Best-So-Far vs. Where-You-Are: New Perspectives on Simulated Annealing for CAD*

Kenneth D. Boese, Andrew B. Kahng and Chung-Wen Albert Tsao

UCLA Computer Science Dept., Los Angeles, CA 90024-1596 USA

Abstract

The **simulated annealing** *(SA) algorithm [14] [5] has been applied to every difficult optimization problem in VLSI CAD. Existing SA implementations use monotone decreasing, or* **cooling**, *temperature schedules motivated by the algorithm's proof of optimality as well as by an analogy with statistical thermodynamics. This paper gives strong evidence that challenges the correctness of using such schedules. Specifically, the theoretical framework under which monotone cooling schedules is proved optimal fails to capture the* **practical** *application of simulated annealing; in practice, the algorithm runs for a finite rather than infinite amount of time; and the algorithm returns the best solution visited during the entire run ("best-so-far") rather than the last solution visited ("where-you-are"). For small instances of classic VLSI CAD problems, we determine annealing schedules that are* **optimal** *in terms of the expected quality of the best-so-far solution. These optimal schedules do not decrease monotonically, but are in fact either* **periodic** *or* **warming**. *(When the goal is to optimize the cost of the where-you-are solution, we confirm the traditional wisdom of cooling.) Our results open up many new research directions, particularly how to choose annealing temperatures dynamically to optimize the quality of the finite time, best-so-far solution.*

Keywords: *Simulated annealing, global optimization, module placement, graph bisection.*

1 Overview

The *simulated annealing* (SA) algorithm has been applied to every difficult optimization problem in VLSI CAD. Its most concentrated application has been in the area of cell placement; studies of SA for placement include those of Kirkpatrick et al. [14], Sechen [23], Rose [22], and Lam and Delosme [16]. In addition, SA has been applied to scheduling and allocation, logic minimization, PLA folding, partitioning, floorplanning, routing, compaction, and transistor sizing, to name only several other areas in the literature [1] [15] [27]. Indeed, SA is now a dominant methodology across the spectrum of synthesis and layout tools.

All existing implementations of SA use monotone cooling temperature schedules, i.e., at each step, the controlling "temperature" parameter either decreases or remains the same. In general, the algorithm ends with a sequence of steps with zero temperature. Over a hundred papers in the VLSI CAD literature alone have studied variant cooling schedules in the context of particular CAD optimizations. Thus, it is highly significant that our work gives strong evidence to challenge the universal use of cooling schedules. We believe that two aspects of the theoretical analysis of SA have misled practitioners into concentrating on the cooling paradigm: 1) the analysis generally considers schedules of infinite rather than finite length, and 2) the quality of a schedule is almost always based on the last solution visited ("where-you-are"), rather than the best solution visited during the entire annealing run ("best-so-far"). For the first time in the CAD literature, we have solved for *optimal* annealing schedules for small instances of three separate CAD problems and found that optimal schedules are *not* monotone cooling. Furthermore, our experiments indicate that using best-so-far optimal schedules rather than where-you-are optimal schedules will produce a reduction in time of between 30% and 45% to achieve the same expected solution quality.

2 Preliminaries

SA can be applied to almost any discrete global optimization problem; such problems are generally of the form:

> Given a finite solution set S and cost function $f : S \rightarrow \Re$, find $s \in S$ such that $f(s) \leq f(s') \;\; \forall \; s' \in S$.

Typically, $|S|$ is very large compared to the number of solutions that can be reasonably examined in practice. Moreover, many important formulations are intractable [7], so that general-purpose heuristics are of interest.

One of the most successful global optimization heuristics is *simulated annealing* (SA), which was proposed independently by Kirkpatrick et al. [14] and Cerny [5] and is motivated by analogies between the solution space of an optimization instance and microstates of a statistical thermodynamical ensemble. From the solution $s_i \in S$ at the i^{th} time step, the SA algorithm (Figure 1) generates a "neighbor" solu-

*Partial support for this work was provided by a GTE Graduate Fellowship and by NSF MIP-9110696 and NSF Young Investigator Award MIP-9257982.

tion s' and decides whether to adopt it as s_{i+1}, based on the cost difference $f(s') - f(s_i)$ and the value of a *temperature* parameter T_{i+1}. For each s_i, the set of possible neighbors s' is called its *neighborhood* $N(s_i)$; together the neighborhoods $N(s)$ for all $s \in S$ induce a topology over S called its *neighborhood structure*. Over the M steps for which the SA algorithm is executed, a *temperature schedule* $T_1, T_2, \ldots, T_M$ guides the optimization process. Typical SA practice uses a large initial temperature and a final temperature of zero, with T_i monotonically decreasing according to a fixed schedule or in order to maintain some measure of "thermodynamic equilibration".

SA Algorithm Template

0. $s_0 \leftarrow$ random solution in S
1. For $i = 0$ to $M - 1$
2. Choose $s' \leftarrow$ a random element from $N(s_i)$
3. **if** $f(s') < f(s_i)$
4. $s_{i+1} \leftarrow s'$
5. **else**
6. $s_{i+1} \leftarrow s'$ with probability $e^{-[f(s')-f(s_i)]/T_{i+1}}$
7. otherwise $s_{i+1} \leftarrow s_i$
8. Return s_M

8a. Return s_i, $0 \le i \le M$, such that $f(s_i)$ is minimum.

Figure 1: The simulated annealing algorithm for a given bound of M time steps.

The SA algorithm enjoys certain theoretical attractions [15]. Using Markov chain arguments and basic properties of Gibbs-Boltzmann statistics, one can show that for any finite S, SA will converge to a globally optimal solution given infinitely large M and a temperature schedule that converges to zero sufficiently slowly. In other words,

$$Pr(s_M \in R) \rightarrow 1 \quad as \quad M \rightarrow \infty \tag{1}$$

where $R \subset S$ is the set of all globally optimal solutions, so that SA is "optimal" in the limit of infinite time. Several groups have proved that specific temperature schedules guarantee convergence of SA to a global optimum, e.g., Hajek [9] showed the optimality of "logarithmic cooling" with $T_i = \frac{a}{\log(i+1)}$ when a is sufficiently large (see also [8] [20]).

3 Best-So-Far vs. Where-You-Are

Theoretical analysis of annealing has always been performed with respect to a "where-you-are" (WYA) implementation, where the algorithm returns whichever solution is last visited (line 8 of Figure 1). According to the theoretical analysis, it is this single solution s_M that in the limit $M \rightarrow \infty$ has probability 1 of being optimal. On the other hand, a practical implementation will never ignore all of the solutions $s_0, s_1, \ldots, s_{M-1}$: certainly, one can maintain the best solution seen so far, and return it if it is better than s_M (line 8a of Figure 1). We call this more realistic variant "best-so-far" (BSF) annealing. Traditional convergence proofs for WYA annealing also apply to BSF annealing, since optimality according to Equation (1) trivially implies optimality of the BSF variant. However, the results of Section 5 below show that BSF-optimal temperature schedules differ markedly from the traditional "cooling" that is suggested by both the physical annealing analogy and the WYA analysis.

The extensive literature on simulated annealing contains little mention of either non-monotone cooling schedules or best-so-far analysis. Lasserre et al. [17] use a BSF implementation in comparing simulated annealing to other iterative optimization heuristics, but do not explore the implications of using BSF as opposed to WYA. A more direct reference to BSF is contained in the 1988 work of Hajek [9], which establishes necessary and sufficient conditions for the infinite-time WYA optimality of monotone decreasing temperature schedules. Hajek briefly suggests ([9], p. 315) a similar analysis for BSF: "It would be interesting to know the behavior of $\min_{n \le k} V(X_n)$ rather than the behavior of $V(X_k)$."

Non-cooling schedules have been investigated by Hajek and Sasaki in [10], which shows the existence of a special class of optimization problems for which monotone cooling schedules are suboptimal. An ancillary result of [10] is that for neighborhood structures where the costs of any two neighboring solutions differ either by zero or a constant, there exists an optimal annealing schedule where all T_i are either 0 or $+\infty$ (cf. our results showing optimal "periodic" schedules in Section 5.2 below). While the BSF criterion is not mentioned in [10], the authors do suggest a similar measure of schedule quality, specifically, the expected number of time steps required to first encounter a solution with cost less than or equal to some prescribed constant. Finally, our study of optimal *finite-time* schedules follows in the direction established by Strenski and Kirkpatrick [25].

4 Computing Optimal Schedules

Strenski and Kirkpatrick [25] use numerical methods to estimate optimal finite-time schedules according to an exact equation for computing expected WYA cost. We now state their technique and then extend it to compute optimal schedules according to the BSF criterion.

For any given finite schedule length M, an *optimal temperature schedule* is one for which simulated annealing has the lowest expected solution cost after M steps, assuming that all initial states s_0 are equally likely. We use $P(i)$ to denote the $1 \times |S|$ row vector whose jth element $[P(i)]_j$ gives the probability that solution s^j is the current solution at step i.[1] Because each s^j has equal probability of being the initial state, we have that $P(0) = [\frac{1}{|S|}, \frac{1}{|S|}, \ldots, \frac{1}{|S|}]$. We let $A(T_i)$ denote the $|S| \times |S|$ transition matrix induced by temperature T_i, i.e., $[A(T_i)]_{jk}$ equals the probability of moving from solution s^j to solution s^k in one step at temperature T_i. Thus, we can calculate each $P(i)$ re-

[1] Here, we use superscripts to denote indices of particular solutions $s^j \in S$. We continue to use subscripts of solutions to denote time steps, so that s_i is the current solution at step i.

cursively as $P(i) = P(i-1)A(T_i)$. Let C be the $|S| \times 1$ column vector of costs for solutions in S. Then the expected WYA cost $E[f(s_M)]$ is equal to

$$E[f(s_M)] = P(0) \cdot A(T_1) \cdot A(T_2) \cdot \cdots \cdot A(T_M) \cdot C. \quad (2)$$

To compute the expected BSF cost of a temperature schedule, we first sort the solutions $s^j \in S$ in order of increasing cost $f(s^j)$, so that s^1 is the optimal solution and $s^{|S|}$ is the solution with highest cost. For each solution s^j and temperature T, we define a new transition matrix $B^j(T)$ such that

$$[B^j(T)]_{k\ell} = \begin{cases} 1 & \text{if } k = \ell \text{ and } k \leq j \\ 0 & \text{if } k \neq \ell \text{ and } k \leq j \\ [A(T)]_{k\ell} & \text{if } k > j \end{cases}$$

In other words, transitions in $B^j(T)$ are the same as in $A(T)$, except that a self-move is the only possible transition from a solution of cost less than or equal to $f(s^j)$. In this way, each solution with cost less than or equal to $f(s^j)$ becomes a "sink" in transition matrix $B^j(T)$. We define $P^j(i)$, a $1 \times |S|$ probability vector, such that $P^j(0) = P(0)$ and $P^j(i) = P^j(i-1)B^j(T_i)$, $\forall i > 0$. For instance, $P^1(i)$ contains the probability distribution of solutions at step i if the global optimum s^1 has been converted to a sink in all steps up to i.

We use $d^j(i)$ to denote the probability of ever reaching a solution with cost $f(s^j)$ or less within the first i steps. The value of $d^j(i)$ is given by the summation

$$d^j(i) = \sum_{\ell=0}^{j} [P^j(i)]_\ell$$

Note that $d^1(M)$ is equal to the first element in $P^1(M)$, and so equals the probability of ever reaching the global optimum. For $j > 0$, the probability that s^j is the BSF solution after M steps is simply $d^j(M) - d^{j-1}(M)$, i.e., the probability of ever reaching a solution of cost $f(s^j)$ or lower, minus the probability of ever reaching a solution of cost $f(s^{j-1})$ or lower.[2] Thus, the expected cost of the BSF solution is

$$d^1(M)f(s^1) + \sum_{j=2}^{|S|} [d^j(M) - d^{j-1}(M)] f(s^j)$$

Because the calculation of each $d^j(M)$ requires the same number of matrix multiplications as $E[f(s_M)]$, the calculation of BSF cost is $\Theta(|S|)$ times more expensive than the calculation of WYA cost. Note that the BSF formula is linear in each $A(T_i)$ and $B^j(T_i)$, so that exact derivatives may be computed for use with numerical optimization techniques.

The experiments of Section 5 determine optimal annealing schedules with respect to a discrete set of possible values of T_i: in addition to $T_i = 0$, we also choose from among 100 evenly-spaced temperature values > 0, such that the overall range $[0, +\infty]$ is effectively represented.[3] For larger values of M, it is impossible to exhaustively enumerate all possible temperature schedules, and we therefore use an iterative method to generate locally optimal schedules. At each iteration, we test all possible perturbations of a single temperature T_i ($i = 1, \ldots, M$) to an adjacent temperature value above or below T_i, then deterministically adopt the single perturbation which yields the greatest improvement in expected solution cost. The search terminates when a locally optimal schedule is found.[4] For each estimation, we begin from several different initial schedules and report the best locally optimal schedule; we have observed very few distinct local optima, with almost all being qualitatively very similar.

5 Experimental Results: BSF-Optimal Schedules

In this section, we study small instances of classic combinatorial optimization problems – graph placement and graph bisection – which are prominent in the CAD literature. For each of these problem instances, we compute locally BSF- and WYA-optimal temperature schedules as described above. Recall that the methods of Section 4 require iterative optimization of chain products involving M matrices of size $|S| \times |S|$. Thus, our experiments have been performed for the largest instances and time bounds that are consistent with our available hardware and the objective of determining *optimal* schedules.

5.1 Graph Placement

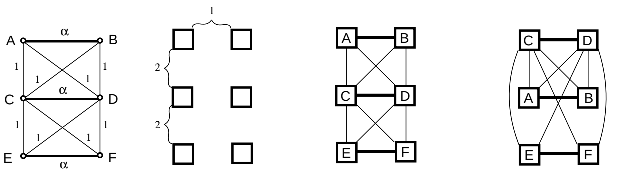
Given an edge-weighted graph G with n nodes, and given a set L of n locations with all $\frac{n(n-1)}{2}$ inter-location distances, the graph placement problem is to find a one-to-one mapping from the nodes of G onto L so that the weighted sum of edge lengths of G is minimized. This formulation captures minimum-wirelength module placement as in the original SA work of [14] and the well-known Timberwolf package [23].

We consider the six-node graph placement instance shown in Figure 2. Because of symmetries, there are only 17 distinct classes of solutions for this instance. We choose the neighborhood operator to be a swap of node pairs that are adjacent vertically, horizontally, or diagonally in the current solution. Figure 2 shows the global optimum and the unique local optimum configurations when the edge weight $\alpha = 5$.

[2] In the case where $f(s^j) = f(s^{j-1})$, we arbitrarily force the algorithm to return s^{j-1} as the BSF solution whenever a run visits both s^j and s^{j-1} during its execution.

[3] Our experiments have used $T_i \in \{1, 2, \ldots, 100\}$, as well a range of values chosen such that the transition probabilities for an "average" disimproving move are $0.01, 0.02, 0.03, \ldots, 0.99, \approx 1.00$. We found that results are qualitatively the same with either range of T_i values.

[4] Strenski and Kirkpatrick [25] also find locally optimal, rather than globally optimal, (WYA) schedules. Their method uses the partial derivatives of the expected WYA cost $E[f(s_M)]$ with respect to each T_i to afford a gradient-descent method. Note that our WYA-optimal schedules in Section 5.2 are essen-



(a) graph G (b) locations (c) global optimum (d) local optimum

Figure 2: Six-node graph placement instance with edge weights as shown in (a); thick edges have weight $\alpha = 5$. Available locations are in the Manhattan plane, as shown in (b). The global optimum and the local optimum solutions are respectively given in (c) and (d).

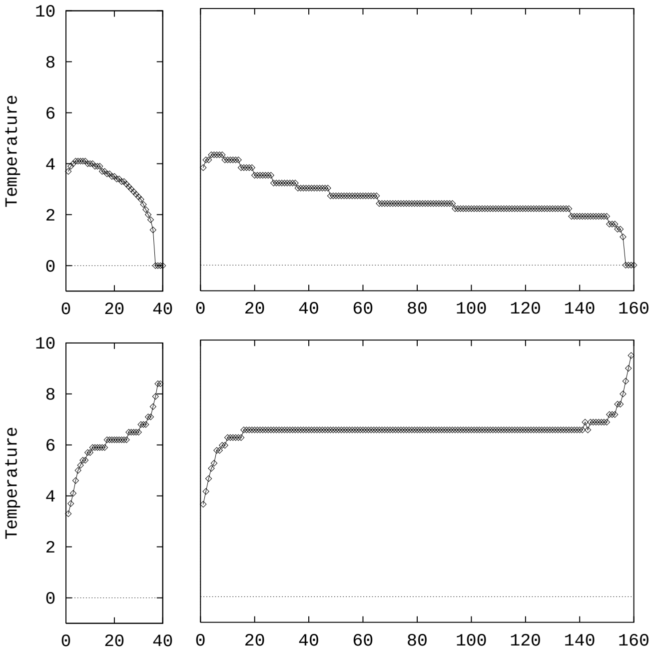


Figure 3: Locally WYA-optimal (top) and BSF-optimal (bottom) temperature schedules of lengths 40 and 160 steps for the 6-node graph placement instance.

For this problem instance, the contrast between WYA- and BSF-optimal temperature schedules is quite dramatic (Figure 3). The WYA-optimal schedules are monotone decreasing, as would be expected from the traditional intuition. In contrast, the BSF-optimal schedules are *monotone increasing*, and are clearly superior to the optimal WYA schedules when judged by the practical BSF criterion (see Table 1 and Figure 4). For example, the BSF quality of the 80-step WYA-optimal schedule can be achieved by a BSF-optimal schedule that is approximately 30% shorter (about 57 time steps). It is also interesting to note the poor WYA quality of BSF-optimal schedules, particularly for larger values of M; this might indicate the irrelevance, in practice, of optimizing $f(s_M)$.

tially identical to those obtained in [25] using gradient descent.

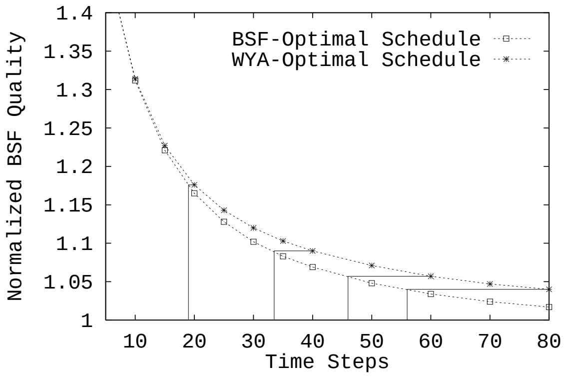


Figure 4: Expected BSF quality of solutions obtained by BSF-optimal and WYA-optimal schedules for the 6-node graph placement instance. Solution quality is normalized to the cost of the optimal solution.

	BSF-Optimal Schedule		WYA-Optimal Schedule	
Number of Steps	BSF Quality	WYA Quality	BSF Quality	WYA Quality
5	1.471	1.473	1.471	1.471
10	1.312	1.326	1.314	1.315
15	1.221	1.268	1.227	1.230
20	1.165	1.255	1.176	1.182
25	1.128	1.257	1.143	1.152
30	1.102	1.264	1.120	1.132
35	1.083	1.270	1.103	1.117
40	1.069	1.276	1.090	1.105
50	1.048	1.284	1.071	1.088
60	1.034	1.287	1.057	1.076
70	1.024	1.290	1.047	1.066
80	1.017	1.290	1.040	1.058

Table 1: Expected BSF and WYA solution quality produced by locally optimal schedules for the 6-node graph placement instance. Solution quality is normalized to the cost of the optimal solution.

5.2 Graph Bisection

We have also studied a small instance of the graph bisection problem. Given an edge-weighted graph $G = (V, E)$ with an even number of nodes, the graph bisection problem seeks a partition of V into disjoint U and W, with $|U| = |W|$, such that sum of the weights of edges $(u, w) \in E$ with $u \in U$, $w \in W$ is minimized. Graph bisection is basic to recursive netlist partitioning, floorplanning and area estimation. Use of annealing to solve the graph bisection problem is less common than use of such iterative greedy methods as the Kernighan-Lin algorithm [13] or its enhancement by Fiduccia and Mattheyses [6]. Nevertheless, SA has been well-studied in the context of graph bisection, notably by Johnson et al. [12]. The annealing algorithm has also been carefully compared against iterative methods in [3] and [26].

We have studied the same highly-structured instance treated by Strenski and Kirkpatrick in [25]. This instance consists of a complete graph of eight nodes, with edge weights calculated as shown in Figure 5(a). Each of the eight nodes is represented by a leaf in the height-3 binary tree shown in the figure; the edge between any two nodes has weight α^k, where k is the depth of the least common ancestor between the two nodes in the binary tree. Both our experiments and those of [25] use $\alpha = 3$. The globally optimum partition is $\{1,2,3,4\}\{5,6,7,8\}$, which corresponds to solution A in Figure 5(b) with cost $f(A) = 16$. Because of symmetries in the edge weight construction, there are only five classes of equivalent solutions, whose multiplicities and relative transition probabilities are given in Figure 5(b).

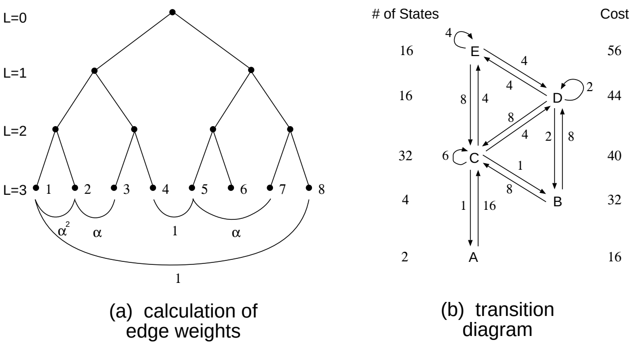


Figure 5: Edge weight calculation and state transition diagram for the complete graph used as a bisection instance by Strenski and Kirkpatrick (as in [25], we use $\alpha = 3$).

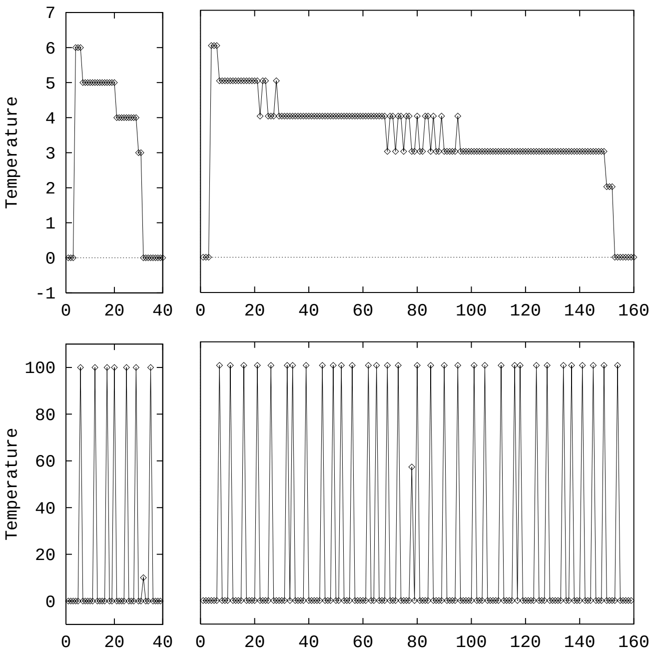


Figure 6: Locally WYA-optimal (top) and BSF-optimal (bottom) temperature schedules of length 40 and 160 steps for the 8-node graph bisection instance of [25].

We computed locally WYA- and BSF-optimal temperature schedules for this bisection instance, again following the experimental protocol above, and using the discrete range of temperatures $\{0, 1, 2, \ldots, 100\}$. Locally optimal schedules of $M = 40$ and 160 time steps are shown in Figure 6. Our WYA-optimal schedules almost exactly match the results of [25], and again confirm the traditional cooling intuition except in the first four steps, which have temperature 0. On the other hand, our locally BSF-optimal schedules are completely different, containing temperatures of only 0 and 100 (essentially $+\infty$). These schedules are very nearly *periodic*, and are evocative of the "iterated descent" methodologies discussed by Baum [4] and Johnson [11] and the "steepest ascent-descent" method of Lasserre et al. [17]. Table 2 compares the expected BSF and WYA solution costs for locally optimal schedules of various lengths. Again, optimal schedules in terms of the traditional WYA objective are clearly suboptimal when measured by their *practical*, BSF utility. The BSF-optimal schedules are considerably more effective, using approximately 30% fewer steps to achieve the same expected BSF quality as the WYA-optimal schedules.

	BSF-Optimal Schedule		WYA-Optimal Schedule	
Number of Steps	BSF Quality	WYA Quality	BSF Quality	WYA Quality
5	2.038	2.038	2.038	2.038
10	1.780	1.981	1.797	1.797
15	1.603	1.978	1.661	1.675
20	1.475	1.920	1.548	1.586
25	1.379	2.036	1.462	1.514
30	1.306	1.967	1.396	1.453
35	1.248	2.039	1.343	1.402
40	1.202	1.969	1.298	1.359
50	1.135	1.970	1.231	1.289
60	1.090	2.032	1.181	1.236
70	1.063	2.048	1.143	1.195
80	1.041	2.044	1.112	1.164

Table 2: Expected BSF and WYA solution quality produced by locally optimal schedules for the 8-node graph bisection instance of Strenski and Kirkpatrick. Solution quality is normalized to the cost of the optimal solution.

Finally, we note that the 8-node graph bisection instance is defined in [25] in such a way that it generalizes to complete graphs with 2^k nodes (Strenski and Kirkpatrick analyze only the 8-node instance). We have also computed locally BSF-optimal and WYA-optimal temperature schedules for the 16-node instance with $\alpha = 3$, where the symmetries in the solution space allow us to reduce the number of solution classes to 28. For this instance, we found that WYA-optimal schedules are similar to those for the 8-node instance; however, BSF-optimal schedules are no longer periodic, but are instead nearly monotonically increasing, similar to the BSF-optimal schedules for the graph placement instance of Section 5.1.

A third set of experiments was performed on a small instance of the traveling salesman problem (TSP) with $n = 6$ cities [2]. The TSP is one of the most well-studied problems in the combinatorial optimization literature [18], and arises in mask lithography, plotting, PCB drilling, daisy-chain signal routing, and probe-testing. Our computations gave optimal schedules that are qualitatively similar to those obtained for the graph placement problem (except that the BSF-optimal schedules decrease slightly for the last fifteen to twenty steps). We also found a 45% time reduction for BSF-optimal schedules compared to WYA-optimal schedules with 80 steps.

6 Conclusions

In this paper, we have explored the implications of (finite-time) best-so-far analysis of the simulated annealing algorithm. This BSF analysis is more consistent with annealing practice than the traditional WYA analysis, because the best solution seen can easily be stored and returned at the the end of the algorithm execution. The study of *finite-time* annealing also reflects practical reality, since applications of the annealing algorithm are certainly limited to finite amounts of CPU time. The far-reaching consequences of BSF analysis are apparent even at first glance. For example, "infinite-time optimality" holds for a much wider range of schedules under the BSF criterion and yields an immediate challenge to the traditional wisdom of monotone cooling to zero: given infinite time, any schedule that is bounded *away from* zero will eventually visit a globally optimal solution, assuming that all solutions are reachable at non-zero temperatures.

To assess the practical effect of the BSF criterion on finite-time annealing strategies, we have numerically estimated BSF- and WYA-optimal schedules for small instances of classic VLSI CAD problem formulations. Although our analysis is restricted to small problems due to the computational complexity of finding *optimal* schedules, we use instances of *real* CAD problems, with realistic neighborhood structures and solution spaces. While WYA-optimal schedules confirm the traditional regime of monotone cooling, the BSF-optimal schedules are no longer monotone cooling, but are instead periodic or warming. Moreover, the BSF criterion can yield tangible improvements in practice: we obtain between 30% and 45% reductions in run length versus WYA-optimal schedules, while maintaining the same expected BSF solution quality.

Our intuition regarding BSF annealing is that "reachability" and "mobility" are critical to success, but are at odds with the WYA-optimal practice of reducing the temperature to zero in order to minimize $f(s_M)$. Put another way, all annealing schedules may be viewed as trading off between reachability among solutions (high T) and a bias to lower-cost solutions (low T). The WYA criterion forces the low-cost bias to dominate at the end of the run, while the BSF criterion may allow reachability to take precedence, particularly when searching over a large number of local minima is likely to return better results than cooling to a single local minimum. Thus, it is perhaps not surprising that BSF-optimal schedules may actually be "warming", even at the end of the annealing execution.

In conclusion, best-so-far analysis opens the door to completely new hill-climbing regimes, as well as new theoretical fronts (e.g., the Markov analysis of best-so-far annealing). A major area of our current research lies in the application of non-conventional temperature schedules to benchmarks for larger real-world problems. To this end, the experiments described here point to periodic "iterated descent" methods [4] [11] [17] and adaptive construction of "non-monotone" (warming) schedules as especially promising directions. We also believe that the tuning of BSF-optimal annealing strategies to the statistical parameters of optimization cost surfaces [24] will provide an important research direction.

References

[1] E. H. L. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines: a Stochastic Approach to Combinatorial Optimization and Neural Computing* (Wiley, Chichester, 1989).

[2] K. D. Boese, A. B. Kahng and C. W. Tsao, "Best-So-Far vs. Where-You-Are: New Directions in Simulated Annealing for CAD", *UCLA CSD TR-920050*, 1992.

[3] T. N. Bui, S. Chaudhuri, F. T. Leighton, and M. Sipser, "Graph Bisection Algorithms with Good Average Case Behavior", *Combinatorica* 7(2):171–191, 1987.

[4] E. B. Baum, Iterated descent: a better algorithm for local search in combinatorial optimization problems, *Proc. Neural Information Processing Systems* , D. Touretzky, ed., 1987.

[5] V. Cerny, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm, *J. Optimization Theory and Applications* 45(1) (1985), 41-51.

[6] C.M Fiduccia and R.M. Mattheyses, "A Linear Time Heuristic for Improving Network Partitions", *ACM/IEEE Design Automation Conf.*, 1982, pp. 175-181.

[7] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman, New York, 1979).

[8] S. Geman and D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Trans. on Pattern Analysis and Machine Intelligence* 6 (1984), 721-741.

[9] B. Hajek, Cooling schedules for optimal annealing, *Math. Oper. Res.* 13(2) (1985), 311-329.

[10] B. Hajek and G. Sasaki, Simulated annealing - to cool or not, *Systems and Control Letters* 12 (1989), 443-447.

[11] D. S. Johnson, Local optimization and the traveling salesman problem, *Proc. 17th Intl. Colloquium on Automata, Languages and Programming* (1990), 446-460.

[12] D. S. Johnson, C. R. Aragon, L. A. McGeoch and C. Schevon, Optimization by simulated annealing: an experimental evaluation; part i, graph partitioning, *Operations Research* 37 (1989), 865-892.

[13] B. W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs", *Bell System Tech. J.* Feb. 1970, pp. 291-307.

[14] S. Kirkpatrick, Jr. C. D. Gelatt, and M. Vecchi, Optimization by simulated annealing, *Science* 220(4598) (1983), 671-680.

[15] P. J. M. Laarhoven and E. H. L. Aarts, *Simulated Annealing : Theory and Applications* (D. Reidel, Boston, 1987).

[16] J. Lam and J. M. Delosme, "Performance of a New Annealing Schedule", *Proc. ACM/IEEE Design Automation Conf.*, 1988, pp. 306-311.

[17] J. B. Lasserre, P. P. Varaiya and J. Walrand, Simulated annealing, random search, multistart or SAD?, *Systems and Control Letters* 8 (1987), 297-301.

[18] E. L. Lawler, J. K. Lenstra, A. Rinnooy-Kan and D. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization* (Wiley, Chichester, 1985).

[19] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout* (Wiley-Teubner, Berlin, 1990).

[20] M. Lundy and A. Mees, Convergence of an annealing algorithm, *Math. Programming* 34 (1986), 111-124.

[21] F. Romeo and A. Sangiovanni-Vincentelli, Probabilistic hill climbing algorithms: properties and applications, *Proc. Chapel Hill Conf. on VLSI* (1985), 393-417.

[22] J. Rose and W. Klebsch, "Temperature Measurement and Equilibrium Dynamics of Simulated Annealing Placements", *IEEE Trans. on CAD* 9(2) (1990), pp. 253-259.

[23] C. Sechen and A. Sangiovanni-Vincentelli, "The timberwolf placement and routing package", *IEEE J. of Solid-State Circuits* 20(2) (1985), 510-522.

[24] G. B. Sorkin, "Efficient simulated annealing on fractal energy landscapes", *Algorithmica* 6 (1991), 367-418.

[25] P. Strenski and S. Kirkpatrick, Analysis of finite length annealing schedules, *Algorithmica* 6 (1991), 346-366.

[26] L. Tao and Y. C. Zhao, "Multi-Way Graph Partition by Stochastic Probe", *technical report* CSD-91-01, Concordia University, December 1991.

[27] D. F. Wong, H. W. Leong and C. L. Liu, *Simulated Annealing for VLSI Design* (Kluwer Academic, Boston, 1988).