E. Hong, "Relaxing synchronization in distributed simulated annealing," Ph.D. dissertation, Univ. Missouri-Rolla, 1992.
[5] R. Jayaraman and F. Darma, "Error tolerance in parallel simulated annealing technique," in *Proc. Int. Conf. Comput. Design*, 1988.
[6] R. Jayaraman and R. Rutenbar, "Floorplanning by annealing on a hypercube multiprocessor," in *Proc. Int. Conf. Comput.-Aided Design*, 1987.
[7] S. Kirkpatrick, C. D. Gellatt, Jr., and M. P. Vecci, "Optimization by simulated annealing," *Science*, vol. 220, pp. 975–986, 1983.
[8] H. Lutfiyya, B. McMillin, P. Poshyanonda, and C. Dagli, "Composite stock cutting through simulated annealing," *Mathematics Computer Modeling*. New York: Pergamon, vol. 16, 1992, pp. 57–74.
[9] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, pp. 1087–1092, 1953.
[10] J. Rose, D. Blythe, W. Snelgrove, and Z. Vranesic, "Fast, high quality VLSI placement on an MIMD multiprocessor," *Proc. Int. Conf. Comput.-Aided Design*, 1986, pp. 42–45.
[11] J. Rose, W. Klebsch, and J. Wolf, "Temperature measurement and equilibrium dynamics of simulated annealing placement," *IEEE Trans. Comput.-Aided Design*, vol. 9, Mar. 1990.
[12] S. White, "Concepts of scales in simulated annealing," in *Proc. IEEE Int. Conf. Comput. Design*, Port Chester, NY, Nov. 1984, pp. 646–651.