Proceedings of the 1992 Winter Simulation Conference
ed. J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson

UNITING PROBABILISTIC METHODS FOR OPTIMIZATION

Bennett L. Fox

Mathematics Department
University of Colorado
Campus Box 170
P.O. Box 173364
Denver, Colorado 80217-3364, U.S.A.

ABSTRACT

We integrate the following probabilistic search methods: simulated annealing, [a probabilistic version of] tabu search, and [a radically-modified version of] genetic algorithms. In addition, we use a rejectionless version of simulated annealing, alone or in combination with the other two methods, while - in contrast to previous work - neither explicitly nor implicitly altering the cooling schedule. We note the implication of this for speed-up when searches are carried out independently on parallel processors.

1 INTRODUCTION

We modify the usual simulated-annealing setup by enlarging the state space to introduce and exploit both temporal and spatial memory. With temporal memory, we can reduce the chance of revisiting recently visited states - as is done in *tabu search*. This is especially important when the tentative-move probabilities depend on the objective-function values in the neighborhood of the current state. Without (tabu) penalties, a move out of a local minimum very likely then would be followed immediately by a move back to the same local minimum when the uphill move is to a state with no other downhill neighbors. To get [finite] temporal memory, we use a Cartesian product of the original state space. To get - in addition - spatial memory, the states become subsets of that Cartesian product rather than merely elements of it. The goal of spatial memory is to *diversify* the search by making it much less local and myopic. Spatial memory is also a prominent feature of genetic algorithms, but other than also using crossover operators (defined in Section 3) our scheme has little in common with genetic algorithms.

Besides introducing two-dimensional memory, our scheme speeds up simulated annealing by implicitly skipping direct self-loops in the search path [by considering the transition matrix conditioned on acceptance] while - in contrast to previous work [e.g., see Greene and Supowit (1986)] - neither explicitly nor implicitly altering the cooling schedule [by using a discrete-time counterpart QUICKER to the Lewis-Shedler thinning algorithm (1979) for generating variates from nonhomogeneous Poisson processes to determine the transition number on which the current self loop ends]. The Lewis-Shedler thinning algorithm is also described in Bratley, Fox, Schrage (1987, section 5.3.17) and in Devroye (1986, section VL.1.3). Parts of this paper are condensed (with some loss of precision) from Fox (1992a), where - among other things - connections to probabilistic tabu search are considered in detail. The loop-skipping scheme is detailed in Section 4; its correctness and speed are analyzed in Fox (1992b).

Given that the next state differs from the current state, it can be generated in O(1) time; regarding the work to compute the objective-function values at the neighbors as a sunk cost [to make intelligent tentative moves], the implicit constant is small. The work to compute the length of the current self loop is proportional to the number of geometric variates that QUICKER generates. Fox (1992b) shows that this number converges in quadratic mean to one as the transition number on which the self loop starts goes to infinity, provided that the cooling schedule cools at a rate slower than $\{(U-L)/\log(j+1)\}$ where j indexes transition numbers, U is an upper bound on the objective function over the feasible set, and L is a lower bound. We refer to such slower rates [i.e., with the numerator greater than U-L] as *subcanonical*. This convergence in quadratic mean implies that the work to execute simulated annealing is proportional to the number of accepted moves.

Under certain (weak) conditions, this implies that the expected remaining time to visit an optimal state when searching independently on m processors [or m disjoint

teams of processors] becomes linear in 1/m as the time spent unsuccessfully searching gets large. Fox (1992b) details an argument depending only on linear algebra. The intuition behind this result is Keilson's (1979, p. 92) metaprinciple, translated from a reliability setting to ours and extended to a chain that has a transition matrix that is only asymptotically stationary: the longer the search has not found an optimal state, the more exponentially distributed is the residual time to visit an optimal state. To complete the heuristic motivation: the expectation of the minimum of m iid exponential variates is 1/m times the expectation of the first. The qualification that the time spent unsuccessfully searching be large is unimportant. If the qualification does not hold, the problem is easy and so speed-up then matters little. Osborne and Gillett (1991) give empirical results for simulated annealing in which the running times (on one processor) to find a near-optimal solution are approximately exponentially distributed.

Our discussion is phrased in terms of minimizing a (possibly nonlinear) objective function over a subset of a Cartesian product of {0,1}, i.e., over a subset of the vertices of the unit hypercube. A problem where the objective function is (relatively) easily minimized given a setting of all its integer variables [such as mixed-integer programming] can be recast easily in that form. By contrast, genuine global optimization over continuous variables requires other ideas - though most of what we discuss here has counterparts there.

Sections 2 and 3 discuss temporal and spatial memory, respectively. Section 4 spells out QUICKER. Next, section 5 argues that cooling, as in simulated annealing, has practical importance. No familiarity with tabu search or genetic algorithms is assumed, but acquaintance with simulated annealing is needed - especially in Sections 4 and 5 - to appreciate our discussion. Section 6 gives our recommendations.

2 TEMPORAL MEMORY

The k-fold (say) Cartesian-product state space has a generic element which keeps track of the most recent k-step history of the search path, with the following exception: a rejected move leaves us at the same state, with no history update for that move. To inhibit short-run oscillation in the search path, a nonnegative (state-dependent) penalty is added to the original objective function. That penalty is zero when at a local minimum with respect to the original objective function (and state space) and all the elements of the current state are distinct (i.e., there are no duplicates in the k-step history). Assuming, with hardly any loss of generality, that each local minimum in the original state space corresponds to at least one state in the enlarged space with no duplicate elements, a minimum of the modified objective function corresponds to a minimum of the original objective function in an obvious way. Thus, we have set up an equivalence with the original problem.

3 SPATIAL MEMORY

Now each state is a collection (*population* in genetic-algorithm jargon) of elements of the form described in Section 2, with the following exceptions:

(A) An element can have the form (-,-,...,-,x) where the dashes correspond to dummy elements. Such elements arise when x is generated randomly (not necessarily uniformly) from the original state space, each state getting positive mass. {Such elements can be at an arbitrary distance from any element of the current or preceding state. Thus, the search is not local. To get good mutual spacing between such elements, while still guaranteeing irreducibility, with small positive probability g they are generated independently and with probability 1-g they can be generated antithetically; e.g., see Bratley, Fox, and Schrage (1987, problem 2.2.9).}

(B) An element can have the form (-,-,...,-,x,y,...,z) when its history has less than k steps.

(C) An element can have the form (-,-,...,-,[u,v],x,y,...,z) when x was generated by *crossover* from the pair [u,v]. {If (A:B) and (C:D) are conformally-partitioned bit strings, then crossover produces (A:D) and (C:B). Thus, the crossover above will produce an x' which becomes part of some other element. The penalty function can penalize a candidate move that undoes a previous crossover. Diversification is enhanced by crossover. A natural crossover location is a point [if it exists] where the problem nearly decouples.}

The other elements of the current state arise from search *intensification*: generally this means that their respective last components are close [say, in the Hamming metric] to the last components of elements of the preceding state on the search path. If each population consisted only of one such element, then we would be back to the scheme in Section 2. If, in addition, each element had only one component [i.e., history were not remembered], then we

would be back to a naive state space.

Each state has (say) r elements. The next state has s candidate elements, from which r are picked - without duplication - by some mechanism, such as the one detailed below. These s candidate elements include at least r of the form in (A) above. Only the latter are (conditionally) random, given the current state. On the other hand, the selection of the elements for the next state from the candidate elements is random - even given these candidates. Each of the s candidates has a positive probability of being selected. To inhibit concentration of the elements picked in a small part of the state space [i.e., to stimulate diversification], we

(i) First, according to the probabilistic scheme described shortly, pick the element that is to have smallest objective-function value (modified to incorporate penalties) among those chosen. {One sorts the objective-function values of the candidate elements. Next, by a routine combinatorial argument [see Appendix A], one calculates the number n(v) of possible next states whose smallest objective-function value among its elements is v. Such states are defined to have value v. We assume that the tentative-move probabilities are such that the (unconditional) probability that an element with value v is chosen is proportional to a product of the form $n(v) \cdot f(v) \cdot p(v)$, where n(v) is defined above, f(v) is a user-specified weighting factor (varying inversely with v), and p(v) is the probability [computed according to the standard simulated-annealing criterion - see Appendix B] that a tentative move to any given state with value v is accepted. Given v, we choose uniformly among the candidates with that value.}

(ii) Second, with a user-specified (small) positive probability q, choose without duplication the remaining r-1 elements uniformly over the candidate elements with values at least as large as that chosen in (i). {This can be done by standard methods; e.g., see Bratley, Fox, and Schrage (1987, problem 1.9.4) or Devroye (1986, chapter XII). This random selection of the remaining elements assures that the chain is irreducible.} With probability 1-q, choose the remaining r-1 elements with the values at least as large as that chosen in (i) by taking the r-1 among these with smallest objective-function values further modified as follows. Add (another) penalty inversely proportional to the respective Hamming distance of the last component of each candidate from the last component of the element chosen in (i). {The reason only last components are considered here is that previous history is irrelevant in trying to get good spacing between each element chosen in (ii) and the element chosen in (i).}

With this scheme, the tentative-move probabilities are described implicitly, but that does not matter. Even without candidate elements of the form (A), the overall search is much more than a concatenation of local searches because the candidate elements probabilistically "interact" with each other in forming the next state. Fox (1992a) tailors a stratified version of random restarting to generate the initial population.

To set up an equivalence with the scheme in Section 2 [and hence with the original problem], we define a pseudo-objective function: it is the minimum of the (modified) objective-function values of its elements. This is consistent with the convention, adopted in (i) above, for assigning values to states.

The distinction between temporal memory and spatial memory is blurred because spatial memory, as we have defined it, lets us remember previously-considered good elements of the original state space that we might otherwise have forgotten in taking a search direction that originally seemed promising but did not work out well.

We now list some differences between a standard genetic algorithm (SGA), as described in Goldberg (1989) for example, and our scheme:

1. Our scheme is integrated with tabu search and simulated annealing, whereas an SGA is not. {It seems obvious that incorporating temporal memory together with tabu penalties is good. If one believes that cooling speeds up the search, then the search algorithm should use simulated annealing (ideally, we think, on a state space allowing exploitation of temporal and spatial memory). Section 5 gives a rationale for why cooling generally can be expected to help.}

2. Populations in our scheme cannot have duplicate elements, whereas those in an SGA generally do -- thus limiting the scope of the next population.

3. No matter how an SGA is initialized, it can easily lead to a population concentrated in a small region; from that point onwards, its search is essentially local (especially without *mutation*). With our scheme, this (so-called premature

convergence) cannot happen. {Mutation flips bits in the state independently with a low probability for each bit.}

All these differences favor our scheme.

Using candidate elements of the form (A), (B), and (C), as well as those arising from search intensification generates conditional neighborhoods given those elements [in particular, a realization of the random elements of the form (A)]. Each conditional neighborhood remains unchanged for the duration of the corresponding self loop. This makes it practical to compute acceptance probabilities, given such neighborhoods, and hence to use QUICKER. The only state explicitly generated is the next state. Some theoretical issues related to conditional neighborhoods are discussed in Appendix C.

4 LOOP SKIPPING

Let s be state (relative to any of the spaces defined above). suppose that s is entered from a different state on transition i. The algorithm QUICKER(s,i) below generates the transition number N on which the next transition to a different state occurs. In other words, N is the transition number on which the path exits the loop. Our algorithm assumes that the successive temperatures are non-increasing, implying that the successive acceptance probabilities are non-increasing.

We let a(s,h) be the acceptance probability (to move to a different state), given that we are at state s at transition h - see Appendix B for its calculation. Denote by G(s,j) a geometric variate with parameter a(s,j). Let V be a standard uniform variate.

```
Algorithm QUICKER(s,i)
 Set j ← i
 Until exit, repeat
   Generate G(s,j)
   Set N ← j + G(s,j)
   Generate V
   If V < a(s,N)/a(s,j), then
       Exit with N
     Else
       Set j ← N
End
```

With QUICKER, there is a major distinction between simulated time [which includes wheel spinning in self loops] and computer time [which does not]. Thus, papers which analyze certain convergence rates in terms of simulated time give results that would be misleadingly slow if one were to think of simulated time and computer time as having the same order of magnitude. Chiang and Chow (1988) give an inhomogeneous continuous-time Markov-chain version of simulated annealing. They do not say how they would simulate. In particular, the only way which seems practical to generate holding times notes that each such time corresponds to the first arrival in an inhomogeneous Poisson process (with rate function depending on the current state) and then uses the Lewis-Shedler thinning algorithm cited above. This has the effect of introducing self loops corresponding to the rejected arrivals. Relative to discrete-time simulated annealing with QUICKER, it is slower (by a constant factor) and more complex; likewise, for the subsequent generation of the next state. Chiang and Chow consider the probability as a function of (continuous) time that the chain is in the set of optimal states. They find the (slow) rate of convergence of that probability to one as (continuous, simulated) time gets large. A more appropriate measure would be the rate as computer time gets large; on average, the latter is proportional to the number of accepted moves (accepted "arrivals" in this case).

5 COOLING

In practice, one saves the state (relative to the original state space) with the best objective-function value among those visited. Assuming that the Markov chain (over whatever state space above is used} is irreducible, one may then ask why not every tentative move is accepted (as in most versions of probabilistic tabu search). One heuristic answer is that we increasingly want to inhibit uphill moves as the transition number increases. Taken by itself, this answer has led some to use *geometric cooling*: the schedule is piecewise constant, all jumps are downward, the ratio of successive heights of the flat pieces is a constant, and the temperature is bounded away from zero.

Simulated-annealing theory suggests that this answer, though fine as far as it goes, is incomplete. Assuming that the chain is irreducible and weakly-reversible, if cooling is no faster than the canonical rate [defined by replacing U-L by the maximal "depth" among local, non-global minima], then Hajek (1988) shows that the probability that the current state is optimal converges to one as the transition number increases. If a subcanonical cooling rate is chosen to guarantee that the number of accepted moves goes to infinity with probability one as the transition number goes to infinity, then excursions of arbitrary duration and distance away from the set of optimal states occur after any fixed transition number. Together with the convergence-in-probability result, this already suggests that the search path is *attracted* to the

set of optimal states - even from far away. If such excursions did not occur, it would be conceivable that - while short-range attraction is strong - long-range attraction could be weak.

The convergence-in-probability result is not entirely an artifact of the self-loop sequences, because - for example - a local, non-global minimum can be buried deeper with respect to its neighbors than any global minimum is buried with respect to its neighbors. At such local minima, self loops would be stochastically longer than self loops at global minima if they were to begin at the same transition number. Again, this suggests attraction to the set of optimal states. Heuristically, this attraction holds up to the first visit to the set of optimal states as well as to later visits. Only the time to the first visit matters in practice. Although at first sight the convergence-in-probability result has little to say about that time, we have just argued that using a subcanonical cooling rate tends to shorten that time. However, if the goal is only to visit a good (not necessarily optimal) state in a reasonable amount of time, it is not clear whether even a heuristic argument can be given to prefer one schedule over another *unless* one believes that most good states are clustered near the optimal set. If the goal is to find an optimal state, then generally a lot of computer time will be needed and asymptotic results may become relevant. We have argued that they then do become relevant in practice as well as in theory.

6 CONCLUSIONS

We recommend the hybrid algorithm in Section 3, implemented with QUICKER and a subcanonical cooling rate. Combining these features creates synergy.

ACKNOWLEDGMENTS

Fred Glover was apparently the first to call attention to the importance of tabu penalties and diversification. This research was partially supported by the Air Force Office of Scientific Research and the Office of Naval Research Contract # F49620-90-C-0033.

APPENDIX A

Here we show how to compute the quantity n(v) defined in Section 3. Sorting the modified objective-function values of the candidate elements in increasing order, denote their distinct values and respective multiplicities by $c[1] < c[2] < \cdots < c[h]$ and $m(1), m(2), \cdots, m(h)$. Let

$$t(j) = m(j) - 1 + m(j+1) + \cdots + m(h).$$

This is the number of candidate elements with values at least equal to c[j], not counting the one selected in (i) in Section 3. For typographical convenience, let b(i,k) be the number of ways k elements can be selected from i elements. Set

$$w(j) = m(j) \cdot b(t(j),r-1)/2$$

and suppose that t(j) is at least r-1. Of the b(s,r) neighbors, exactly w(j) have c[j] as their smallest element - measured by objective-function value, because the smallest element can be chosen in m(j) ways, the remaining r-1 elements can be chosen in b(t(j),r-1) ways, and the division by 2 avoids double counting. Thus, if v = c[j], then n(v) = w(j). If v differs from c[i] for all i, then n(v) = 0.

APPENDIX B

The notation p(v) for the probability of accepting a move to a state with value v suppresses the value u of the current state and the current transition number j. So, for clarity we rewrite p(v) as p(v;u,j). Denoting the current temperature by T(j), we have

$$p(v;u,j) = \exp\{- \max[0,v-u]/T(j)\}.$$

Thus, downhill moves are always accepted but (assuming that the temperature does not increase with j) the chance of accepting an uphill move decreases with j.

We now calculate the unconditional probability a(x,j) of accepting a move at transition j out of state x with value u by summing $n(v) \cdot f(v) \cdot p(v;u,j)$ over all possible objective-function values v at neighboring states and dividing by the sum over the same range of $n(v) \cdot f(v)$, given a realization of candidate elements of the form (A).

APPENDIX C

Though one can explicitly recast conditional neighborhoods in standard simulated-annealing format by considering unconditional neighborhoods (not given the candidate elements) relative to macrostates (defined below), doing so would be inefficient. However, the fact that this can be done - in principle - is important: due to elements of the form (A) above, when not at a global minimum, the unconditional neighborhood allows a downhill move. Therefore, unlike the usual simulated-annealing situation, the transition matrix conditioned on acceptance does not asymptotically (as the temperature goes to zero) cut off access from any state to an optimal state. Not only is such access heuristically

appealing, it also has positive theoretical implications as Fox (1992) details. Here is the promised definition of a *macrostate*: it is a state, as defined in section 3, together with the candidate elements of the form (A) above for the next state. Using conditional neighborhoods streamlines a messy equivalent formulation of transition probabilities among macrostates.

It is convenient to require that the penalty for an element be a non-increasing function of the number of its dashed components. With this convention, it is routine to check that Hajek's (1988) weak-reversibility condition holds.

REFERENCES

Bratley, P., B. L. Fox, and L. E. Schrage. 1987. *A guide to simulation*. Second edition. New York: Springer-Verlag.

Chiang, T.-S., and Y. Chow. 1988. On the convergence rate of annealing processes. *Siam J. Control Optim.* 26:1455-1470.

Devroye, L. 1986. *Non-uniform random variate generation*. New York: Springer-Verlag.

Fox, B. L. 1992a. Integrating and accelerating tabu search, simulated annealing, and genetic algorithms. Technical report: University of Colorado, Denver. To appear in *Annals of Operations Research.*

Fox, B. L. 1992. Faster simulated annealing. Technical report: University of Colorado, Denver.

Goldberg, D. E. 1989. *Genetic algorithms in search, optimization, and machine learning*. Reading, Mass.: Addison-Wesley.

Greene, J. W., and K. J. Supowit. 1986. Simulated annealing without rejected moves. *IEEE Transactions on Computer-Aided Design* 5:221-228.

Hajek, B. 1988. Cooling schedules for optimal annealing. *Mathematics of Operations Research* 13:311-329.

Keilson, J. 1979. *Markov-chain models - rarity and exponentiality*. New York: Springer-Verlag.

Lewis, P. A. W., and G. S. Shedler. 1979. Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly* 26:403-414.

Osborne, L. J., and B. E. Gillett. 1991. A comparison of two simulated annealing algorithms applied to the directed Steiner problem on networks. *ORSA Journal on Computing* 3:213-225.

AUTHOR BIOGRAPHY

BENNETT L. FOX received his Ph.D. in Engineering Science [Operations Research] from the University of California, Berkeley. After working for six years at the RAND Corporation in Santa Monica, California, he spent a year as Ford Visiting Professor at the Graduate School of Business, University of Chicago. Next, he became a professor at the University of Montreal with sabbatical leaves at Yale and at Cornell. He is now a professor in the mathematics department of the University of Colorado, Denver. While he has published papers in several fields, since roughly 1978 his interests have shifted mainly to simulation and computational probability. Recently, he includes in the latter probabilistic methods for global optimization, including simulated annealing, tabu search, genetic algorithms, and combinations thereof.