A user's guide to tabu search*

Fred Glover

Graduate School of Business, University of Colorado, Boulder, CO, USA

Eric Taillard and Dominique de Werra

Department of Mathematics, Swiss Federal Institute of Technology, Lausanne, Switzerland

Abstract

We describe the main features of tabu search, emphasizing a perspective for guiding a user to widerstand basic implementation principles for solving combinatorial or nonlinear problems. We also identify recent developments and extensions that have contributed to increasing the efficiency of the method. One of the useful aspects of tabu search is the ability to adapt a rudimentary prototype implementation to encompass additional model elements, such as new types of constraints and objective functions. Similarly, the method itself can be evolved to varying levels of sophistication. We provide several examples of discrete optimization problems to illustrate the strategic concerns of tabu search, and to show how they may be exploited in various contexts. Our presentation is motivated by the emergence of an extensive literature of computational results, which demonstrates that a well-lWled implementation makes it possible to obtain solutions of high quality for difficult problems, yielding outcomes in some settings that have not been matched by other known techniques.

Keywords: Tabu search, heuristics, combinatorial optimization, artificial intelligence.

1. Introduction

The abundance of difficult optimization problems encountered in practical settings (e.g. telecommunications, logistics, financial planning, transportation and production) has motivated a proliferation of optimization techniques.

Researchers have adapted ideas from many areas in the quest to develop more efficient procedures. Network flow methods, for example, share a heritage with models exploiting ideas from electricity and hydraulics. Simulated annealing is based on a physical process in metallurgy. while so called genetic methods seek to imitate the biological phenomenon of evolutionary reproduction.

*The research of this paper has been supported in part by the joint Air Force Office of Scientific Research and Office of Naval Research Contract No. F49620-90-C-0033, at the University of Colorado and by the Swiss National Research Cowicil Grant No. 20-27926.89.

In this vein, the Tabu Search {TS) method elaborated in this paper may be regarded as a technique based on selected concepts from artificial intelligence. TS is a general heuristic procedure for guiding search to obtain good solutions in complex solution spaces. Its rules are sufficiently broad that it is often used to direct the operations of other heuristic procedures.

One of the main components of TS is its use of *flexible (adaptive)* memory, which plays an essential role in the search process. Discoveries of more refined ways to exploit this memory, and of more effective ways to apply it to special problem settings, provide one of the key research thrusts of the discipline and account for its growing success in treating hard problems.

The organization of this paper is as follows. First, we give a general presentation of TS in the next section. Then we describe some of the features that increase the efficiency of the technique in section 3. Finally, concluding observations are provided in section 4. The goal of our development is to disclose the applicability of TS as a general tool which may be adapted to a wide range of problems. As demonstrated in the literature, performance levels are achieved for the problem domains studied that frequently surpass those of the procedures previously established to be best.

We stress, however, that this paper is not intended as an exhaustive characterization of the elements of TS, but as a first introduction to the nonspecialist and as a foundation to suggest useful directions for advanced study. Our goal is to provide a basic understanding of the procedure that may permit its principles to be applied more easily to the solution of hard problems, and to give a glimpse of additional developments that lie ahead.

2. What is tabu search?

With roots going back to the late 1960's and early 1970's, TS was proposed in its present form a few years ago by Glover (see, e.g. [13]), and has now become an established optimization approach that is rapidly spreading to many new fields. The basic ideas of TS have also been sketched by Hansen [20]. Together with simulated annealing and genetic algorithms, TS has been singled out by the Committee on the Next Decade of Operations Research [3] as "extremely promising" for the future treatment of practical applications.

We begin by sketching the elements of TS in rough terms. The method can be viewed as an iterative technique which explores a set of problem solutions, denoted by X, by repeatedly making moves from one solution s to another solution s' located in the neighborhood N(s) of s.

These moves are performed with the aim of efficiently reaching a solution that qualifies as "good" {optimal or near-optimal) by the evaluation of some objective function $f(s)$ to be minimized.

As an initial point of departure, we may contrast TS with a simple descent method, which may be formulated as follows:

(a) Choose an initial solution s in X
stop: = false

Repeat:

(b) Generate a sample V^* of solutions in $N(s)$.
Find a best s' in V^* (i.e. such that
$f(s') \leq f(s)$ for any s in V^*).
if $f(s') \quad f(s)$ then stop: = true
else $s := s'$ (2.1)

Until stop

By narrowing the choice of s' to a best element of V^* we are restricting the class of descent methods to a proper subset of the larger collection sometimes called "improving methods" – a restriction that has important consequences for the goal of restructuring this approach by tabu search.

First note that the most straightforward implementation of the preceding approach is to take $V^* = N(s)$, i.e. to scan the entire neighborhood of s. In some circumstances, however, such a process is impractical because it is too time-consuming. Therefore, in such cases we stipulate that a set $V^* \subset N(s)$ with $|V^*| \quad |N(s)|$ be scanned. In the extreme, one may take $|V^*| = 1$, thereby eliminating a direct comparison among elements, and dissolving the distinction between a best element and an arbitrary member of V^*. This latter form of sampling in fact underlies the policy adopted by simulated annealing (which also may be viewed as a way of restricting a descent method), employing rejection criteria that may result in generating a number of sets with $[V^*[= 1$.

Stepping beyond the case where $|V^*| = 1$, the preceding descent technique can be given an added dimension of refinement by considering its implicit reliance on solving a local optimization problem (either by itemization or more intricate means), to find $an\ s'$ that yields $\min \{f(s') | s' \in V^* \quad N(s)\}$, possibly in an approximate way. Building on this viewpoint, a key concept of tabu search is to constrain the definition of this local optimization problem in a strategic way – making systematic use of memory to exploit knowledge beyond that contained in the function $f(s)$ and the neighborhood $N(s)$. This also leads to a general orientation whereby the set V^* is to be generated strategically rather than randomly, with the goal of allowing the minimum of $f(s')$ over V^* to be more nearly representative of the minimum over $N(s)$, subject to the special restructuring of $N(s)$ derived by reliance on memory. In this sense, we may call the method a metaheuristic, since at each step it uses a heuristic to move from one solution to the next, guiding the search in X

Why should such a search be guided? If the descent approach is applied as described in (2.1) it will (at best) lead to a local minimum $of\ f$ where it will be trapped. This is the first type of difficulty that can arise with standard heuristic descent techniques.

To circumvent such an outcome, a guidance procedure must be able to accept a move from s to s' (in v^*) even if $f(s') > f(s)$. But when a solution s' worse than s may be accepted, cycling may occur, causing the process again to be trapped by returning repeatedly to the same solution (perhaps after some interval of intermediate solutions). Simulated annealing attempts to provide guidance by accepting a disimproving s' (if it happens to be the one currently sampled) with a certain probability that depends uponf (s), $f(s')$ and a parameter identified with temperature. The tabu search approach, by contrast, seeks to counter the danger of entrapment by incorporating a memory structure that forbids or penalizes certain moves that would return to a recently visited solution. The variant called *probabilistic tabu search* creates probabilities for accepting moves as a monotonic function of evaluations created by a penalty of the method.

The notion of using memory to forbid certain moves (i.e. to render them tabu) can be formalized in general by saying that the solution neighborhood depends on the time stream, hence on the iteration number k. That is, instead of $N(s)$ we may refer to a neighborhood denoted $N(s,k)$. (More precise notation would indicate that $N(s,k)$ depends on solutions and neighborhoods encountered on iterations prior to k.) For instance, suppose memory is employed that recalls solution transitions over some time horizon, and creates $N(s,k)$ by deleting from $N(s)$ each solution that was an immediate predecessor of s in one of these transitions. We may express the form of the procedure that uses the modified (tabu) neighborhoods as follows.

(a) Choose an initial solution s in X
$s^* := s$ and $k := 1$

(b) While the stopping condition is not met do
k := k + 1
Generate $v^* \quad N(s,k)$
Choose the best s' in v^*
$s := s'$
if $f(s') < f(s^*)$, then $s^* := s'$ (2.2)
end while

There may be several possible stopping conditions, but simplest is some logical combination of the following:

- An optimal solution is found.
- $N(s,k+1) = O$.
- k is greater than the maximum number of iterations allowed.
- The number of iterations performed since s^* last changed is greater than a specified maximum number of iterations.

It is to be emphasized again that a nontrivial, strategically generated sample of the solution neighborhood is examined at each step, and a best element from the

sample is selected (subject to avoiding moves that are classified tabu). The goal is therefore to make improving moves to the fullest extent allowed by the structure of *N(s,* k), balancing trade-offs between solution quality and computational effort in examining larger samples.

Characterized in this way, TS may be viewed as a *variable neighborhood method:* each step redefines the neighborhood from which the next solution will be drawn, based on the conditions that classify certain moves as tabu.

A crucial aspect of the procedure involves the choice of an appropriate definition of *N(s,k).* Due to the exploitation of memory, *N(s,k)* depends upon the trajectory followed in moving from one solution to the next. As a starting point, consider a form of memory embodied in a tabu list *T* that records the I Tl solutions most recently visited, yielding *N(s,k)* = N(s) – *T.*

Such a *recency based* memory approach will prevent cycles of length less than or equal I TI from occurring in the trajectory. This formulation normally has only a theoretical interest, since keeping track of the I Tl most recent solutions may be extremely space consuming. (It also represents a relatively limited form of *recency based* memory.)

As a basis for identifying a more practical and effective type of memory structure, we may amend our definition of *T* by defining the neighborhood *N(s)* in terms of moves for transforming the solution *s* into new solutions. Specifically, for each solution s consider a set *M(s)* of legal moves *m* which can be applied to s in order to obtain a new solution s' = *s* Ea *m.* Then we have *N(s)* = {s'l 3 m e *M(s)* with *s'* = *s* Ea m}. Generally in TS it is useful to employ a set of moves *M(s)* that is reversible, i.e. 'rim e *M(s),* 3 m-^{1} e M(M(s)) such that *(s* Ea m) Ea m-^{1} = *s.* With this stipulation, *T* may consist of the last I Tl "reverse moves" associated with the moves actually performed.

Instead of a single tabu list *T* it may be more convenient to use several lists *Ti.* Then a tabu status is assigned to some constituents *Ti* of *m* (or of s) to indicate that these are currently forbidden for composing a move. In general, the tabu status of a move is a function of the tabu status of *attributes* that change in going from the current solution to the next (such as variables that change values, or elements that exchange positions, or items that are transferred from one set to another).

This leads to a collection of *tabu conditions* of the form

$$\text{ti(m,s)} \text{ e } 1i \text{ (}i = 1, \ldots, p). \tag{2.3}$$

A move *m* applied to the current solutions will then be a tabu move (i.e. implicitly a member of the list T) if all conditions (2.3) are satisfied.

We now briefly illustrate some of the preceding concepts with a simple example [42] where TS has produced very good solutions; the quadratic assignment problem (QAP). Subsequently, we will use this example to motivate the discussion of additional elements fundamental to tabu search.

QAP FORMULATION

N items (workshops of a factory) have to be assigned to *N* different locations. The distance *aii* between locations *i* and *j* is given as well as the value b_{pq} of a flow (circulation of parts) between the items *p* and *q*.

An assignment of the items to the locations is sought that minimizes the sum of the products of distances and associated flow values. Mathematically, the problem consists in finding a permutation ¢ of {1, . . . , N} to minimize

$$f(\phi) = \sum \sum a_{ij} b_{\phi(i)\phi(j)}. \tag{2.4}$$

For a complete formulation and some applications the reader is referred to Burkard [1]. Applying our previous notation, in this problem, *X* consists of the N! permutations of {1, . . . , *NJ ;* a move *m* in *M(* ¢) may consist of the exchange of two items *p* and *q.* The permutation *1&* obtained from ¢ by move *m* then is given by

$$\left.\begin{aligned} 1\& &= \phi\, fB\, m \\ 1\&(i) &= 4>(i) \\ 7r(p) &= 4>(q) \\ 1\&(q) &= 4>(p) \end{aligned}\right| \quad \text{for all } i : t: p,q.$$

The difference/(¢ fB m) -/(¢) can be computed easily, permitting exploration of the entire neighborhood *N(t/>)* in time $O(N^2)$ (see Finke et al. [7]).

One possibility for creating a recency based memory is to identify attributes of a move by the unordered pair of items *(p, q).* The attribute pair that must be forbidden in order to prevent the reverse move likewise is *(p,q)* (disallowing these two items to be exchanged again, noting that an immediate exchange of this type would revisit the solution just left). Making this attribute pair the basis of a tabu restriction permits each of *p* and *q* independently to be exchanged at once with other items. Unfortunately, however, this type of restriction does not satisfy the property of forbidding a return to solutions already visited, since the sequence of moves *(q,r) (p,s) (q,s) (p,r) (p,q) (r,s)* does not change anything.

Another way of identifying attributes of an exchange move is to introduce additional information, referring not only to items exchanged but to positions occupied by these items at the time of exchange. To do this without the memory burden of introducing 4-element vectors, we conceive a move to be composed of two half-moves *(p,* t/J(q)), the first placing item *p* in location /(q) and the second placing item *q* in location *t/>(p).* The reverse move is prevented by disallowing the two attribute pairs *(p,* t/J(p)) and *(q,*t/J(q)) from occurring simultaneously, and it is these two pairs that are stored in the tabu list of our TS implementation for QAP.

More precisely, we define a move *m* (exchange of items *p* and *q)* to be tabu if it sends both *p* and *q* to locations they occupied within the last *t* iterations. This

tabu condition admits some moves prevented by the previously illustrated condition. and also forbids some moves permitted by the previous condition, hence is neither uniformly more restrictive or less restrictive.

The current restriction has the appeal of being based not only on attributes of solutions (corresponding to items located in specified positions). In addition, we allow the size t of the primary tabu list, consisting of (item, location) pairs, to be changed during the solution process.

We now return to a discussion of general TS features before completing the QAP illustration.

2.1. ADDmONAL TABU SEARCH ELEMENTS: ASPIRATION CRITERIA

It is not difficult to realize that tabu conditions based on selected attributes of moves and solutions may be too drastic in the sense that they may also forbid moves leading to unvisited solutions, and in particular to unvisited solutions that may be attractive. It is therefore necessary to overrule the tabu status of moves in certain situations. This is performed by means of *aspiration level conditions.*

As an example, consider a tabu move m; if this move gives a new solution which is better than the best found so far, then one would be tempted to drop the tabu status of m and accept the move. This can be done by saying that m (applied to s) may be accepted if it has a level of aspiration $a(s,m)$ which is better than a threshold value $A(s,m)$. More generally, we may conceive of $A(s,m)$ as defining a set of preferred values (for elements) for a function (or mapping) $a(s,m)$, and aspiration may be represented in the form

$$a_i(s,m) \in A_i(s,m) \quad (i = 1, \ldots, l). \tag{2.5}$$

Then the tabu status of a move m will be rendered inoperative if at least one (or a specified number) of the conditions (2.5) is satisfied. If many aspiration conditions are simultaneously satisfied by more than one move, rules of choice must be defined.

In the QAP illustration, we use the simple type of aspiration criterion that accepts a tabu move if it produces a new best solution. Notationally, this corresponds to defining $a(s,m) = f(s \oplus m)$ and letting $A(s,m)$ be the interval of values smaller than $f(s^*)$, where s^* is the best solution found so far. In addition, for types of problems where the entries (bpq) of the flow matrix span a large sample of numerical values, we use a second type of aspiration condition which may be denoted by $a'(s,m) \in A'(s,m)$. Under this condition, a tabu move m that sends item p to location j and item q to location i is accepted (no matter what the value of $f(s \oplus m)$ may be) if p was not at j and q was not at i within the last u iterations. In such a case, $A'(s,m)$ contains every move (or "pair of half moves") that has not been performed during the preceding u iterations, and $a'(s,m) = m$. Obviously, the value of u has to be set at a value larger than the size of the neighborhood; for example, it turns out that a value of ten times the size of the neighborhood is convenient for most QAP instances.

2.2. INTENSIFICATION AND DIVERSIFICATION

Besides the above described components. TS requires a few additional ingredients in order to behave as an intelligent search technique. The use of a memory up to now has been limited to a short term horizon: TS remembers the most recently performed steps in order to avoid coming back to a solution visited earlier, or to one of a class of solutions visited earlier, where the class is implicitly determined by attributes used to define tabu status.

Memory is also used in TS in a kind of learning process: having visited several solutions, it is deemed worthwhile to observe whether the good solutions visited so far have some common properties (such as the presence or absence of certain elements). This generates an *intensification* scheme for the search. At some stage the neighborhood $N(s,k)$ is restricted (or the criteria for evaluating moves are altered) to favor solutions with properties that occurred often in good solutions previously visited, or more generally in subsets of good solutions differentiated by clustering criteria. This form of bias can be used either to "structure" the current solution to satisfy such properties (as by a partial restart procedure), and then to discourage the properties from being violated, or can simply be used to encourage such properties to become incorporated as the method progresses. Some of the relevant considerations are discussed in the context of *consistent* and *strongly determined* variables in [12].

These ideas are lately finding favor in other procedures, and may provide a bridge for integrating components of tabu search with components of other methodologies. For example, a first level strategy to reinforce the incorporation of consistent elements from high quality solutions (without undertaking differentiation by clustered subsets) has proved beneficial in genetic algorithms. In the setting of the traveling salesman problem, where ingenuity makes it possible to identify "crossover" operations that preserve shared structures [45], these ideas have produced very good results. In particular, Muhlenbein [33] has noted the relevance of genetic algorithms of this type for classes of traveling salesman problems that satisfy a *building block property,* in which optimal (or exceedingly good) tours can be pieced together from segments of logically optimal tours. In this case, by augmenting a genetic algorithm approach with local optimization, and applying a "maximal edge-preserving crossover", impressive outcomes are achieved for a number of classical test problems. Similar uses of such ideas by Ulder et al. [43] and Kohlen and Pesch [27] have also produced very attractive results for these problems.

The effectiveness of such a special instance of the intensification strategy in the TSP context suggests that broader forms of the strategy deserve to be examined more closely in other settings, where the building block phenomenon may not apply. Approaches to reinforce elements of elite solutions based on clustering and statistical discrimination (e.g. incorporating frequency based memory over attribute groupings), illustrate natural foundations for such a strategy. From a more specialized perspective, approaches such as the use of *scatter search* and *structured*

combinations [17] provide an opportunity to establish links with genetic algorithms, while permitting the space of solutions to be exploited in ways unavailable to genetic crossover, and may prove useful in this context.

Intensification by itself is insufficient to yield the best outcomes for general classes of optimization problems. The complementary notion of *diversification* must be invoked to allow the most effective search over the set X. Strategic pursuit of solutions with varying characteristics provides an essential counterbalance to the intensification component of tabu search.

Thus, over the longer term, tabu search induces an alternation between sequences of steps that intensify the search in a promising region and steps that diversify the search across contrasting regions. One way to do this is by designing $N(s,k)$ to incorporate solutions that are separated by a specified degree from solutions visited before. An evident form of diversification results by modifying the objective function to decrease f for solutions "far from s" and to increase f for solutions "close to s". When s is replaced by a set S of previously generated solutions, the meanings of "far from" and "close to" become more subtle, and typically involve references to frequency based memory as well as recency based memory. Several specific ways of realizing diversification are discussed with the applications that follow.

The preceding elements summarize the basic ingredients needed to design a TS procedure. To derive a highly efficient metaheuristic, special issues of design and implementation must be addressed. We discuss some of these issues in the next section.

3. Refinements of TS

In considering refinements to obtain good solutions more efficiently than by an elementary TS procedure, we focus on easy-to-implement methods that have proved their merit, in the spirit of favoring techniques that are both easy to program and capable of achieving effective outcomes. Our illustrated adaptations to the QAP (introduced in section 2) provides a useful basis for exemplifying some of these features. The method succeeds in obtaining some of the best outcomes in the QAP literature by means of a Pascal program of approximately 2 pages!

We have already mentioned two diversification techniques that also turn out to be valuable in the QAP application. First is a simple strategy of varying the tabu list size. Second is an approach of variably penalizing the objective function to destroy the structure of the local minima one wants to escape, subsequently called the shifting penalty approach.

In general, an effective implementation of TS may be achieved by focusing on three levels:

(1) tactical,

(2) technical,

(3) computational.

The tactical level seeks to embed greater intelligence in the search process. The technical and computational levels are chiefly concerned with limiting the possibility of cycling and with speeding-up the iterative computation of the search.

Ways to introduce improvements in each of the areas will be reviewed in the next sub-sections.

3.1. TACTICAL IMPROVEMENTS

The first question to ask in designing an iterative search procedure is related to the quality of the chosen neighborhood: "Are the defined moves good?" Frequently, the answer to this question must be based on intuitive good sense, and on practical experimentation with alternatives. However, the tools of statistical analysis and graphical simulation can be highly useful in this quest.

Simple statistics may disclose significant information about the nature of the moves performed by the search. For example, if the amplitude of objective function changes produced by the moves is very small, then it is reasonable to think that the search will have some trouble discovering good trajectories for escaping from local minima (since many trajectories look the same). On the other hand, if this amplitude is quite high, the search may have some difficulty finding local minima whose quality is close to that of a global minimum.

When both problem type and move type are susceptible to graphical visualization, exhibiting the solutions generated by the search can help to establish the appropriateness (or inappropriateness) of the neighborhood used to define admissible moves. This is relevant even where relatively simple moves are used as a foundation of a search procedure. While simple moves can be valuable for developing a method without a heavy investment of human time and resources (particularly for problems that have not been extensively studied, and whose subtleties are unknown), they are also susceptible to weaknesses that make it important to identify special "adjunct moves" that periodically can be invoked to counter these weaknesses. Such adjunct moves then fulfil the role of a diversification strategy, altering the terrain visited in a way that is unlikely to occur by applying the simple moves alone.

We give an example for the Euclidean traveling salesman problem, where graphical simulation led to identifying a useful diversification move as an adjunct to a rudimentary move structure (see Fiechter [6]). (Subsequently we identify additional ways to generate more powerful alternatives.) The problem in this case is defined to a collection of randomly placed cities on a unit square, where the goal is to obtain a shortest tour, that is, a closed path that passes exactly once through every city before returning to the first city visited.

A highly popular and very simple move for the traveling salesman problem is the "2-opt" move (see fig. 1), which is applied iteratively to transform an initial tour into one that is locally optimal (i.e. that cannot be further improved by such moves).

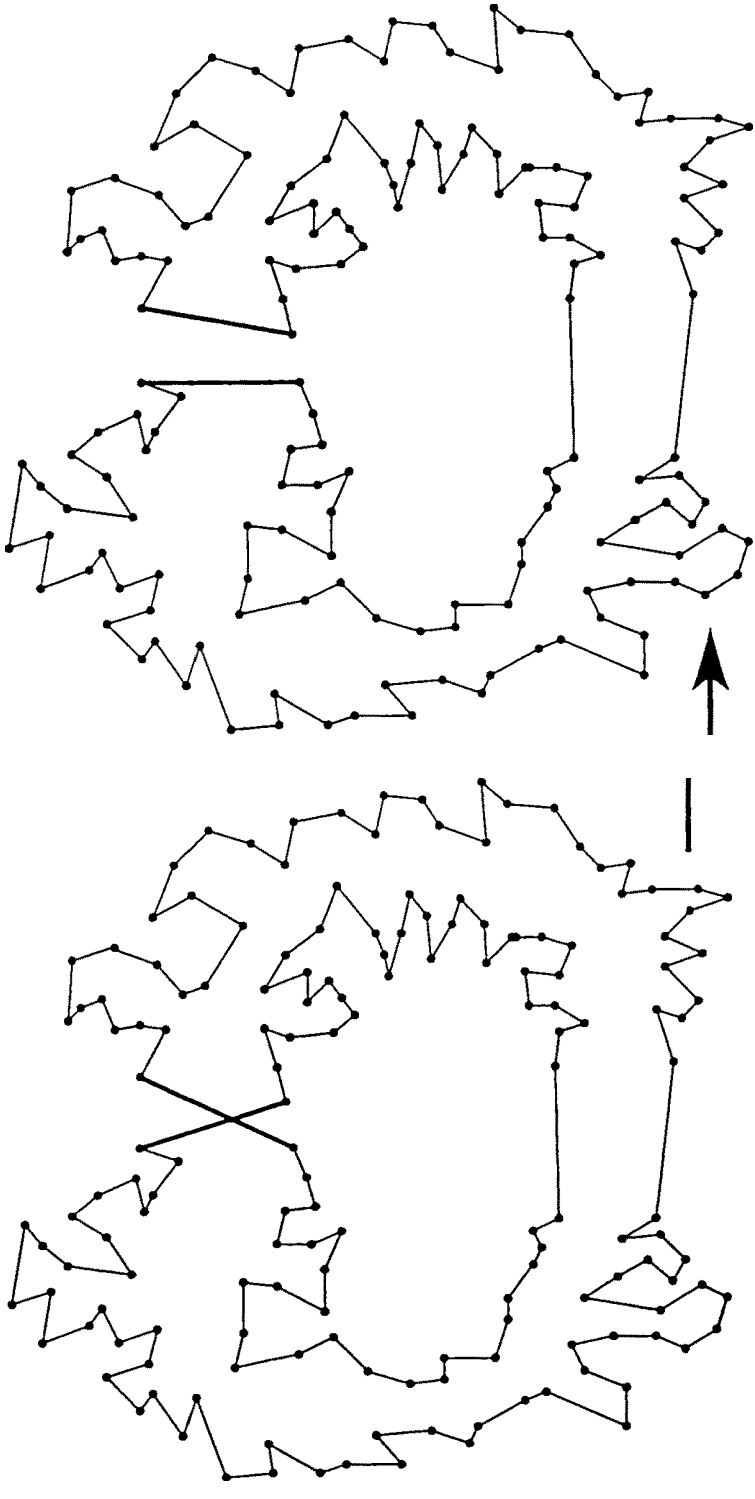

Fig. 1. 2-opt move.

Such 2-opt moves make it possible to quickly find reasonably good local optima of the problem. However, a number of other moves, requiring more effort to implement, are known to be stronger. A TS method based on 2-opt moves is partly affected by their short-sightedness, and typically will "cross" two short edges when it initiates a retreat from a local minimum. (Crossed edges are known to be undesirable in Euclidean traveling salesman problems.) If the tabu list size is too small (which implies that more of the small 2-opt changes are required to escape the basin of attraction than the size of the list), the search will cycle around a local minimum, successively creating and removing small crosses. Conversely, if the size of the list is somewhat larger, moves that allow escape from a local minimum will indeed be performed, but the tabu restrictions may render it impossible (without knowledge of an appropriate aspiration condition) to remove the small crosses previously created. Under these circumstances, 2-opt moves by themselves evidently do not define a sufficiently rich neighborhood to obtain very good results.

Using graphical representations of relatively small problems not only disclosed this phenomenon, but also disclosed a way to create a type of diversifying move to counter it. Specifically, comparing solutions obtained by a "2-opt neighborhood version" of TS with known optimal solutions, a relevant diversifying move was discovered to have the form shown in fig. 2.

The new move usefully modifies and expands the search alternatives at the local level, leading to configurations that are unlikely to be visited by a search based on 2-opt moves (since three 2-opt moves are needed to achieve the same outcome as one of the new moves, and one of the component moves may have a very high cost).

Once the diversification effect of the new move is achieved, the simpler 2-opt moves are used to refine the outcome by progressing to a good local optimum. Sequences of 2-opt moves leading to local optima are therefore alternated with sequences of diversifying moves to escape from these local optima.

It should be noted that, among more advanced moves that are incorporated into heuristics for TSPs, a class of compound moves due to Lin and Kernighan (30] has long been known for its effectiveness, and has been incorporated into a highly efficient procedure by Johnson [24}. Also, a new class of "generalized insert" moves recently has been proposed by Gendreau et al. (10] which shows considerable promise. At this writing, neither of these types of moves has been embedded in a tabu search procedure for TSPs, but gains may be expected by doing so, and a similar application of graphical analysis may offer the possibility to enhance such potential implementations.

3.1.1. Discovering improvements by target analysis

Although graphical simulation is admittedly helpful, many types of optimization problems do not lend themselves readily to its use. Can anything be done to determine

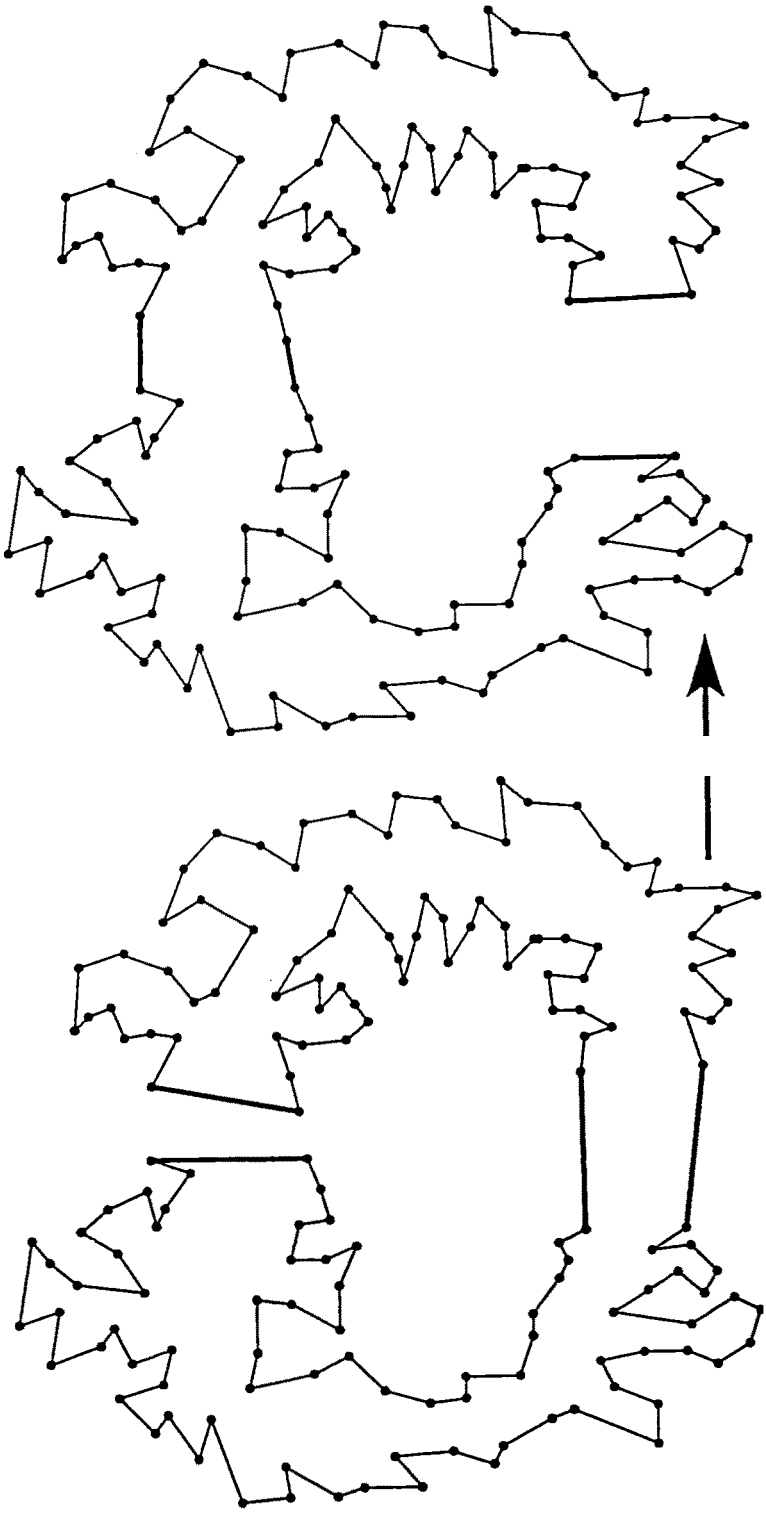

Fig. 2. A diversifying move.

the nature of improved strategies for these problems? Fortunately, the answer to this question is yes.

A learning approach called *target analysis* is conveniently suited for designing improved implementations of tabu search. To begin, target analysis launches an extensive effort to obtain optimal or very high quality solutions to sample problems from the class to be solved. This effort is based on investing greater solution time than normally would be appropriate, and may incorporate a variety of schemes to obtain the best outcomes currently possible. A set consisting of one or more of the best solutions generated for each problem during this initial phase provides the targets to be sought during the next phase of the approach.

The second phase re-solves each sample problem with the goal of collecting information to determine what rules will make the "right choices" to guide successive current solutions to a best solution (e.g. a target solution closest to the current one). Provisional rules are designed as parameterizations of the information collected. To focus on identifying relevant information about preferred solution trajectories, choices may be made during this second phase based on "illegitimate" information derived from hindsight, permitting uniformly good moves to be selected for approaching a target solution, although possibly no currently known rules would prescribe these choices.

The third phase consists of characterizing evaluation functions to give a master decision rule, based on establishing specific combinations and parameter values for the provisional rules of the second phase. Two useful ways to link the information of the second phase to the master decision rule of the third phase are by discriminant analysis [19] and the creation of *move scores* [29].

Move scores, which identify how moves preferably should be evaluated, in contrast to how they actually are evaluated, have provided a useful discovery for scheduling problems that appears potentially relevant for solving other kinds of problems. Tahu search evaluations using standard rules were shown on average to produce high scoring (good) moves when improving moves were unavailable, and to produce medium to low scoring moves when only non-improving moves were available. This phenomenon was exploited by using frequency based memory in a simple diversification approach that penalized frequently occurring solution attributes, activating the penalties precisely during the absence of admissible improving moves. Significant solution improvements resulted for the scheduling problems studied.

To date, these strategies have not yet been applied to develop methods for most other types of problems. A useful extension of target analysis in this context would be to subdivide its operation to develop different classes of rules according to whether the goal is intensification or diversification (e.g. focusing on reaching closest elite solutions during intensification, and on jumping from the vicinity of one elite solution to that of another during diversification).

3.1.2. Shifting penalty approach

The shifting penalty tactic for diversifying the search is illustrated by the procedure of Hertz and de Werra [23] and Costa [4] for solving time table planning

problems. The goal of these problems is to find a feasible course schedule subject to numerous constraints belonging to several categories.

The shifting penalty tactic is applied in these instances to guide the search to discover a feasible solution. First, to define the problem objective, a penalty function is created for every constraint based on the degree to which it is violated for any given schedule. The global objective function then is expressed as a weighted sum of the penalty functions. An important constraint is given a higher weight than another of less importance.

At the beginning of the search, the procedure is driven to satisfy the most important constraints because this will provide the greatest improvements in the objective function. After that, the search undertakes to satisfy constraints associated with lower weights. In order to diversify the search, the shifting penalty tactic dramatically decreases the highest weights after a given number of iterations, and then maintains the decreased values for some number of iterations before reinstating the original values. This standardly causes the new solution to have a different structure than exhibited by the solutions visited before the diversification.

Gendreau et al. [11] have created an adaptation of TS for the vehicle routing problem that automates a rule to change the weights associated with each relaxed constraint. At the beginning of the search, every weight is set to 1. Then, after every 10 iterations, a weight associated with a constraint that was always violated during the past 10 iterations is multiplied by 2; a weight associated with a constraint that was never violated during these 10 iterations is divided by 2, and otherwise the weights stay unchanged.

3.1.3. Strategic oscillation and the principle of proximate optimality

The shifting penalty tactic is an instance of a procedure called *strategic oscillation,* which represents one of the basic diversification approaches for tabu search [12]. The idea is to drive the search toward and away from selected boundaries of feasibility (or selected functional values), either by manipulating the objective function (e.g. with penalties and incentives) or simply by compelling the choice of moves that lead in specified directions. The oscillation may cross the boundary and penetrate to selected depths on either side, or may always approach and retreat from the same side. A common implementation is to alternate a series of constructive (or incrementing) moves with a series of destructive (or decrementing) moves. Recent applications of strategic oscillation have proved beneficial in solving graph partitioning problems [36] and course scheduling problems [32].

In certain settings, strategic oscillation also acquires additional power by an implementation linked to a concept called the Proximate Optimality Principle (POP). This principle stipulates that good solutions at one level are likely to be found close to good solutions at an adjacent level. The term "level" can refer to a stage of a

constructive or destructive process (such as incorporating a specified number of nodes, edges, or variables into a partial solution at that stage), or can refer to a given measure of distance from a specified boundary.

The POP notion is exploited in tabu search by remaining at each successive level for a chosen number of iterations, and then incorporating the best solution found (for the conditions defining the level) to initiate a move to the next level. For example, a level may be characterized by compelling values of a function to fall in a given range, or by requiring a given number of elements to be included in a partial construction. Then the process can be controlled to remain at a specified level by employing a neighborhood whose moves maintain the functional values (or numbers of elements) within the specified limits. This type of approach is sufficiently general to be applied in a TS branching algorithm for mixed integer programming problems [16]. An application of the method by Kelly et al. [25] demonstrates its ability to achieve outcomes that are significantly better than found with alternative procedures for a class of difficult confidentiality problems.

Interesting similarities and contrasts exist between the POP concept of "level" and the simulated annealing notion of "temperature". Each refers to a succession of adjacent states. However, temperature is a measure of energy, as reflected in an objective function change, and the SA mechanism for incorporating this measure is to bias acceptance criteria to favor moves that limit "negative change", according to the temperature value. The SA mechanism also does not seek to maintain a construction at a given level with a design for isolating and carrying forward best solutions to an adjacent level. The types of levels encompassed by the POP notion cover a wide range of strategic alternatives (by comparison to temperature, for example), as derived in reference to numbers of variables or constraints, parametric representations of costs or requirements, hierarchies of aggregation or disaggregation, and so forth.

The POP concept may be viewed as a heuristic counterpart of the so called principle of optimality in dynamic programming. However, it does not entail the associated *curse of dimensionality* manifested in the explosion of state variables that normally occurs when dynamic programming is applied to combinatorial problems. Ifthe premise underlying this concept is valid, it suggests that systematic exploration of effective definitions of "levels" and "closeness", and the design of move neighborhoods for exploiting them, may disclose classes of strategies that usefully enlarge the options currently employed.

3.2. TECHNICAL IMPROVEMENfS OF THE SEARCH

From now on, we suppose that the types of moves and the criteria for evaluating them are fixed, that other constituents of TS are to be improved. We examine the issues of appropriate neighborhood sizes, tabu list structures and aspiration conditions.

3.2.1. Neighborhood sizes and candidate lists

A complete neighborhood examination with TS provides generally high quality solutions (see [37,40-42]) but may be very expensive in terms of CPU-time. For this reason, it is often important to apply TS in conjunction with a strategy that isolates regions of the neighborhood containing moves with desirable features, putting these moves on a list of candidates for current examination. A few prominent strategies of this type that have found successful application with TS are as follows.

Neighborhood decomposition strategies. A useful candidate list approach is to decompose the neighborhood into coordinated subsets at each iteration. A TS aspiration threshold, or means of linking the examination of subsets, commonly is applied to limit the frequency of selecting moves from subsets whose current alternatives are gauged less attractive. This strategy generally makes it possible both to speed up the computation time and to generate some diversity in the search. Laguna et al. (28] have used this approach to limit the domains of jobs that are shifted in machine scheduling and Fiechter [6] has applied such a decomposition to limit exchanges in traveling salesman problems. More recently, in a practical vehicle routing application, Semet and Taillard [38] have succeeded in cutting down computation times by a factor of 3 while simultaneously obtaining better solutions, applying such a decomposition approach to cyclically scan about one fourth of all possible moves at each iteration.

Elite evaluation candidate lists. Another technique for scanning a subset of the neighborhood is to store a collection of the most promising, or "elite" (highest evaluation) moves on the candidate list. At a given iteration, the moves belonging to the candidate list are examined first, followed by a subset of the regular neighborhood, gradually replacing candidate list moves that are no longer attractive. Periodically, after a specified number of iterations or when the quality of moves on the candidate list deteriorates below a chosen threshold, a significantly larger portion of the current neighborhood is examined to reconstruct the candidate list. This technique is motivated by the assumption that a good move, if not performed at the present iteration, will still be a good move for some number of iterations. (More precisely, after an iteration is performed, the nature of a recorded move implicitly may be transformed. The assumption is that a useful proportion of these transformed moves will inherit attractive propertied from their antecedents.)

A simplified variant of this strategy is to perform every move on the elite candidate list in succession, provided the move remains valid when its tum arrives. This approach, applied to TSPs in [6], permits a large number of moves to be performed scanning new neighborhoods and rebuilding the list, although at some risk of making less desirable moves. The approach may be improved by introducing an aspiration level threshold that moves must satisfy to be selected (see, e.g. [18]).

Preferred attribute candidate lists. In some applications it can be advantageous to isolate certain attributes of moves that are expected also to be attributes of good

solutions, and to limit consideration to those moves whose composition includes some portion of these "preferred" attributes. For example, in traveling salesman problems it often happens that good solutions are primarily composed of edges that are among the 10 or 20 shortest edges meeting one of their endpoints. Some studies have attempted to limit consideration entirely to tours constructed from such a collection of edges.

A preferred attribute candidate list, by contrast, seeks to organize moves so that they do not have to be composed entirely of such special elements. However, one or more of these elements are required to be incorporated in a "key segment" of a move, so that all moves containing such a segment can be generated very efficiently. This approach has been used by Gendreau et al. [10] and by Johnson [24] in application to TSPs without reference to TS. More recently, Gendreau et al. [11] have made an effective adaptation of this approach in the TS context for VRP problems. Related advances for VRPs are reported in the study of Osman {34].

Sequential fan candidate lists. A type of candidate list that is highly exploitable by parallel processing is a *sequential fan* candidate list. The basic idea is to generate some p best alternative moves at a given step, and then to create a fan of solution streams, one for each alternative. The several best available moves for each stream are again examined, and only the p best moves overall (where many or no moves may be contributed by a given stream) provide the p new streams at the next step.

In the setting of tree search methods such a sequential fanning process is sometimes called *beam search.* A useful refinement called *filtered beam search* has been proposed and studied by Ow and Morton (35] and other refinements (beyond the tree search setting) have been suggested by Glover [15]. A recent implementation for QAP problems that appears highly promising has been carried out by Gavish [9]. For use in the tabu search framework, it is to be noted that TS memory and restrictions can be carried forward with each stream and hence "inherited" in the selected continuations. In this case, a relevant variation is to permit the search of each stream to continue for some number of iterations until reaching a new local optimum. Then a subset of these can be selected and carried forward. Since a chosen solution can be assigned to more than one new stream, different streams can embody different missions in TS, as by giving different emphasis to intensification and diversification. It may be noted that the type of staging involved in successive solution runs of each stream may be viewed as a means of defining levels in the context of the Proximate Optimality Principle, and hence this way of implementing a parallel solution method may also give another approach for exploiting the POP notion.

3.2.2. Tabu list types

The choice of appropriate types of tabu lists depends on the problem. Although no single type of list is uniformly best for all applications, some guidelines are

possible. If the size of the neighborhood is small enough to store an item of infonnation for each move attribute used to define a tabu restriction, it is generally worthwhile to store the iteration number that identifies when the tabu restriction associated with such an attribute may be discarded. This makes it possible to test the tabu status of a move in constant time (by reference to whether the move embodies a "tabu attribute"), and the necessary memory space depends on the neighborhopd size and not on the "tabu list size" (i.e. the number of elements that might otherwise be recorded, one attribute group for each iteration, to cover the number of iterations that detennine tabu status).

In a general way, it appears that tabu lists designed to insure the elimination of cycles of length proportional to the tabu list size provide very good results. (For reasons not mathematically identified, this "worst case" insurance translates into an empirical phenomenon where repeated cycles disappear entirely once the tabu list size achieves a critical length.) In the tabu list types used for very large problems (or complex attribute definitions), as in the TSP implementation of [6], it may not be possible to store an item of information for each (reverse) move. In this case attributes of moves may be stored in sequential tabu lists. In the indicated TSP study, every edge added to the current solution by a move was incorporated in a list T_{in} and every deleted edge was incorporated in a list T_{out}. A move was defined tabu if both the incoming edges belonged to T_{out} and the outgoing edges belonged to T_{in}. In another study using a different type of candidate list strategy [26], it appeared preferable to store only the shorter of the two added edges on T_{in} and the longer of the two deleted edges on *Tom*. In both studies $|T_{out}|$ was set to a larger value than $|T_{in}|$, based on the fact that a TSP tour has *O(N)* edges and the total number of edges is in *O(N2).*

We now discuss the value to be given to the parameter *t* that embodies the tabu list size.

3.2.3. Tabu list size

Empirically, tabu list sizes that provide good results often grow with the size of the problem. However, no single rule (even an empirical one) gives good sizes for all classes of problems. This is partly because an appropriate list size depends on the strength of the tabu restrictions employed (where stronger restrictions are generally coupled with smaller sizes). The way to identify a good tabu size for a given problem class and choice of tabu restrictions is simply to watch for the occurrence of cycling when the size is too small and the deterioration in solution quality when the size is too large (caused by forbidding too many moves). Best sizes lie in an intennediate range between these extremes. In fact, the best approach is to allow the size in this intermediate range to vary.

These considerations may be clarified by our illustrated adaptation of TS for the QAP: in fig. 3 *we* plot the value of the best solution found and the mean cost

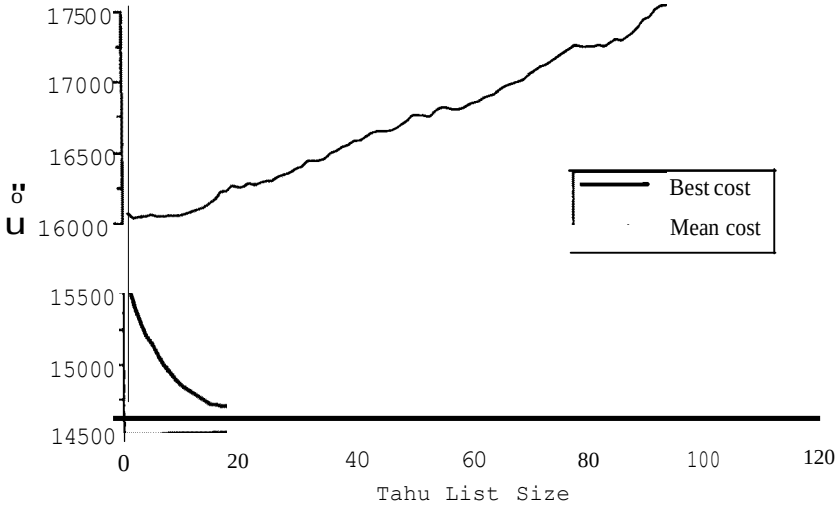

Fig. 3. Influence of tabu list size.

of the visited solution during a search with a given number of iterations. These values are plotted as functions of the tabu list size. We see that both curves decrease for small sizes and then increase as the size grows. However, the minima are reached for a smaller size for the mean cost than for the best cost. This translates into the fact that a small tabu list size is preferable for expioring the solution near a local optimum and a larger tabu list size is preferable for breaking free of the vicinity of this local minimum.

Varying the tabu list size during the search provides one way to take advantage of this effect. For the QAP, the approach of Taillard [42] selects this size randomly from an interval ranging from $t_{min} = L0.9NJ$ to the value $t_{max} = rl.tNl$ (where N is the dimension of the problem). The chosen size is maintained constant for $2t_{max}$ iterations, and then a new size is selected by the same process.

Other simple types of dynamic tabu list approaches include systematically varying the list sizes over three different ranges (small, medium and large), as applied to telecommunications bandwidth packing problems by Laguna and Glover [29], and introducing *"moving tabu gaps"* as in the QAP approaches of Skorin-Kapov [39] and Chakrapani and Skorin-Kapov [2]. In each of these cases, the dynamic tabu list approach proved superior to using a static list of fixed size. Further improvements were obtained in these studies by incorporating a longer term frequency based memory as well as the short term recency based memory of the tabu list. We sketch how these improvements were obtained subsequently.

Finally, we note the relevance of two additional forms of dynamic tabu list strategies, one based on uses of *hashing functions,* as suggested by Hansen and Jaumard [21] and as explored in detail by Woodruff and Zemel [46], and the other

based on *sequential logic,* as proposed by Glover [16] and as studied in comparative implementations by Dammeyer and Voss [5].

3.2.4. Aspiration criteria

In most applications, an aspiration criterion takes the form: a tabu m is available (or compelled) to be selected if it can reach a solution $s_k \oplus m$ better than the best solution S_k^* obtained up to iteration k, i.e. if

$$f(s_k^*) > f(s_k \oplus m). \tag{3.1}$$

It can also be useful to apply aspiration criteria differently at different phases of search. Such an approach is particularly motivated by findings involving the use of frequency based long term memory in solving machine scheduling problems [29], alluded to earlier in the discussion of target analysis. In this case the memory records the number of times move attributes (or solution attributes) occur over the search history – specifically, the number of times a job was moved to occupy a particular position for processing. The frequency counts are then weighted to penalize moves whose attributes have higher associated frequencies, applying the findings of target analysis by activating the penalties only after the search began to slow its rate of producing improved solutions, and then only in situations where no admissible (non-tabu) improving moves existed.

A surprising aspect of this implementation was that the best tabu list size did not appear to grow with problem size, and that a variable list size also became less important. Overall, the solutions obtained were considerably superior to those that did not incorporate the longer term memory activated in restricted non-improving phases.

The use of such an approach that penalizes frequently performed moves can be implemented as follows. First, one counts the number of times each move m is performed, in order to compute its frequency fm. Then, a penalty pm is associated with each move:

$$pm = \begin{cases} 0 & \text{if } m \text{ meets an aspiration criterion;} \\ w \cdot fm & \text{otherwise,} \end{cases} \tag{3.2}$$

where w is a constant. Then the value of a move is:

$$f(x + m) - f(x) + pm. \tag{3.3}$$

The weight w depends on the problem, on the move type and on the neighborhood. However, in several applications (VRP, QAP, electrical network design) we observed that this weight is approximately proportional to the square root of the size of the neighborhood multiplied by the standard deviation of the value (without penalty)

of every move tried during the search. In addition, such a frequency based memory is almost unaffected by the tabu list size; this means that a good weight w may be found by optimizing this parameter independently.

The asymmetry between improving and nonimproving search phases underlying the exploitation of such penalties suggests the merit of aspiration criteria that permit tabu status to be disregarded during improving phases, provided this status was created by a move which was also an improving move (or by an improving move of the current improving phase). Otherwise, tabu status is implemented in the usual manner during the improvement phase, subject to a secondary aspiration criterion. This criterion allows tabu moves to be selected if they are the only improving moves available (once an improving phase is in progress). The move then is chosen by selecting a "least tabu" or "most improving" tabu move, or by a varying mix of these criteria. Once a local optimum is reached, such an approach then shifts to a more rigid "better than best" aspiration criterion (e.g. accompanied by frequency based memory as previously indicated), and this criterion is maintained until a new improving phase is launched.

The application of such a strategy, and of related differential phase strategies that combine recency based and frequency based memory in more sophisticated ways, provide a means of exploiting aspiration levels with a considerable degree of adaptiveness, and constitute another area warranting fuller investigation.

3.3. COMPUTATIONAL IMPROVEMENTS

We now focus on the issue of speeding up the execution of each iteration of the search. A fundamental tactic, when it can be executed, is to replace the double evaluation of the objective function $/(sJ$ and $f(sk$ E9 $m)$, needed to determine the attractiveness of move m applied to solution sk, by the evaluation of the function $8(sk,m)$ defined as the simplified algebraic expression $f(sk$ ED $m) - f(sk)$· If the size of the neighborhood is small enough to store $8(sk> m)$ for every move m, then it is often advantageous to express $8(skt m)$ as a function of $A(sk - hm)$ and the move $mk-1$ performed at the previous iteration. In other words, information computed at one step may help to accelerate the computation required at the next step. Computational simplification of the types have provided significant speed-ups for a number of applications, including the machine scheduling study of Laguna et al. (28] and the QAP study of Taillard [42] previously discussed. In the QAP study the refined objective function evaluation cut down the computation of the neighborhood from $O(N^4)$ to $O(N^3)$ and the stored evaluations further reduced the computations to $O(N^2)$

Sometimes such simplifications are not feasible, and other approaches must be sought to speed up the search. The use of a profiler-a program that analyzes the time spent in the procedures and lines of an application-can be helpful in this regard.

In a practical vehicle routing problem [38], the search was accelerated by a factor of 1000 by isolating the most expensive routine with such an analysis, and

then significantly reducing the number of times the routine was invoked (by a combined strategy of algebraic simplification, partial neighborhood examination and a relaxed objective function computation). A similar approach has also proved useful in accelerating the solution of scheduling problems in [40].

3.3.1. Parallel processing

Parallelization methods provide the ultimate means of speeding up the search. At equivalent costs, distributed computers are potentially more powerful than sequential ones. Concurrent examination of different moves from the neighborhood often makes it possible to reach a speed-up factor close to the ideal one

$$\text{ideal speed-up} = \frac{T_n + T_u}{T_n/p + T_u}, \tag{3.4}$$

where p is the number of processors, Tn is the sequential time to perform the computation to be parallelized and T_u is the time associated with the non-parallelizable part of the algorithm. For example, Tn may correspond to the time spent to evaluate the neighborhood and T_u may correspond to the update of the information structure when a move is performed.

Using such a technique, significant speed-up can be achieved for many problems. Assuming that the problem and the type of move are adapted to perform many moves in parallel, such a technique may be useful where a great quantity of moves must be performed and where obtaining the global minimum is not a prime necessity. This may be illustrated in our traveling salesman example with the type of diversification move described previously. As noted, after such a move is executed, a series of 2-opt moves is performed to reach a new local optimum. The structure of the problem suggests that appropriate tour modifications will probably occur on portions of the tour that are near the edges modified by the new move (see fig. 4). If the end points of the four paths to be reoptimized are fixed, every path may be optimized simultaneously.

It has been found advantageous to perform several diversifying moves successively using an elite evaluation candidate list. As a result, there are many places where the tour has been optimized again. This fact is exploited in [6] by cutting the tour into subpaths having about the same number of vertices-approximately 50-and optimizing every subpath in parallel. Then, another way of cutting the tour is selected, repeating until improvements become negligible.

Unfortunately, the assumptions needed to be able to perform parallel moves based on such an implicit decomposition are extremely strong and are met only fo.r a few problems. A quite natural and less restricted parallelization process, that works for every problem where we have undertaken to apply it, is to perform many independent searches at a time, each starting with a different initial solution or/and using a different set of parameters.

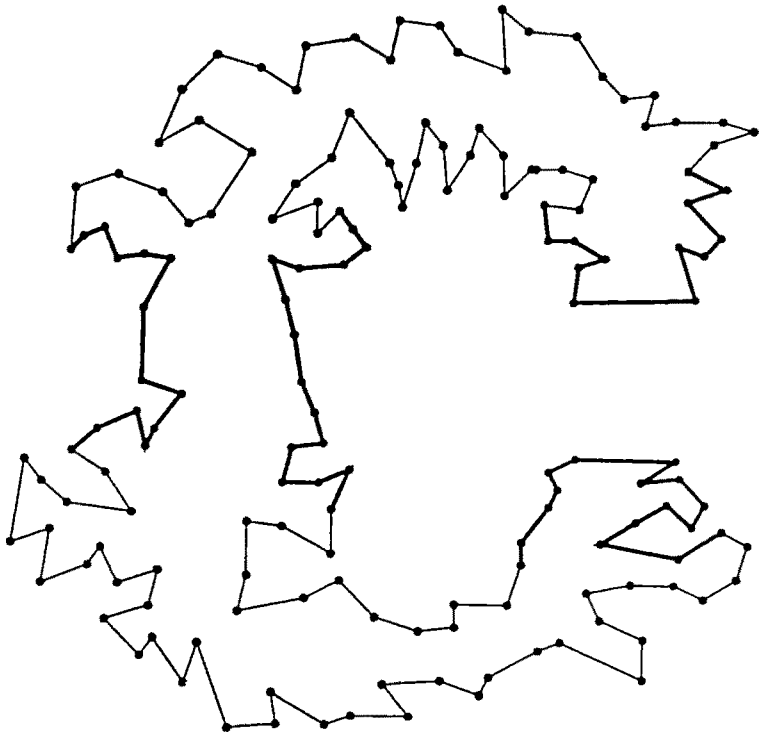

Fig. 4. Parts of the tour that probably have to be reoptimized after a new diversifying move.

Surprisingly, this straightforward type of parallelization is very efficient for a number of processors not exceeding a few dozen. It is also easy to implement, and may be efficient even when simulated on a sequential computer (as for example where the parameters and structuring of the method are not optimally tuned). Results from applying this type of parallel implementation may be found in [40-42]. We also recall that the type of parallel processing approach discussed in connection with sequential fan candidate lists merits fuller consideration.

4. Concluding remarks

We have undertaken to present the main features of tabu search and to sketch some of the improvements that can be obtained by various refinements. Experiments have shown that TS is able to obtain results that match or surpass the best known outcomes in a variety of optimization settings. Nevertheless, our understanding of TS is not complete. New implementations continue to teach us new lessons and to suggest new refinements.

At the same time, mathematical analysis to explain (and ultimately enhance) the performance of tabu search is an area open for examination. Evidently, such an

analysis will embody elements of artificial intelligence (concerning integrated uses of memory) and likely will call upon concepts of probabilistic reasoning. In addition, graph theoretic arguments may prove relevant, based on representing solutions as nodes and moves as arcs-with the goal of determining why trajectories guided by certain forms of memory (in solution spaces with particular structures) turn out to be more effective than others. Finally, connections between TS and heuristic elaborations of dynamic programming (as embodied, for example in the POP notion and strategic oscillation) may provide a fruitful avenue for exploration. The present gap between empirical efficiency and mathematical demonstration invites an effort to bring these realms into closer harmony, with the possibility of creating a foundation for the "next generation" of tabu search.

References

[1] R.E. Burkard, Quadratic assignment problems, Eur. J. Oper. Res. 15(1984)283-289.
[2] J. Chak:rapani and J. Skorin-Kapov, Massively parallel tabu search for quadratic assigment problem, Ann. Oper. Res. (1993), this volume.
[3] Committee on the Next Decade of Operations Research (Condor), Operations research: The next decade, Oper. Res. 36(1988).
[4] D. Costa, A tabu search algorithm for computing an operational time table, Working Paper, Departement de Mathematiques, Ecole Polytechnique Federale de Lausanne, Switzerland (1990).
(5) F. Dammeyer and S. Voss, Dynamic tabu list management using the reverse elimination method, Ann. Oper Res. (1993), this volume.
(6) C.N. Fiechter, A parallel tabu search algorithm for large scale traveling salesman problems, Working Paper 90/l, Departement de Mathematiques, Ecole Polythechnique Federale de Lausanne, Switzerland (1990).
(7) G. Finke, R.E. Burkard and F. Rendl, Quadratic assignment problems, Ann. Discr. Math. 31(1987)61-82.
(8) A. Friei.e, J. Yadegas, S. El-Horbaty and D. Parkinson, Algorithms for assignment problems on an array processor, Parallel Comp. 11(1989)151-162.
(9) B. Gavish, Manifold search techniques applied to the quadratic asssignrnent problem, Technical Report, Owen Graduate School of Management, Vanderbilt University (1991).
(10) M. Gendreau, A. Hertz and G. Laporte, New intersection and post-optimization procedures for the traveling salesman problem, CRT-708, Centre de Recherche sur les Transports, Universite de Montreal (1990).
(11) M. Gendreau, A. Hertz and G. Laporte, A tabu search heuristics for the vehicle routing prol:ilem, CRT-777, Centre de Recherche sur les Transports, Universite de Montreal (1991) to appear in Manag. Sci.
(12) F.Glover, Heuristics for integer programming using surrogate constraints, Dec. Sci. 8(1977)156-166.
[13] F. Glover, Future paths for integer programming and links to artificial intelligence, Comp. Oper. Res. 13(1986)533-549.
[14] F. Glover, Tabu search-Part I, ORSA J. Comput. 1(1989)190-206.
[15] F. Glover, Candidate list strategies and tahu search, CAAI Research Report, University of Colorado, Boulder (July 1989).
(16) F. Glover, Tahu search-Part Il, ORSA J. Comput. 2(1990)4-32.
(17] F. Glover, Tahu search for nonlinear and parametric optimization (with links to genetic algorithms). Technical Report, Graduate School of Business and Administration. University of Colorado at Boulder (1991), to appear in Discr. Appl. Math.
[18] F. Glover, R. Glover and D. Klingman, The threshold asssignment algorithm, Math. Progr. Study 26(1986)12-37.

[19] F. Glover, D. Klingman and N. Phillips, A network related nuclear power plant model with an intelligent branch and bound solution approach, Ann. Oper. Res. 21(1990)317-332.
[20] P. Hansen, The steepest ascent mildest descent heuristic for combinatorial programming, *Congress on Numerical Methods in Combinatorial Optimization,* Capri, Italy (1986).
[21] P. Hansen and B. Jaumard, Algorithms for the maximum satisfiability problem, Computing 44(1990)279-303.
[22] A. Hertz and D. de Werra, Using tabu search techniques for graph coloring, Computing 39(1987)345-451.
[23] A. Hertz and D. de Werra, Informatique et Horaires Scolaires, Output 12(1989)53-56.
[24] D. Johnson, Local optimization and the traveling salesman problem, *Proc. 17th Annual Colloquim on Automata, l.Anguages and Programming* (Springer, 1990) pp. 446-461.
[25] J.P. Kelly, B.L. Golden and A.A. Assad, Large-scale controlled rounding using tabu search with strategic oscillation, Ann. Oper. Res. (1993), this volume.
[26] J. Knox, The application of tabu search to the symmetric traveling salesman problem, Ph.D. thesis, Graduate School of Business, University of Colorado.
[27] A. Kohlen and D. Pesch, Genetic local search in combinatorial optimization, to appear in Discr. Appl. Math. (1991).
[28) M. Laguna, J.W. Barnes and F. Glover, Tahu search methods for a single machine scheduling problem, J. Int. Manufacturing 2(1991)63- 74.
[29] M. Laguna and F. Glover, Integrating target analysis and tabu search for improved scheduling systems, School of Business, University of Colorado, Boulder (1991), to appear in Expert Systems with Application: An International Journal.
[30] S. Lin and B.W. Kernighan, An effective heuristic algorithm for the traveling salesman problem, Oper. Res. 21(1973)498-516.
[31] M. Malek, M. Guruswamy, H. Owens and M. Pandya, Serial and parallel search techniques for the traveling salesman problem, Ann. Oper. Res. (1989).
[32[E.L. Mooney and R.L. Rardin, Tahu search for a class of scheduling problems, Ann. Oper. Res. (1993), this volume.
[33] H. Muhlenbein, Parallel genetic algorithms and combinatorial optimization, to appear in SIAM J. Optim. (1991).
[34] l.H. Osman, Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem, Ann. Oper. Res. (1993), this volume.
[35] P.S. Ow and T.E. Morton, Filtered beam search in scheduling, Int. J. Prod. Res. 26(1988)35-62.
(36] E. Rolland and H. Pirkul, Heuristic search for graph partitioning, *31st Joint National TIMS/ORSA Meeting,* Nashville, TN (1991).
(37] F. Semet and I. Loewenton, The traveling salesman problem under accessibility constraints, Report ORWP 92/02, DMA, EPFL (1992).
[38] F. Semet and E. Taillard. Solving real-life VRPS efficiently using TS, Ann. Oper. Res. (1993), this volume.
[39] J. Skorin-Kapov, Extensions of a tabu search adaptation to the quadratic assigrunent problem, Harriman School Working Paper HAR-90-006, State University of New York at Stony Brook (1990).
[40] E. Taillard, Parallel tabu search for the jobshop scheduling problem, Research Report ORWP 89/ 11, EPFL, DMA Lausanne, Switzerland (1989).
(41] E. Taillard, Some efficient heuristic methods for the flowshop sequencing problem, Euro. J. Oper. Res. 47(1990)65- 79.
(42) E. Taillard, Robust taboo search for the quadratic assignment problem, Parallel Comp. 17(1991)443-455.
[43] N. Ulder, E, Aarts, H.-J. Bandelt, P. van Laarhoven and E. Pesch, Genetic local search algorithms for the traveling salesman problem, *Proc.]st Int. Workshop on Parallel Problem Solving,* ed. Schwefel and Manner, Lecture Notes in Computer Science 496(1991) pp. 109-116.
(44] D. de Werra and A. Hertz, Tahu search tehcniques: A tutorial and an application to neural networks, OR Spectrum (1989)131-141.
(45] D. Whitley, T. Starkweather and D. Fuguay, Scheduling problems and traveling salesman: The genetic edge recombinition operator, *Proc. 3rd Int. Conj. of Genetic Algorithms,* Fairfax, VA (1989).
[46] D.L. Woodruff and E. Zemel, Hashing vectors for tabu search, Ann. Oper. Res. (1993), this volume.