Greedy Local Search

Bart Selman
Dept. of Computer Science
Cornell University
Ithaca, NY 14853
selman@cs.cornell.edu

Greedy local search search methods have a long history in the area of combinatorial optimization. One of the earliest application of local search was to find good solutions for the Traveling Salesman Problem (TSP). In this problem, one is given a weighted graph and the goal is to find the shortest closed path that visits each node exactly once. The TSP is prototypical of the large class of NP-hard optimization problems, for which it is widely believed that no efficient (polynomial time) algorithm exists (Cook 1971; Garey and Johnson 1979; Papadimitriou and Steiglitz 1982). A local search method for the TSP proceeds as follows. Start with a randomly generated closed path. Subsequently, we make small ("local") changes to the path to try to find a shorter one. One example of such a local change is the so-called $k-$change, in which we remove k edges from the path and replace them with k other edges. A $k-$change is made whenever it improves the current solution. If no more local improvement can be found, the procedure terminates. Lin (1965) and Lin and Kernighan (1973) showed that this simple procedure, using $k = 3$, leads to solutions that are surprisingly close to the optimal solution.

The basic local search framework allows for several variations. For example, there is the choice of the initial solution, the nature of the local changes considered, and the manner in which the actual improvement of the current solution is selected. Lin and Kernighan found that multiple runs with different random initial paths would lead to the best solutions. Somewhat surprisingly, starting with "good" initial paths did not necessarily lead to better final solutions. The reason for this appears to be that the local search mechanism itself is powerful enough to improve upon the initial solutions — often quickly giving better solutions than those generated using other methods. As for the nature of the local changes considered, Lin and Kernighan found that a $3-$change approach is clearly better than considering only $2-$changes, but the additional computational cost of considering 4-changes does not appear to pay off. In general, finding the right local changes to consider requires an empirical evaluation of strategies. Another issue is that of how to select the actual changes made to the current solution. The two extremes are: *first-improvement* (also called "hill-climbing"), in which any favorable change is accepted, and *steepest-descent*, in which the best possible local improvement is selected at each step. Steepest-decent is sometimes referred to as *greedy* local search, but this term is also used to refer to local search in general.

A local search method does not necessarily reach a global optimum because the algorithm terminates when it reaches a state where no further improvement can be found. Such states are referred to as *local optima*. In 1983, Kirkpatrick *et al.* introduced a technique for escaping from such local optima. The idea is to allow the algorithm to make occasional changes that do not improve the current solution, *i.e.*, changes that lead to equally good or possibly inferior solutions. Intuitively speaking these non-improving moves can be viewed as injecting *noise* into the local search process. Kirkpatrick *et al.* referred to their method as *simulated annealing*, as it was inspired by the annealing technique used to reach low energy states in glasses and metals. The amount of "noise" introduced is controlled with a parameter, called the temperature T. Higher values of T correspond to more noise. The search starts off at a high temperature, which is slowly lowered during the search in order to reach increasingly better solutions.

Another effective way of escaping from local minima is the *tabu search* method (Glover 1989). During the search, the algorithm maintains a "tabu" list containing the last L changes, where L is a constant. The local search method is prevented from making a change that is currently on the tabu list. With the appropriate choice of L, this methods often forces the search to make upward (non-improving) changes, again introducing noise into the search process.

GENETIC ALGORITHMS can also be viewed as performing a form of local search (Holland 1975). In this case, the search process proceeds in parallel. Solutions are selected based on their "fitness" (*i.e.*, solution quality) from an evolving population of candidates. Noise is introduced in the search process via random mutations.

A recent new area of application of local search methods is in solving NP-complete *decision problems*, such as the Boolean satisfiability (SAT) problem. An instance of SAT is a Boolean formula consisting of a set of clauses (constraints), where each clause is a disjunction of literals. A literal is a Boolean variable or its negation. The goal is to determine whether a given instance is satisfiable or not. That is, does there exist an assignment to the Boolean variables such that all clauses are satisfied? The problem is NP-complete for formulas with at least three variables per clause (Cook 1971). The best traditional methods for solving the SAT problem are based on a systematic backtrack-style search procedure, called the Davis, Putnam, and Loveland procedure (Davis and Putnam 1960; Davis *et al.* 1962). These procedures can currently solve hard, randomly generated instances with up to 400 variables (Mitchell *et al.* 1992; Crawford and Auton 1993; and Kirkpatrick and Selman 1994). In 1992, Selman *et al.* showed that a greedy local search methods, called GSAT, could solve instances with up to 700 hundred variables. Recent improvements on the local search strategy enables us to solve instances with up to 3000 variables (Selman *et al.* 1994). For closely related work, in the area of scheduling, see Minton *et al.* (1992).

An important difference in applying local search to decision problems, as opposed to optimization problems, is that *near*-solutions are of no particular interest. For decision problems, the goal is to find a solution that satisfies all constraints of the problem under consideration. (See also CONSTRAINT SATISFACTION and HEURISTIC SEARCH.) In practice, this means that, for example, GSAT and related local search procedures spend most of their time satisfying the last few remaining constraints. Recent work has shown that incorporating random walk style methods in the search process greatly enhances the effectiveness of these procedures.

Since Lin and Kernighan's successful application of local search to the TSP, and the many subsequent enhancements to the local search method, local search techniques have proved so powerful and general, that such procedures have become the approach of choice for solving large combinatorial search problems.

References

[1] Cook, S.A., The complexity of theorem-proving procedures. *Proc. STOC-71* (1971) 151–158.

[2] Crawford, J.M. and Auton, L.D. Experimental results on the cross-over point in satisfiability problems. *AAAI-93* (1993) 21–27.

[3] Davis, M., Logemann, G., and Loveland D. A machine program for theorem-proving. *Comm. Assoc. for Comput. Mach.*, 5 (1962) 394–397.

[4] Davis, M. and Putnam, H. A computing procedure for quantification theory. *J. of the ACM*, 7 (1960) 201–215.

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

[6] Glover, F. Tabu search — part I. *ORSA Journal on Computing*, 1(3) (1989) 190–206.

[7] Holland, J.H. *Adaptation in natural and artificial systems.* University of Michigan Pres.

[8] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. Optimization by simulated annealing. *Science*, 220 (1983) 671–680.

[9] Kirkpatrick, S. and Selman, B. Critical Behavior in the Satisfiability of Random Boolean Expressions. *Science*, 264 (May 1994) 1297–1301.

[10] Lin, S. Computer solutions of the traveling salesman problem. *BSTJ*, 44, no 10 (1965), 2245–69.

[11] Lin, S. and Kernighan, B.W. An effective heuristic for the traveling-salesman problem. *Oper. Res.* 21 (1973) 498–516.

[12] Minton, S., Johnston, M., Philips, A.B., and Laird, P. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58 (1992) 161–205,

[13] Mitchell, D., Selman, B., and Levesque, H.J. Hard and easy distributions of SAT problems. *Proc. AAAI-92*, San Jose, CA (1992) 459–465.

[14] Papadimitriou, C.H. and Steiglitz, K. *Combinatorial optimization.* Englewood Cliffs, NJ: Prentice-Hall, Inc. (1982).

[15] Selman, B., Kautz, H.A., and Cohen, B. Noise strategies for improving local search. *AAAI-94,* Seattle, WA (1994) 337–343.

[16] Selman, B., Levesque, H.J., and Mitchell, D. A new method for solving hard satisfiability problems. *Proc. AAAI-92*, San Jose, CA (1992) 440–446.