TABU SEARCH FUNDAMENTALS AND USES

by

Fred Glover

US West Chair in Systems Science
Graduate School of Business, Box 419
University of Colorado
Boulder, Colorado 80309-0419

E-mail: fred.glover@colorado.edu

REVISED and EXPANDED: June 1995

Abstract

Tabu search has achieved widespread successes in solving practical optimization problems. Applications are rapidly growing in areas such as resource management, process design, logistics, technology planning, and general combinatorial optimization. Hybrids with other procedures, both heuristic and algorithmic, have also produced productive results. We examine some of the principal features of tabu search that are most responsible for its successes, and that offer a basis for improved solution methods in the future.

Note: This expanded version contains additional illustrations and information on candidate list strategies, probabilistic tabu search, strategic oscillation and parallel processing options. Sections have also been added on principles of intelligent search.

Acknowledgement: This work has been supported in part by the National Science and Engineering Council of Canada under Grants 5-83998 and 5-84181.

Background

Tabu Search (TS) is a *metaheuristic* that guides a local heuristic search procedure to explore the solution space beyond local optimality. Widespread successes in practical applications of optimization have spurred a rapid growth of tabu search in the past few years. TS procedures that incorporate basic elements describe in this paper, and hybrids of these procedures with other heuristic and algorithmic methods, have succeeded in finding improved solutions to problems in scheduling, sequencing, resource allocation, investment planning, telecommunications and many other areas. Some of the diversity of tabu search applications is shown in Table 1. (See also the survey of Glover and Laguna (1993), and the volume edited by Glover, Laguna, Taillard and de Werra (1993).)

Tabu search is based on the premise that problem solving, in order to qualify as intelligent, must incorporate *adaptive memory* and *responsive exploration*. The use of adaptive memory contrasts with "memoryless" designs, such as those inspired by metaphors of physics and biology, and with "rigid memory" designs, such as those exemplified by branch and bound and its AI-related cousins. The emphasis on responsive exploration (and hence purpose) in tabu search, whether in a deterministic or probabilistic implementation, derives from the supposition that a bad strategic choice can yield more information than a good random choice. (In a system that uses memory, a bad choice based on strategy can provide useful clues about how the strategy may profitably be changed. Even in a space with significant randomness which fortunately is not pervasive enough to extinguish all remnants of order in most real world problems a purposeful design can be more adept at uncovering the imprint of structure, and thereby at affording a chance to exploit the conditions where randomness is not all-encompassing.)

ILLUSTRATIVE TABU SEARCH APPLICATIONS

Scheduling
- Flow-Time Cell Manufacturing
- Heterogeneous Processor Scheduling
- Workforce Planning
- Classroom Scheduling
- Machine Scheduling
- Flow Shop Scheduling
- Job Shop Scheduling
- Sequencing and Batching

Design
- Computer-Aided Design
- Fault Tolerant Networks
- Transport Network Design
- Architectural Space Planning
- Diagram Coherency
- Fixed Charge Network Design
- Irregular Cutting Problems
- Lay-Out Planning

Location and Allocation
- Multicommodity Location/Allocation
- Quadratic Assignment
- Quadratic Semi-Assignment
- Multilevel Generalized Assignment

Logic and Artificial Intelligence
- Maximum Satisfiability
- Probabilistic Logic
- Clustering
- Pattern Recognition/Classification
- Data Integrity
- Neural Network Training
- Neural Network Design

Technology
- Seismic Inversion
- Electrical Power Distribution
- Engineering Structural Design
- Minimum Volume Ellipsoids
- Space Station Construction
- Circuit Cell Placement
- Off-Shore Oil Exploration

Telecommunications
- Call Routing
- Bandwidth Packing
- Hub Facility Location
- Path Assignment
- Network Design for Services
- Customer Discount Planning
- Failure Immune Architecture
- Synchronous Optical Networks

Production, Inventory and Investment
- Flexible Manufacturing
- Just-in-Time Production
- Capacitated MRP
- Part Selection
- Multi-item Inventory Planning
- Volume Discount Acquisition
- Fixed Mix Investment

Routing
- Vehicle Routing
- Capacitated Routing
- Time Window Routing
- Multi-Mode Routing
- Mixed Fleet Routing
- Traveling Salesman
- Traveling Purchaser
- Convoy Scheduling

Graph Optimization
- Graph Partitioning
- Graph Coloring
- Clique Partitioning
- Maximum Clique Problems
- Maximum Planner Graphs
- P-Median Problems

General Combinational Optimization
- Zero-One Programming
- Fixed Charge Optimization
- Nonconvex Nonlinear Programming
- All-or-None Networks
- Bilevel Programming
- General Mixed Integer Optimization

TABLE 1

These basic elements of tabu search have several important features, summarized in Table 2.

PRINCIPAL TABU SEARCH FEATURES

Adaptive Memory

Selectivity (including strategic forgetting)

Abstraction and decomposition (through explicit and attributive memory)

Timing:
- recency of events
- frequency of events
- differentiation between short term and long term

Quality and impact:
- relative attractiveness of alternative choices
- magnitude of changes in structure or constraining relationships

Context:
- regional interdependence
- structural interdependence
- sequential interdependence

Responsive Exploration

Strategically imposed restraints and inducements
(*tabu conditions* and *aspiration levels*)

Concentrated focus on good regions and good solution features
(*intensification processes*)

Characterizing and exploring promising new regions
(*diversification processes*)

Non-montonic search patterns
(*strategic oscillation*)

Integrating and extending solutions
(*path relinking*)

TABLE 2

Tabu search is concerned with finding new and more effective ways of taking advantage of the concepts embodied in Table 2, and with identifying associated principles that can expand the foundations of intelligent search. As this occurs, new strategic mixes of the basic ideas emerge, leading to improved solutions and better practical implementations. This makes TS a fertile area for research and empirical study.

The remainder of this paper is divided into three main parts. Section 1 and its subsections are devoted to presenting the main concepts and strategies of tabu search, and to showing how they interrelate. Section 2 focuses on specific aspects of implementation, with illustrations of useful ways to organize memory processes to enhance the efficiency of the search. Section 3 discusses special considerations for advanced solution capabilities. Implications for future developments are discussed in the concluding section.

1. *Tabu Search Foundations*

The basis for tabu search may be described as follows. Given a function $f(x)$ to be optimized over a set X, TS begins in the same way as ordinary local search, proceeding iteratively from one point (solution) to another until a chosen termination criterion is satisfied. Each $x \in X$ has an associated *neighborhood* $N(x) \subset X$, and each solution $x \quad N(x)$ is reached from x by an operation called a *move*.

TS goes beyond local search by employing a strategy of modifying N(x) as the search progresses, effectively replacing it by another neighborhood N*(x). As our previous discussion intimates, a key aspect of tabu search is the use of special memory structures which serve to determine $N^*(x)$, and hence to organize the way in which the space is explored.

The solutions admitted to N*(x) by these memory structures are determined in several ways. One of these, which gives tabu search its name, identifies solutions encountered over a specified horizon (and implicitly, additional related solutions), and forbids them to belong to N*(x) by classifying them *tabu*. (The tabu terminology is intended to convey a type of restraint that embodies a "cultural"

connotation i.e., one that is subject to the influence of history and context, and capable of being surmounted when conditions warrant.)

The process by which solutions acquire a tabu status has several facets, designed to promote a judiciously aggressive examination of new points. A useful way of viewing and implementing this process is to conceive of replacing original evaluations of solutions by *tabu evaluations*, which introduce penalties to significantly discourage the choice of tabu solutions (i.e., those preferably to be excluded from N*(x), according to their dependence on the elements that compose tabu status). In addition, tabu evaluations also periodically include inducements to encourage the choice of other types of solutions, as a result of aspiration levels and longer term influences.

It should be emphasized that the concept of a neighborhood in tabu search also differs from that used in local search, by embracing the types of moves used in constructive and destructive processes (where the foundations for such moves are accordingly called *constructive neighborhoods* and *destructive neighborhoods*). Such expanded uses of the neighborhood concept reinforce a fundamental perspective of TS, which is to define neighborhoods in dynamic ways that can include serial or simultaneous consideration of multiple types of moves, by mechanisms subsequently identified. We begin by sketching in a general way how tabu search takes advantage of memory (and hence learning processes) to modify the neighborhood structures it works with, and to guide its trajectory through these structures. With this foundation, we then give a more detailed view of the primary TS components.

Explicit and Attributive Memory: The memory used in TS is both explicit and attributive. Explicit memory records complete solutions, typically consisting of elite solutions visited during the search (or highly attractive but unexplored neighbors of such solutions). These special solutions are introduced at strategic intervals to enlarge N*(x), and thereby provide useful options not in N(x).

Attributive memory, by contrast, records information about solution attributes that change in moving from one solution to another. For example, in a graph or network setting, attributes can consist of nodes or arcs that are added, dropped or repositioned by the moves executed. In more abstract problem formulations, attributes may correspond to values of variables or functions. Properly used, attributive memory makes it possible to exert a variety of subtle influences. Sometimes attributes are also strategically combined in TS to create other attributes to be used in such memory, as by hashing procedures or by AI related chunking or "vocabulary building" methods. (Such approaches are discussed in Hansen and Jaumard (1990), Woodruff and Zemel (1992), Battiti and Tecchioli (1992a), Woodruff (1993), Glover and Laguna (1993).)

Because tabu search has several critical components, and the task of integrating them may seem at first to involve a fair amount of effort, a number of implementations have been based only on the first ideas typically developed in a general exposition. However, it is to be stressed that the critical components number only a handful (each with a few key variations), and once digested create an interconnected framework that is considerably more effective than focusing only on one or two of the pieces in isolation.

1.1 Short Term Memory and its Accompaniments

An important distinction in TS arises by differentiating between short term memory and longer term memory. Each type of memory is accompanied by its own special strategies. The most commonly used short term memory keeps track of solution attributes that have changed during the recent past, and is called *recency-based* memory. Recency-based memory is exploited by assigning a *tabu-active* designation to selected attributes that occur in solutions recently visited. Solutions that contain tabu-active elements, or particular combinations of these attributes, are those that become tabu. This prevents certain solutions from the recent past from belonging to N*(x) and hence from being revisited. Other solutions that share such tabu-active attributes are also similarly prevented from being revisited. The use of tabu evaluations, with large penalties assigned to appropriate sets of tabu-active attributes, has the effect of allowing tabu status to vary by degrees.

Managing Recency-Based Memory: The process is managed by creating one or several tabu lists, which record the tabu-active attributes and implicitly or explicitly identify their current status. The duration that an attribute remains tabu-active (measured in numbers of iterations) is called its *tabu tenure*. Tabu tenure can vary for different types or combinations of attributes, and can also vary over different intervals of time or stages of search. This varying tenure makes it possible to create different kinds of trade offs between short term and longer term strategies. It also provides a dynamic and robust form of search. (See, e.g., Taillard (1991), Dell'Amico and Trubian (1993), Glover and Laguna (1993).)

An illustration of how recency-based memory operates is provided in Diagram 1. The problem of this illustration is to find an optimal tree (a subgraph without cycles) on the graph with nodes numbered 1 to 7, as shown in the diagram. (For the case where the objective

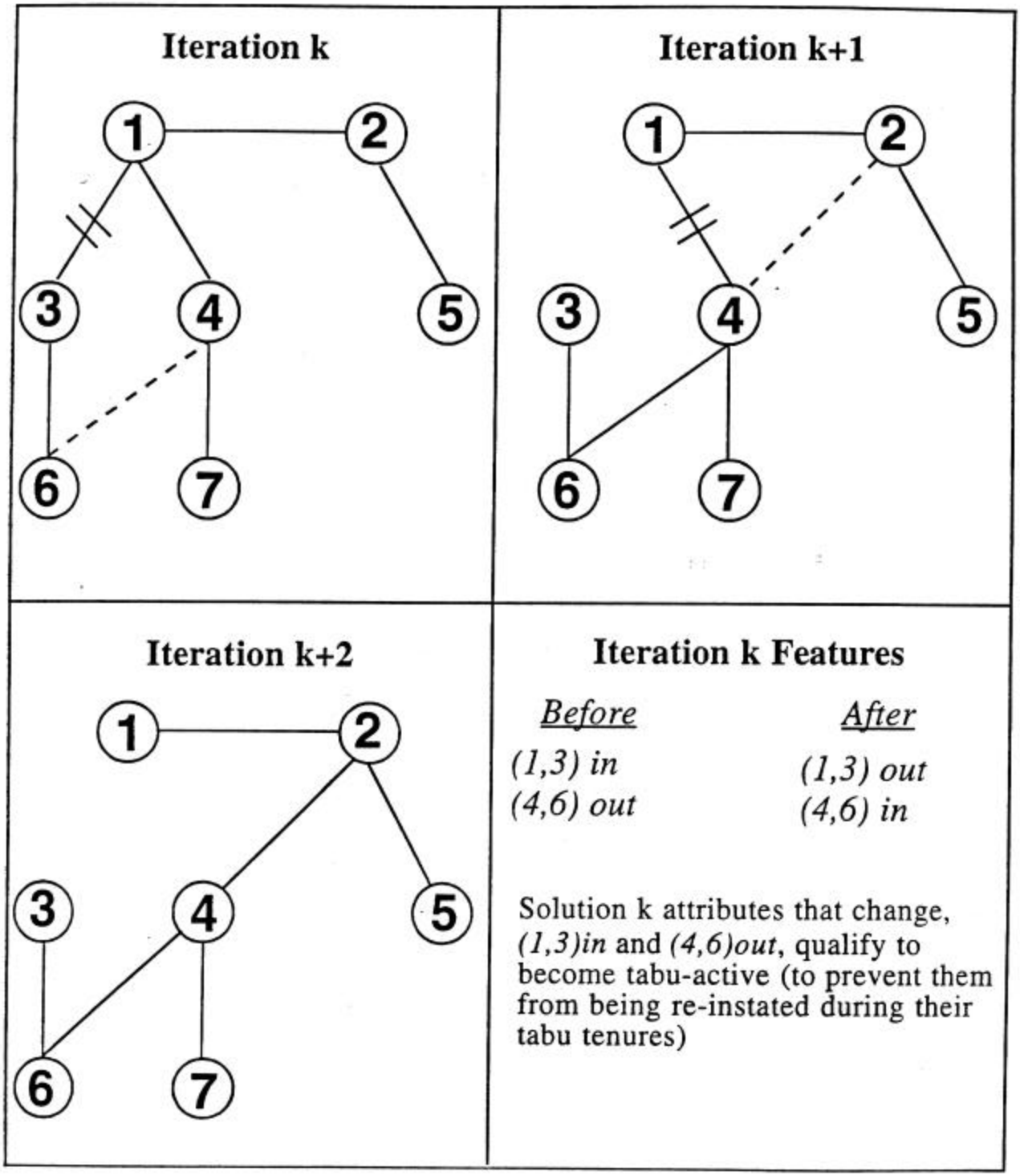


Diagram 1

Example: Optimal Tree Problem with Nonlinear Objective

function is linear the problem is very easy, and so we suppose a more complex nonlinear objective applies, as in electrical power distribution and telecommunication network design problems.) All possible edges that join pairs of nodes will be assumed available for composing the tree, and the three subgraphs illustrated for iterations k, k + 1, and k + 2 (where k is arbitrary) identify particular trees generated at different stages of solving the problem. We suppose that the moves used for changing one tree into another (hence that define the neighborhood N(x), where the solution x corresponds to a particular tree) consist of selecting an edge to be dropped and another to be added, so that the result remains a tree. (The edge dropped must lie on the unique cycle produced by introducing the edge added, or equivalently, the edge added must join the two separate subtrees created by removing the edge dropped.)

The move applied at iteration k to produce the tree of iteration k + 1 consists of dropping the edge (1,3), and adding the edge (4,6), as shown respectively by the edge marked with two crossed lines and the edge that is dotted. The presence of the edge (1,3) and the absence of the edge (4,6) in the tree of the iteration k may be considered as two different solution attributes, which we denote by *(1,3)in* and *(4,6)out*, as indicated in the last box of Diagram 1. Since these are attributes that change as a result of the move, they qualify to be designated tabu-active, and to be used to define the tabu status of moves at future iterations. Assume for the moment we will classify a move to be tabu if any of its attributes is tabu-active. For example, we can specify that *(1,3)in* should be tabu-active for 3 iterations, seeking to prevent edge (1,3) from being added back to the current tree for this duration, and *(4,6)out* should be tabu-active for 1 iteration, seeking to prevent edge (4,6) from being removed from the current tree for this duration. (These conditions effectively seek to avoid "reversing" particular changes created by the move.)

The indicated *tabu tenures* of 3 and 1 of course are very small, and we later discuss how such tabu tenures may be chosen appropriately. However, the rationale for giving a larger tenure to *(1,3)in*

than to *(4,6)out* is important. Specifically, in our illustration, many edges exist that can be added to the tree as part of a move to create a new tree, but somewhat fewer edges exist that can be dropped as part of such a move (since all non-tree edges are available to be added but only the tree edges are available to be dropped). Thus, making *(1,3)in* tabu-active, which prevents edge (1,3) from being added, is much less restrictive than making *(4,6)out* tabu-active, which prevents edge (4,6) from being dropped. (Stated differently, preventing an edge from being added excludes a smaller number of moves than preventing an edge from being dropped.) In general, then, the tabu tenure of an attribute should depend on the restrictiveness of the associated tabu condition.

The terminology used in this example can be relaxed to simply refer to the edges (1,3) and (4,6) as attributes of the move, since the condition of being *in* or *out* is always automatically known from the current solution. Thus, we can simply say that these two edges are tabu-active (with different tenures). If a move is considered that adds edge (1,3) to the tree, it is only necessary to check whether this edge is tabu-active (without keeping separate memory according to whether the edge is present or absent from the tree).

More complex attributes can be used to determine the tabu status of moves, but a warning is necessary. Such attributes must be treated as properties of solutions rather than properties of moves, if cycling is to be avoided during the period that the attributes are tabu-active. (Cycling is normally prevented for much longer durations, typically being eliminated altogether, for reasonable choices of tabu tenures. But the avoidance of cycling is not the sole purpose of recency-based memory.)

We again illustrate by reference to Diagram 1. Suppose at iteration k we create a tabu restriction stipulating that a move which combines the two attributes of adding (1,3) and dropping (4,6) is tabu. That is, we prevent the reversal of the move applied at iteration k. Then, at iteration k + 1, with this condition tabu-active, we could select the move of adding edge (2,4) and dropping edge (1,4), as shown in Diagram 1, to produce the tree of iteration k + 2. Similarly, we stipulate that the reverse

move, which combines the attributes of adding (1,4) and dropping (2,4), is tabu. Now, starting from the tree of iteration k + 2, we can perform two additional moves in succession, the first consisting of adding (1,3) and dropping (2,4), and the second consisting of adding (1,4) and dropping (4,6). Although neither of these moves violates the tabu restrictions, the outcome is to produce the original tree at iteration k.

It is therefore important to note the distinction between making *move attributes* tabu-active (as in the immediately preceding example) and making *solution attributes* tabu-active. Thus, for example, we could have stipulated instead at iteration k that the compound attribute of *(1,3)in* and *(4,6)out* will render a move tabu if the move creates a solution with this attribute. This will avoid the cycling phenomenon just indicated, but checking for such compound attributes within solutions usually requires more memory and effort than checking for simple attributes.

A simpler and more effective alternative is to express tabu restrictions in terms of conjunctions of simple attributes. Thus, for example, we could stipulate a move to be tabu only if all (or some number) of its component solution attributes are tabu-active. This requires no additional memory beyond keeping a record that discloses the tabu-active status of individual attributes, and it also prevents cycling for the duration spanned by the tabu-active status. (This assertion about preventing cycling requires qualification, but the assumptions that validate it are natural.) Such an approach yields less restrictive tabu conditions than those based on "disjunctions," where a move is tabu if any of its attributes are tabu-active, and hence provides greater flexibility for choosing moves. In situations where such flexibility can be appropriately exploited (particularly in certain intensification phases of search, as described later), it can be useful to identify "secondary" solution attributes to incorporate in conjunctive restrictions. For example, in the present illustration another type of attribute that is easily accessed and updated is the *node degree* of the endpoints of edges added and dropped (i.e., the number of tree edges that meet each of these endpoint nodes). Creating tabu restrictions that incorporate conjunctions

based on such attributes provides additional options without a significant increase in processing and with a generally tolerable increase in memory.

As a basis for indentifying attributes that may be used in the ways we have illustrated, a natural possibility is to consider the use of *created attributes*, which result by selecting functions of other simple attributes and identifying the values of these functions as attributes for defining tabu restrictions (Glover (1989a)). The objective function qualifies as one such function, but other possibilities merit consideration. The amount of additional memory depends on the number of relevant values of the functions (which may be compressed into intervals). Alternately, this memory can be made to depend on the length of the tabu tenures for such values (if a circular list is used to record the values generated over these tenures), though this can entail more effort to check tabu conditions. Such concerns lie at the heart of proposals for hashing and chunking.

We emphasize again, however, that the use of increasingly relaxed tabu restrictions by these devices is not invariably desirable, since stronger restrictions have an effect of creating a certain vigor in the search process, avoiding "similar" solutions as well as duplicated solutions. However, in phases devoted to searching highly fertile regions more thoroughly, the ability to draw upon less stringent tabu restrictions can ofer advantages.

We have described at some length these issues of creating tabu restrictions that depend on attributive memory, because attributes such as those illustrated are also a basis for types of TS memory other than recency-based memory. At the same time, however, we have left open the question of designing specific memory structures to handle tabu restrictions conveniently. Examples of such structures for recency-based memory, and associated rules for implementing them, are given in Section 2.1.

Aspiration Levels: Expanding the issue of defining tabu conditions at various levels of restrictiveness, an important element of flexibility in tabu search is introduced by means of aspiration

criteria. The tabu status of a solution is not an absolute, but can be overruled if certain conditions are met, expressed in the form of aspiration levels. In effect, these aspiration levels provide thresholds of attractiveness that govern whether the solutions may be considered admissible in spite of being classified tabu. Clearly a solution better than any previously seen deserves to be considered admissible. Similar aspiration criteria can be defined over subsets of solutions that belong to common regions or that share specified features (such as a particular functional value or level of infeasibility). For example, one such aspiration criterion is based on identifying "conditionally best" objective function values that can be attained by moves that start from particular intervals of values for f(x). Then a move is deemed acceptable if it can attain a new best value for the interval it starts from.

The foregoing approach naturally generalizes by replacing intervals of objective function values with other types of intervals. In this case it is often preferable to stipulate that the move attains an improved objective function value in relation to the interval at the end of a move rather than at the start of a move. This corresponds more closely to a standard tabu restriction, except that it is used to override other tabu restrictions. (Implicitly it corresponds to a special type of conjunction.) Additional examples of aspiration criteria are provided later.

Candidate List Strategies: The aggressive aspect of TS is reinforced by seeking the best available move that can be determined with an appropriate amount of effort. It should be kept in mind that the meaning of best is not limited to the objective function evaluation. (As already noted, tabu evaluations are affected by penalties and inducements determined by the search history. They are also affected by considerations of *influence* as subsequently characterized.) For situations where N*(x) is large or its elements are expensive to evaluate, candidate list strategies are used to restrict the number of solutions examined on a given iteration.

Because of the importance TS attaches to selecting elements judiciously, efficient rules for generating and evaluating good candidates are critical to the search process. Even where candidate list

strategies are not used explicitly, memory structures to give efficient updates of move evaluations from one iteration to another, and to reduce the effort of finding best or near best moves, are often integral to TS implementations. Intelligent updating can appreciably reduce solution times, and the inclusion of explicit candidate list strategies, for problems that are large, can significantly magnify the resulting benefits. Useful kinds of candidate list strategies are indicated in Section 2.2.

We now bring these short term elements together and illustrate how they interact in Diagram 2.

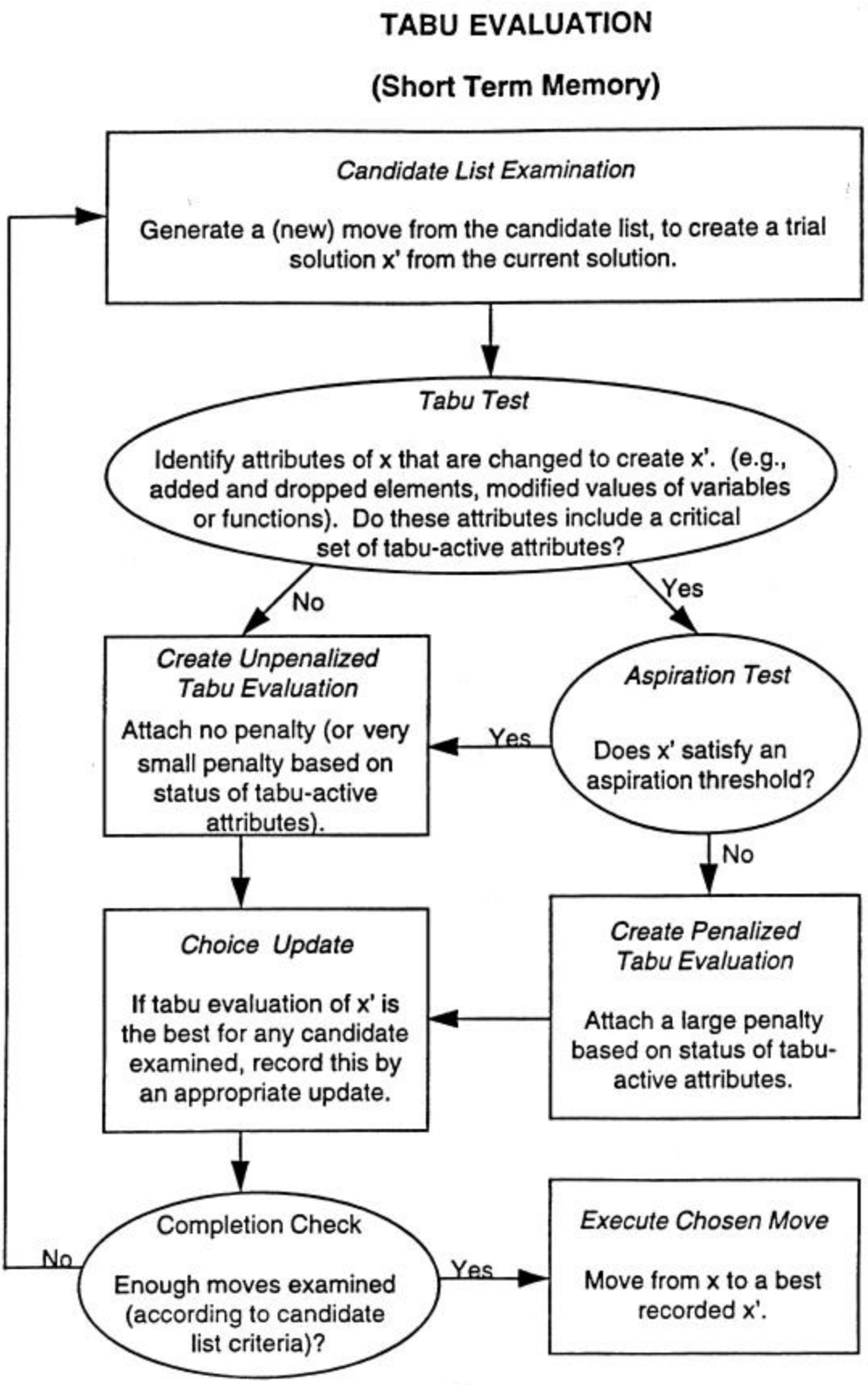

TABU EVALUATION
(Short Term Memory)
Candidate List Examination
Generate a (new) move from the candidate list, to create a trial solution x' from the current solution.
Tabu Test
Identify attributes of x that are changed to create x'. (e.g., added and dropped elements, modified values of variables or functions). Do these attributes include a critical set of tabu-active attributes?
No
Yes
Create Unpenalized Tabu Evaluation
Attach no penalty (or very small penalty based on status of tabu-active attributes).
Aspiration Test
Does x' satisfy an aspiration threshold?
Yes
No
Choice Update
If tabu evaluation of x' is the best for any candidate examined, record this by an appropriate update.
Create Penalized Tabu Evaluation
Attach a large penalty based on status of tabu-active attributes.
Completion Check
Enough moves examined (according to candidate list criteria)?
No
Yes
Execute Chosen Move
Move from x to a best recorded x'.

Diagram 2

The representation of penalties in Diagram 2 either as "large" or "very small" expresses a thresholding effect. In the illustration of Diagram 1, we treated tabu status as an all-or-none type of condition, but differentiation is clearly possible, as by reference to different numbers of tabu-active attributes or to different levels of unexpired tabu tenures. Tabu status generally corresponds to using penalties that yield a greatly deteriorated evaluation or else that chiefly served to break ties among solutions with highest evaluations. (Tie breaking occurs for moves that are no longer tabu in an all-or-none sense, by allowing a lingering diminished influence according to the age of tenures that might otherwise be considered to be expired.) Such an effect of course can be modulated to shift evaluations across levels other than these extremes. If all moves currently available lead to solutions that are tabu (with evaluations that normally would exclude them from being selected), the penalties result in choosing a "least tabu" solution.

It may be noted that the sequence of the Tabu Test and the Aspiration Test in Diagram 2 can be interchanged (that is, by employing the tabu test only if the aspiration threshold is not satisfied). Also, the tabu evaluation can be modified by creating inducements based on the aspiration level, just as it is modified by creating penalties based on tabu status. In this sense, aspiration conditions and tabu conditions can be conceived roughly as "mirror images" of each other.

The TS variant called *probabilistic tabu search* follows a corresponding design, with a short term component that can be represented by the same diagram. The approach additionally keeps track of tabu evaluations generated during the process that results in selecting a move. Based on this record, the move is chosen probabilistically from the pool of those evaluated (or from a subset of the best members of this pool), weighting the moves so that those with higher evaluations are especially favored. Fuller discussions of probabilistic tabu search are found in Glover (1989, 1993), Soriano and Gendreau (1993) and Crainic et al. (1993). Recently, several highly successful implementations of probabilistic tabu search have been developed, particularly for problems involving *noisy evaluations*. The elements underlying these approaches are discussed further in Section 3.6.

1.2 Longer Term Memory

In some applications, the short term TS memory components are sufficient to produce very high quality solutions. However, in general, TS becomes significantly stronger by including longer term memory and its associated strategies. (A number of TS implementations incorporating only short term memory have subsequently been notably improved by introducing longer term memory components.)

Special types of *frequency-based* memory are fundamental to longer term considerations. These operate by introducing penalties and inducements determined by the relative span of time that attributes have belonged to solutions visited by the search, allowing for regional differentiation. *Transition frequencies* keep track of how often attributes change, while *residence frequencies* keep track of relative durations that attributes occur in solutions generated. These memories are also sometimes accompanied by extended forms of recency-based memory.

Perhaps surprisingly, the use of longer term memory does not require long solution runs before its benefits become visible. Often its improvements begin to be manifest in a relatively modest length of time, and can allow solution efforts to be terminated somewhat earlier than otherwise possible, due to finding very high quality solutions within an economical time span. The fastest methods for job shop and flow shop scheduling problems, for example, are based on including longer term TS memory (both explicit memory and attributive memory). On the other hand, it is also true that the chance of finding still better solutions as time grows in the case where an optimal solution is not already found is enhanced by using longer term TS memory in addition to short term memory. Section 2.1 describes forms of frequency-based memory that provide a basis for useful longer term strategies.

Intensification and Diversification: Two highly important longer term components of tabu search are *intensification strategies* and *diversification strategies*. Intensification strategies are based on modifying choice rules to encourage move combinations and solution features historically found good. They can be applied with constructive and destructive neighborhoods as well as transition

neighborhoods, as in restarting procedures that seek to incorporate good attributes into the current level of construction or destruction (conditional upon attributes previously incorporated). Such approaches have worked well by the choice rule designs of probabilistic tabu search (see, e.g., Rochat and Taillard (1995)). Intensification strategies may also initiate a return to attractive regions to search them more thoroughly. A simple instance of this latter type of approach is shown in Diagram 3.

Simple TS Intensification Approach

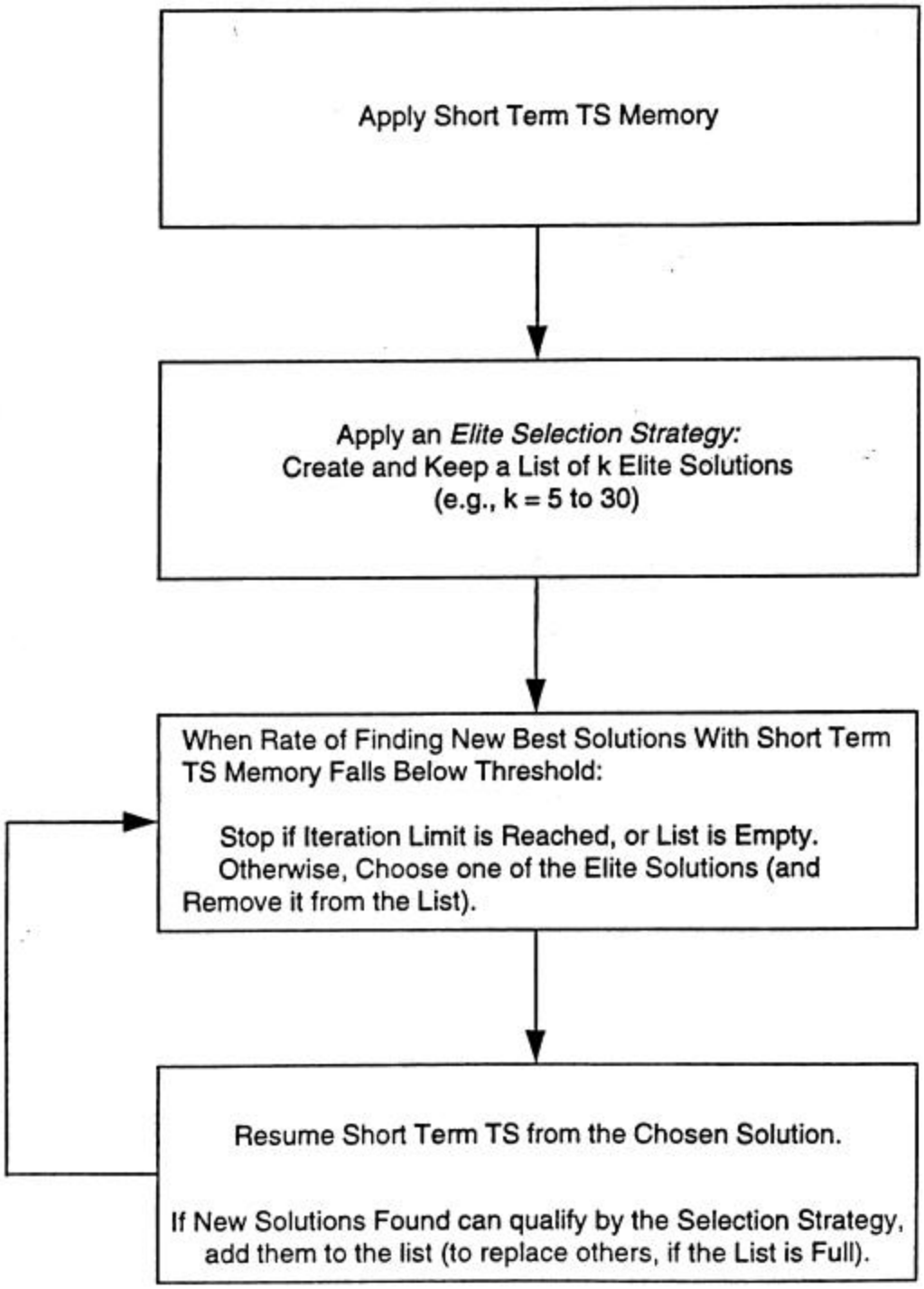


Diagram 3

The strategy for selecting elite solutions is italicized in Diagram 3 due to its importance. Three variants have proved quite successful. One, due to Voss (1993), introduces a diversification measure to assure the solutions recorded differ from each other by a desired degree, and then erases all short term memory before resuming from the best of the recorded solutions. The second variant, due to Nowicki and Smutniki (1993), keeps a bounded length sequential list that adds a new solution at the end only if it is better than any previously seen. The current last member of the list is always the one chosen (and removed) as a basis for resuming search. However, TS short term memory that accompanied this solution also is saved, and the first move also forbids the move previously taken from this solution, so that a new solution path will be launched. (A similar approach, also highly effective, has been introduced in Barnes and Chambers (1992).) The third variant, due to Xu, Chiu and Glover (1995), maintains an ordered list of k best solutions, and then, after a specified number of iterations, begins at the worst member of this list and progresses toward the best. The currently selected member launches a new search, using probabilistic tabu search as an alternative to recovering prior memory. Only a fixed number of iterations is permitted upon restarting from such a solution, before recovering the next, but the list continues to be updated. That is, whenever a better solution is found than the worst that remains to be examined, this new solution is inserted in its appropriate position and the worst remaining solution is removed from the list. This strategy proved quite effective for problems in telecommunication network design.

These approaches are an instance of what is sometimes called a *restructured move* approach, reflecting the fact that the normal set of moves is periodically modified to allow a direct jump to a solution outside the customary neighborhood. A related form of this approach keeps tracks of best unvisited neighbors (from those examined on candidate lists), with a provision for restricting attention to specific types of solutions, such as neighbors of local optima or neighbors of solutions visited on steps immediately before reaching such local optima (Glover (1990a)). Although this "unvisited neighbor"

strategy appears to be unexplored, it is noteworthy that the related strategies previously indicated have provided solutions of remarkably high quality. For example, the study of Vaessens, Aarts and Lenstra (1994) documents that the approach of Nowicki and Smutnicki (1993) is unsurpassed for solving job shop scheduling problems.

Another type of intensification approach is *intensification by decomposition*, where restrictions may be imposed on parts of the problem or solution structure in order to generate a form of decomposition that allows a more concentrated focus on other parts of the structure. A classical example is provided by the traveling salesman problem, where edges that belong to the intersection of elite tours may be "locked into" the solution, in order to focus on manipulating other parts of the tour. The use of intersections may be seen as an extreme instance of a more general strategy that seeks to identify and constrain the values of *strongly determined* and *consistent variables*. In this approach, frequency information keeps track of variables that receive particular values (or that lie in particular ranges) in subsets of elite solutions (Glover (1977)). The quality of the solutions in which these values assignments occur, and the disruptive effect of changing these assignments, provide measures of their strength. Constraining the values of appropriate variables by such information can lead to identifying additional variables that qualify to be similarly constrained, thus imparting a recursive element to the approach. The overall effect may be likened to creating a *combinatorial implosion* of possibilities (in reverse analogy to the notion of a combinatorial explosion), since constraining discrete variables, as by temporarily fixing and dropping them, operates in exactly the opposite way as adding new discrete variables. This type of intensification approach has been applied highly effectively in vehicle routing by Rochat and Taillard (1995).

Intensification by decomposition also encompasses other types of strategic considerations, basing the decomposition not only on indicators of strength and consistency, but also on opportunities for particular elements to interact productively. Within the context of coordinated permutation problems

that can be conveniently defined by reference to graphs (as in scheduling, vehicle routing and TSPs), a decomposition may be based on identifying subchains of an elite solutions, where two or more subchains may be assigned to a common set if they contain nodes that are "strongly attracted" to be linked with nodes of other subchains in the set. An edge disjoint collection of subchains can be treated by an intensification process that operates in parallel on each set, subject to the restriction that the identity of the endpoints of the subchains will not be altered. As a result of the decomposition, the best new sets of subchains can be reassembled to create a new solution. Such a process can be applied to multiple alternative decompositions in broader forms of intensification by decomposition.

Diversification Strategies: TS diversification strategies, as their name suggests, are designed to drive the search into new regions. Often they are based on modifying choice rules to bring attributes into the solution that are infrequently used. Alternatively, they may introduce such attributes by partially or fully re-starting the solution process.

The same types of memories previously described are useful as a foundation for such procedures, although these memories are maintained over different (generally larger) subsets of solutions than those maintained by intensification strategies. A simple diversification approach that keeps a frequency-based memory over all solutions previously generated, and that has proved very successful for machine scheduling problems, is shown in Diagram 4.

Simple TS Diversification Approach

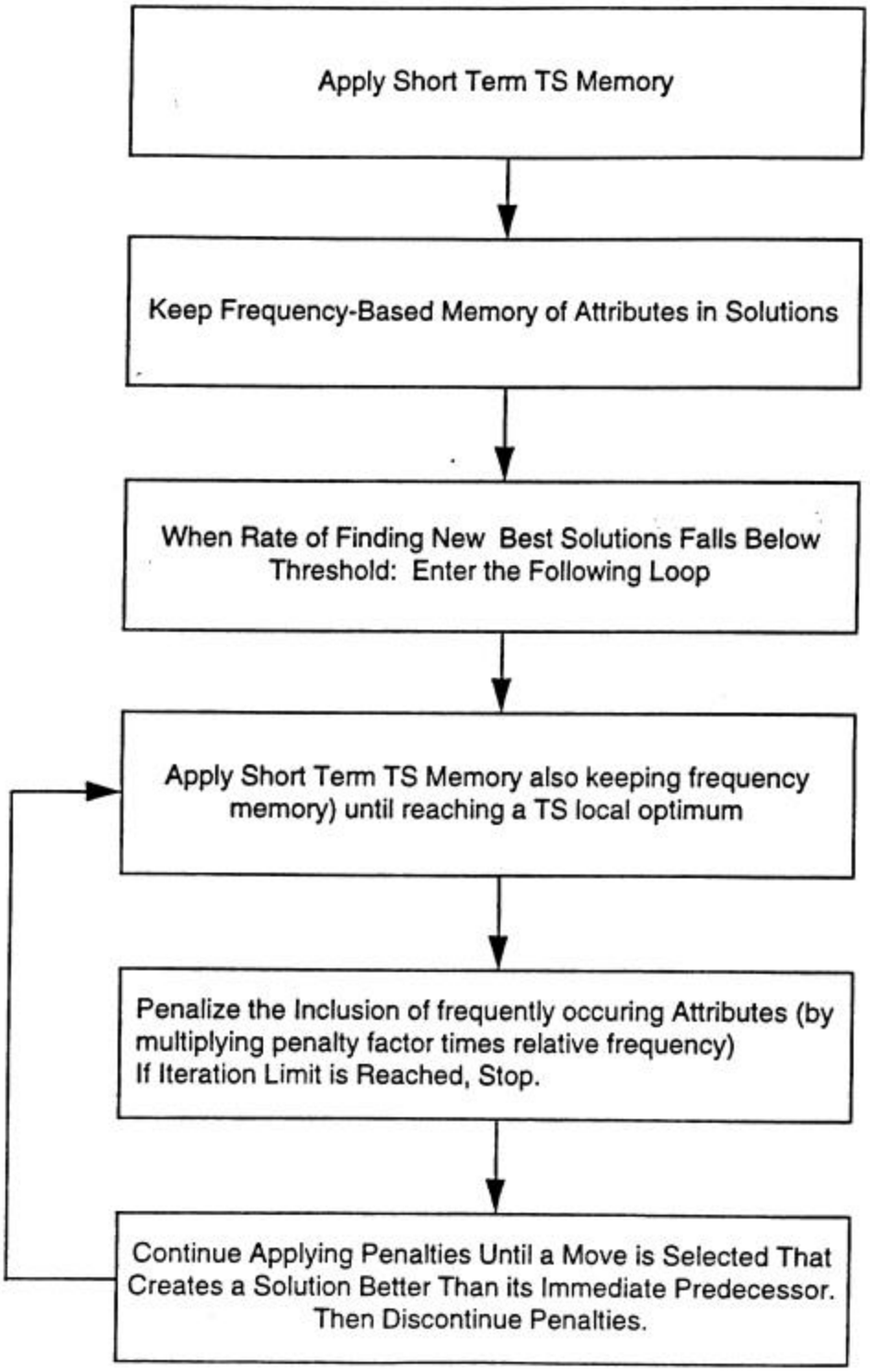


Diagram 4

Significant improvements over the application of short term TS memory have been achieved by the procedure of Diagram 4 (see Laguna and Glover (1993)). However, it should be stressed that the timing for introducing diversification in this approach is important. Diversification is not applied arbitrarily but only at local optima. In addition, best moves are still selected to guide the process (subject to diversifying penalties, which have a limited period of operation).

The TS local optima reached by this approach, and used as a basis for launching a sequence of diversifying steps, naturally may differ from true local optima since tabu search choice rules may exclude some improving moves. The success of this approach suggests the merit of incorporating a TS variant that always continues to a true local optimum once an improving move becomes an acceptable choice based on an aspiration criterion that is activated only after executing an improving move. In this approach, as long as additional improving moves exist, the aspiration criterion allows one of them to be selected, by a tabu evaluation rule that penalizes choices based on their tabu status (restricting attention to the improving set). Once a true local optimum is reached, the special aspiration criterion is discontinued until a new improving move is selected by standard TS rules. This approach embodies an instance of *aspiration by search direction*, and can be usefully refined by taking *spheres of influence* into account (Glover and Laguna (1993)).

The precise manner in which frequency-based memories are used to implement strategies of intensification and diversification (apart from defining these memories over different subsets) provides a fertile area for investigation. Two different general patterns for exploiting these memories are illustrated in Diagram 5.

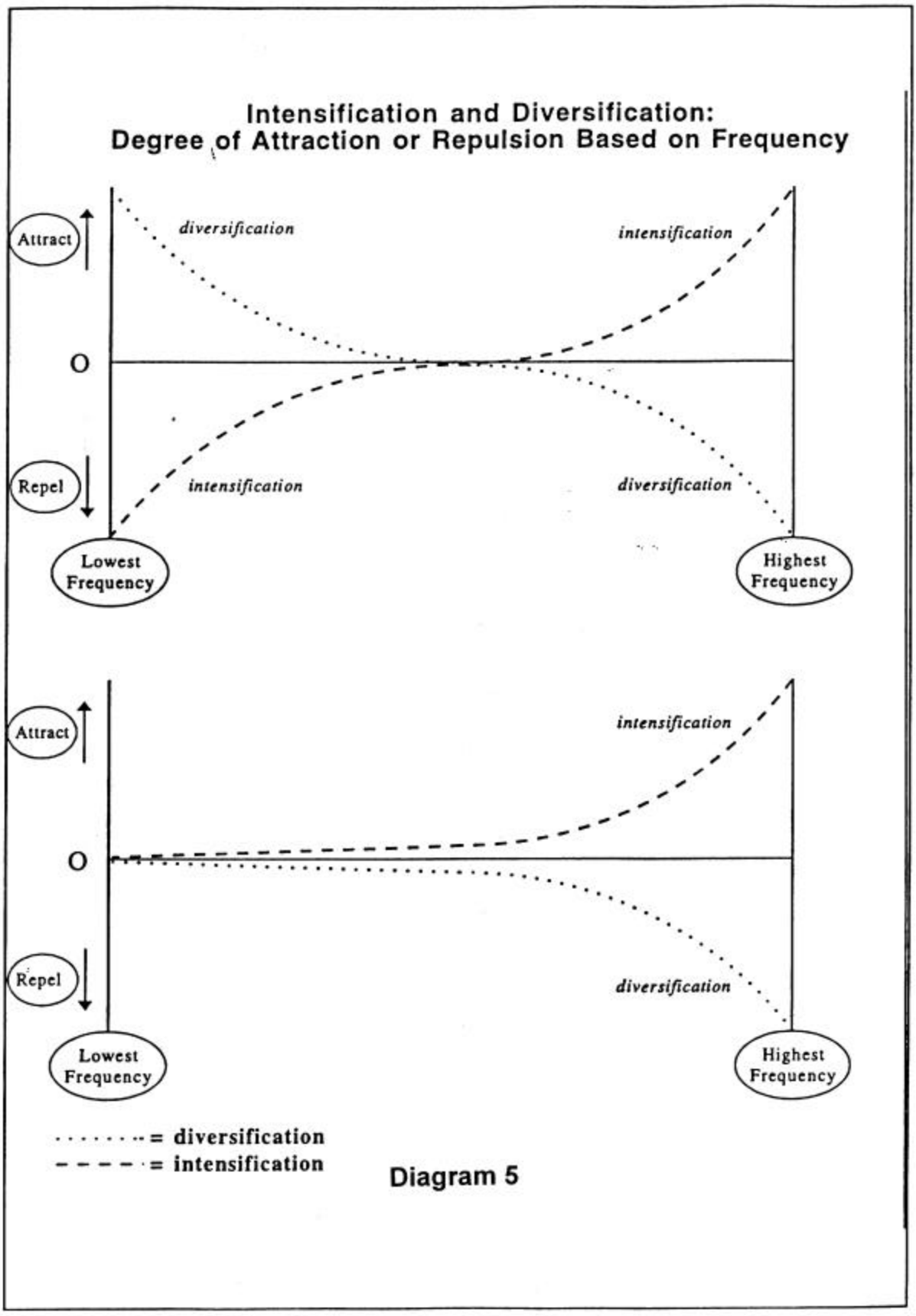

Intensification and Diversification:
Degree of Attraction or Repulsion Based on Frequency
Attract
diversification
intensification
O
Repel
intensification
diversification
Lowest Frequency
Highest Frequency
Attract
intensification
O
Repel
diversification
Lowest Frequency
Highest Frequency
..... = diversification
- - - - = intensification

Diagram 5

A variety of additional alternatives can be inferred from natural variations in these patterns. Diversification strategies that create partial or full restarts are important for problems and neighborhood structures where a solution trajectory can become isolated from worthwhile new alternatives unless a radical change is introduced. Special forms of diversification in these cases have been developed by Hertz and de Werra (1991), Gendreau, Hertz and Laporte (1991), Soriano and Gendreau (1993), Porto and Ribeiro (1993), and Hubscher and Glover (1993).

Diversification strategies can also utilize a long term form of recency-based memory, which results by increasing the tabu tenure of solution attributes. A simple version of this approach that has produced good results (Kelly *et al*. (1991)) is shown in Diagram 6.

The reason for the success of this approach is that it is not "blind" diversification, but implicitly incorporates intensification. That is, each move continues to be made by an aggressive choice rule that selects the best available option from those admissible (again, allowing the use of candidate list strategies, as elaborated in Section 2.2. The enforced requirement of moving progressively away from a particular solution may compel some unattractive moves, but still the best moves are tabu in the class considered. The goal may be expressed as that of *influential diversification*, where *influence* includes the concept of quality. In a probabilistic sense, diversification of this type (that includes intensification concerns) is a stronger form of diversification, under the expectation that solutions of higher quality are distributed in the solution space so that the probability of encountering them is relatively small. Thus, a solution that is "far from" another, but that is of high quality, is less likely to be reached than by a series of random moves that apply the same number of steps. The notion can be refined by considering separation from more than one "reference solution" simultaneously, and by using the path relinking concept subsequently discussed.

Diversification Using Long Term Recency-Based Memory

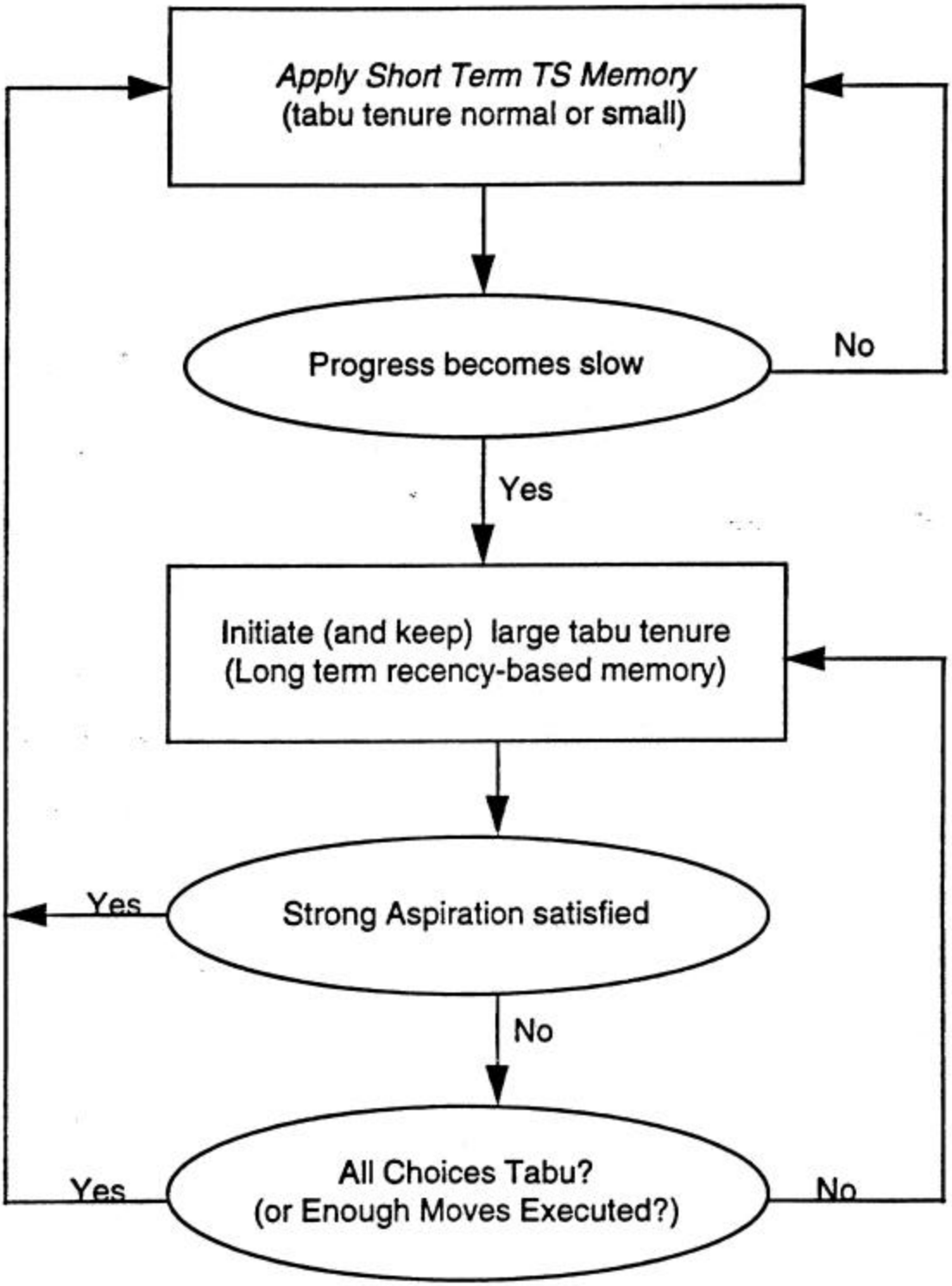


Diagram 6

The determination of effective ways to balance the concerns of intensification and diversification represents a promising research area. These concerns also lie at the heart of effective parallel processing implementations. The goal from the TS perspective is to design patterns of communication and information sharing across subsets of processors in order to achieve the best tradeoffs between intensification and diversification functions. General analyses and studies of parallel processing with tabu search are given in Taillard (1991, 1993), Battiti and Tecchioli (1992b), Chakrapani and Skorin-Kapov (1991, 1993), Crainic, Toulouse and Gendreau (1993a, 1993b), and Voss (1994).

1.3 Strategic Oscillation

Strategic oscillation is closely linked to the origins of tabu search, and provides a means to achieve an effective interplay between intensification and diversification over the intermediate to long term. The approach operates by orienting moves in relation to a *critical level*, as identified by a stage of construction or a chosen interval of values for a functional.

Such a critical level often represents a point where the method would normally stop. Instead of stopping when this level is reached, however, the rules for selecting moves are modified, to permit the region defined by the critical level to be crossed. The approach then proceeds for a specified depth beyond the critical level, and turns around. The critical level again is approached and crossed, this time from the opposite direction, and the method proceeds to a new turning point.

The process of repeatedly approaching and crossing the critical level from different directions creates an oscillatory behavior, which gives the method its name. Control over this behavior is established by generating modified evaluations and rules of movement, depending on the region navigated and the direction of search. The possibility of retracing a prior trajectory is avoided by standard tabu search mechanisms.

A simple example of this approach occurs for the multidimensional knapsack problem, where values of zero-one variables are changed from 0 to 1 until reaching the boundary of feasibility. The

method then continues into the infeasible region using the same type of changes, but with a modified evaluator. After a selected number of steps, the direction is reversed by choosing moves that change variables from 1 to 0. Evaluation criteria to drive toward improvement vary according to whether the movement occurs inside or outside the feasible region (and whether it is directed toward or away from the boundary), accompanied by associated restrictions on admissible changes to values of variables. Implementations of such an approach by Freville and Plateau (1986, 1992) and more recently by Glover and Kochenberger (1995), have generated particularly high quality solutions for multidimensional knapsack problems.

A somewhat different type of application occurs for graph theory problems where the critical level represents a desired form of graph structure, capable of being generated by progressive additions (or insertions) of basic elements such as nodes, edges, or subgraphs. One type of strategic oscillation approach for this problem results by a constructive process of introducing elements until the critical level is reached, and then introducing further elements to cross the boundary defined by the critical level. The current solution may change its structure once this boundary is crossed (as where a forest becomes transformed into a graph that contains loops), and hence a different neighborhood may be required, yielding modified rules for selecting moves. The rules again change in order to proceed in the opposite direction, removing elements until again recovering the structure that defines the critical level. Such rule changes based on the direction and phase of search are typical features of strategic oscillation, and provide an enhanced heuristic vitality. The application of different rules may be accompanied by crossing a boundary to different depths on different sides. An option is to approach and retreat from the boundary while remaining on a single side, without crossing (i.e., electing a crossing of "zero depth").

Both of these examples constitute a constructive/destructive type of strategic oscillation, where constructive steps "add" elements (or set variables to 1) and destructive steps "drop" elements (or set

variables to 0). One-sided oscillations are often especially relevant in constructive/destructive approaches, as in the context of a variety of scheduling and graph theory problems, where a useful structure can be maintained up to a critical point and then is lost (by running out of jobs to assign, or by going beyond the conditions that define a tree or tour, etc.). In these cases, the constructive process builds to the critical level, and then reverses to apply destructive moves. Once a constructive phase consisting of a series of "add moves" is completed, the most attractive "drop move" for the destructive phase is likely to have little relation to the sequence in which elements were added. Nevertheless, TS memory structures are still needed to assure the alternating phases do not effectively cancel each other. A special type of memory structure that has proved highly effective for this, yielding best results in the literature for applications related to resource allocation, is indicated in Section 2.3.

In strategic oscillation approaches it is frequently important to spend additional search time in regions close to the critical level, and especially to spend time at the critical level itself. This may be done by inducing a sequence of tight oscillations about the critical level, as a prelude to each larger oscillation that proceeds to a greater depth. Alternately, if greater effort is permitted for evaluating and executing each move, the method may use "exchange moves" (broadly interpreted) to stay at the critical level for longer periods. A simple option, for example, is to use such exchange moves to proceed to a local optimum each time the critical level is reached. A strategy of similarly applying exchanges at additional levels is suggested by a *proximate optimality principle*, which states roughly that good constructions at one level are likely to be close to good constructions at another. (See Section 3.2.) A simple version of a constructive/destructive form of strategic oscillation is illustrated in Diagram 7. As observed in the table accompanying Diagram 7, the oscillation can also operate by increasing and decreasing bounds for a function g(x). Such an approach has been the basis for a number of effective applications, where g(x) has represented such items as workforce assignments, objective function values, and feasibility/infeasibility levels, to guide the search to probe at various depths with the

associated regions.

When the levels refer to degrees of feasibility and infeasibility, g(x) is a vector-valued function associated with a set of problem constraints (which may summarized, for example, by g(x) b). In this case, controlling the search by bounding g(x) can be viewed as manipulating a parameterization of the selected constraint set. A preferred alternative is often to make g(x) a Lagrangean or surrogate constraint penalty function, avoiding vector-valued functions and allowing tradeoffs between degrees of violation of different component constraints. (Approaches that embody such ideas may be found, for example, in Freville and Plateau (1986), Gendreau, Hertz and Laporte (1993), Kelly, Golden and Assad (1993), Osman (1993), Osman and Christofides (1993), Rochat and Semet (1993), and Voss (1993).)

1.4 Path Relinking

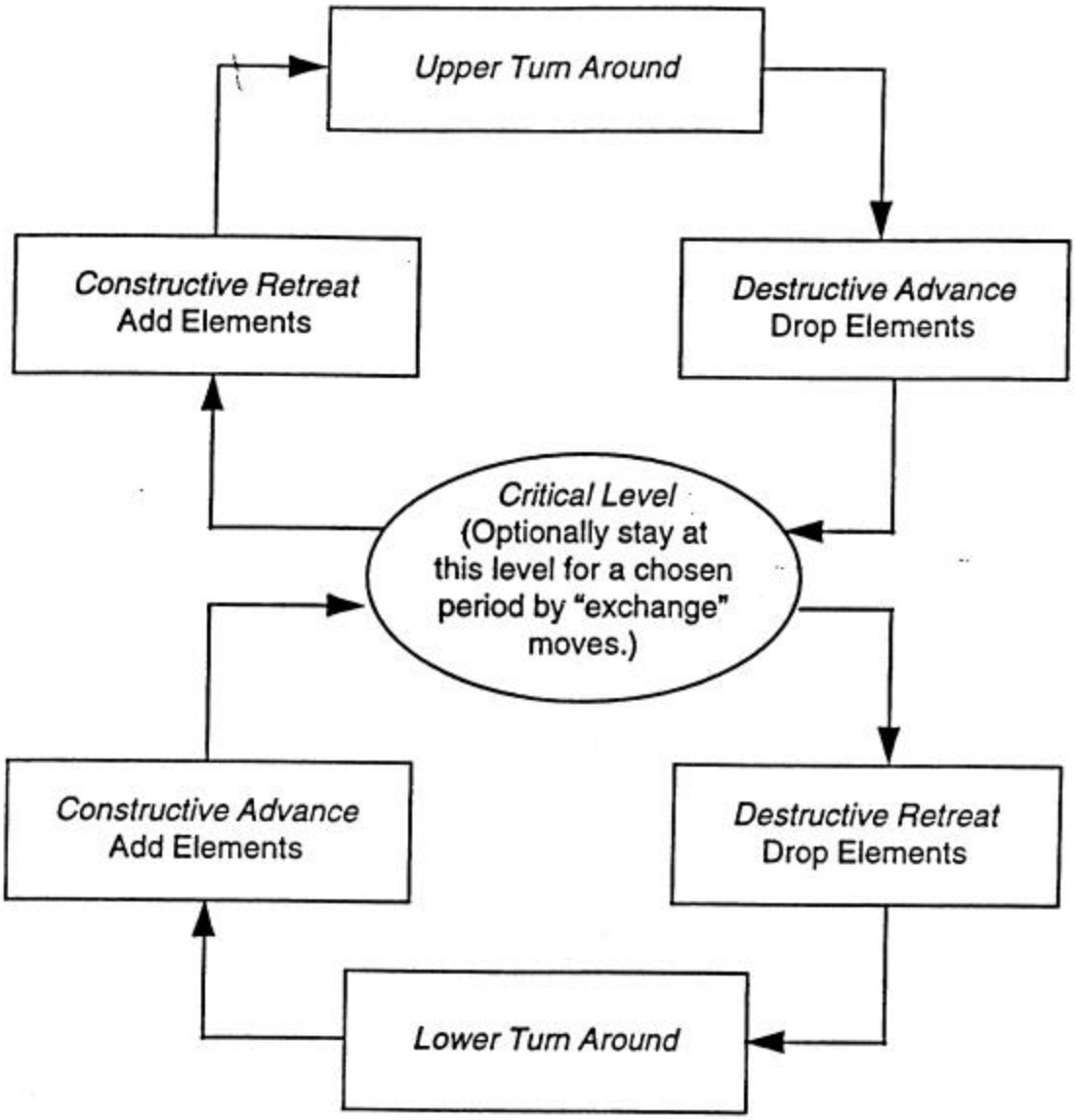
A useful integration of intensification and diversification strategies occurs in the approach called *path relinking* (Glover (1989a, 1993)). This approach generates new solutions by exploring trajectories that "connect" elite solutions by starting from one of these solutions, called an *initiating solution*, and generating a path in neighborhood space that leads toward the other solutions, called *guiding solutions*. This is accomplished by selecting moves that introduce attributes contained in the guiding solutions. (As will be seen, the initiating solution can also be a "null" or "overspecified" solution when constructive and destructive neighborhoods are used.)

The approach may be viewed as an extreme (highly focused) instance of a strategy that seeks to incorporate attributes of high quality solutions, by creating inducements to favor these attributes in the moves selected. However, instead of using an inducement that merely encourages the inclusion of such attributes, the path relinking approach subordinates all other considerations to the goal of choosing moves that introduce the attributes of the guiding solutions, in order to create a "good attribute composition" in the current solution. The composition at each step is determined by choosing the best

move, using customary choice criteria, from the restricted set of moves that incorporate a maximum number (or a maximum *weighted value*) of the attributes of the guiding solutions. As in other applications of TS, aspiration criteria can override this restriction to allow other moves of particularly high quality to be considered.

Specifically, upon identifying a collection of one or more elite solutions to guide the path of a given solution, the attributes of these guiding solutions are assigned preemptive weights as inducements to be selected. Larger weights are assigned to attributes that occur in greater numbers of the guiding solutions, allowing bias to give increased emphasis to solutions with higher quality or with special features (e.g., complementing those of the solution that initiated the new trajectory). More generally, it is not necessary for an attribute to occur in a guiding solution in order to have a favored status. In some settings attributes can share degrees of similarity, and in this case it can be useful to view a solution vector as providing "votes" to favor or discourage

Strategic Oscillation - Using Constructive/Destructive Moves



Move types	*Example Alternatives*	
Constructive	Set $x_j = 1$	increase bounds (L,U) for g(x)*
Destructive	Set $x_j = 0$	decrease bounds (L,U) for g(x)
Exchange	Set $x_p = 1$, $x_q = 0$	maintain bounds for g(x)

* g(x) can be scalar or vector valued function of solution x

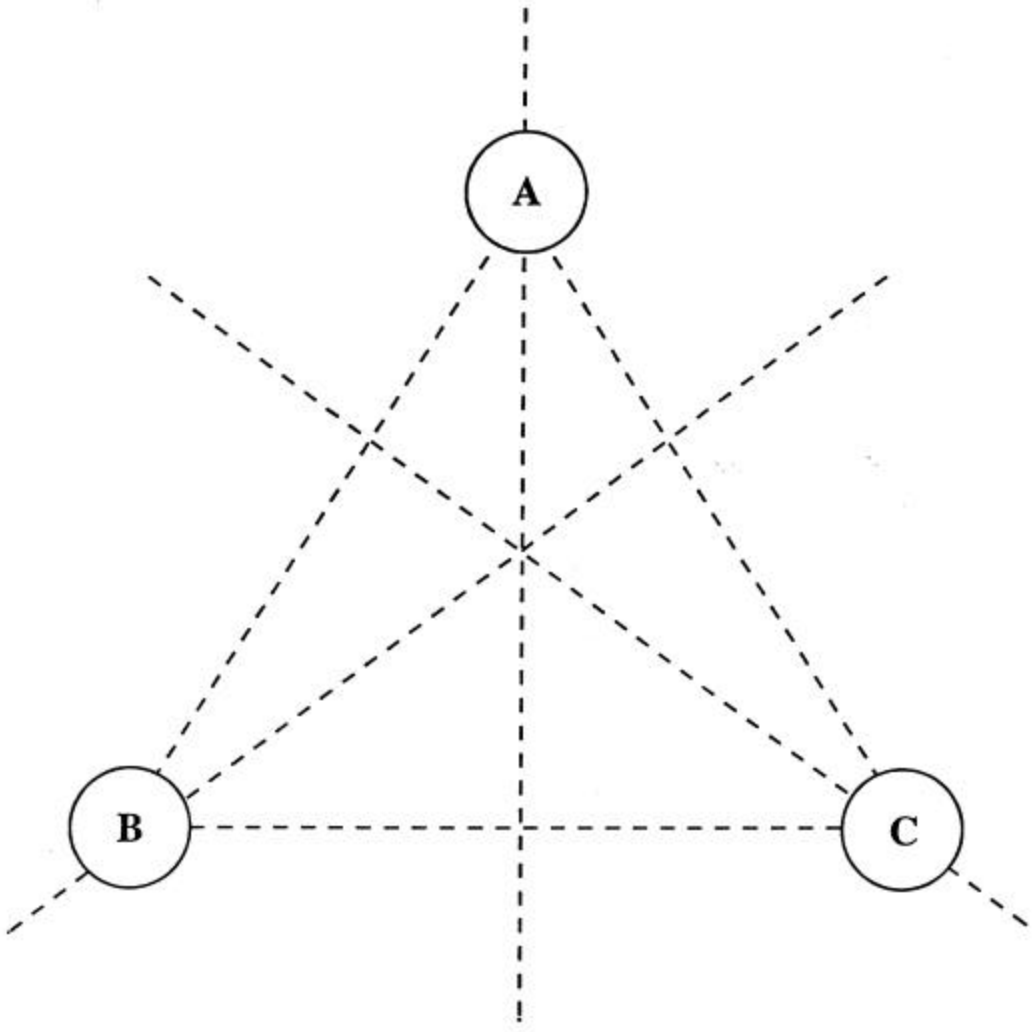
Diagram 7

particular attributes (Glover (1991)). Typically, only the strongest forms of aspiration criteria are allowed to overcome this type of choice rule.

In a given collection of elite solutions, the role of initiating solution and guiding solutions can be alternated. That is, a set of current solutions may be generated simultaneously, extending different paths, and allowing an initiating solution to be replaced (as a guiding solution for others) whenever its associated current solution satisfies a sufficiently strong aspiration criterion. Because their roles are interchangeable, the initiating and guiding solutions are collectively called *reference solutions*.

An idealized form of such a process is shown in Diagram 8. The chosen collection of reference solutions consists of the three members, A, B and C. Paths are generated by allowing each to serve as initiating solution, and by allowing either one or both of the other two solutions to operate as guiding solutions. Intermediate solutions encountered along the paths are not shown. The representation of the paths as straight lines of course is oversimplified, since choosing among available moves in a current neighborhood will generally produce a considerably more complex trajectory.

Path Relinking in Neighborhood Space



Intensification: Generate paths from similar solutions

Diversification: Generate paths from dissimilar solutions

Aspiration: Explore deviations from paths at attractive neighbors

Diagram 8

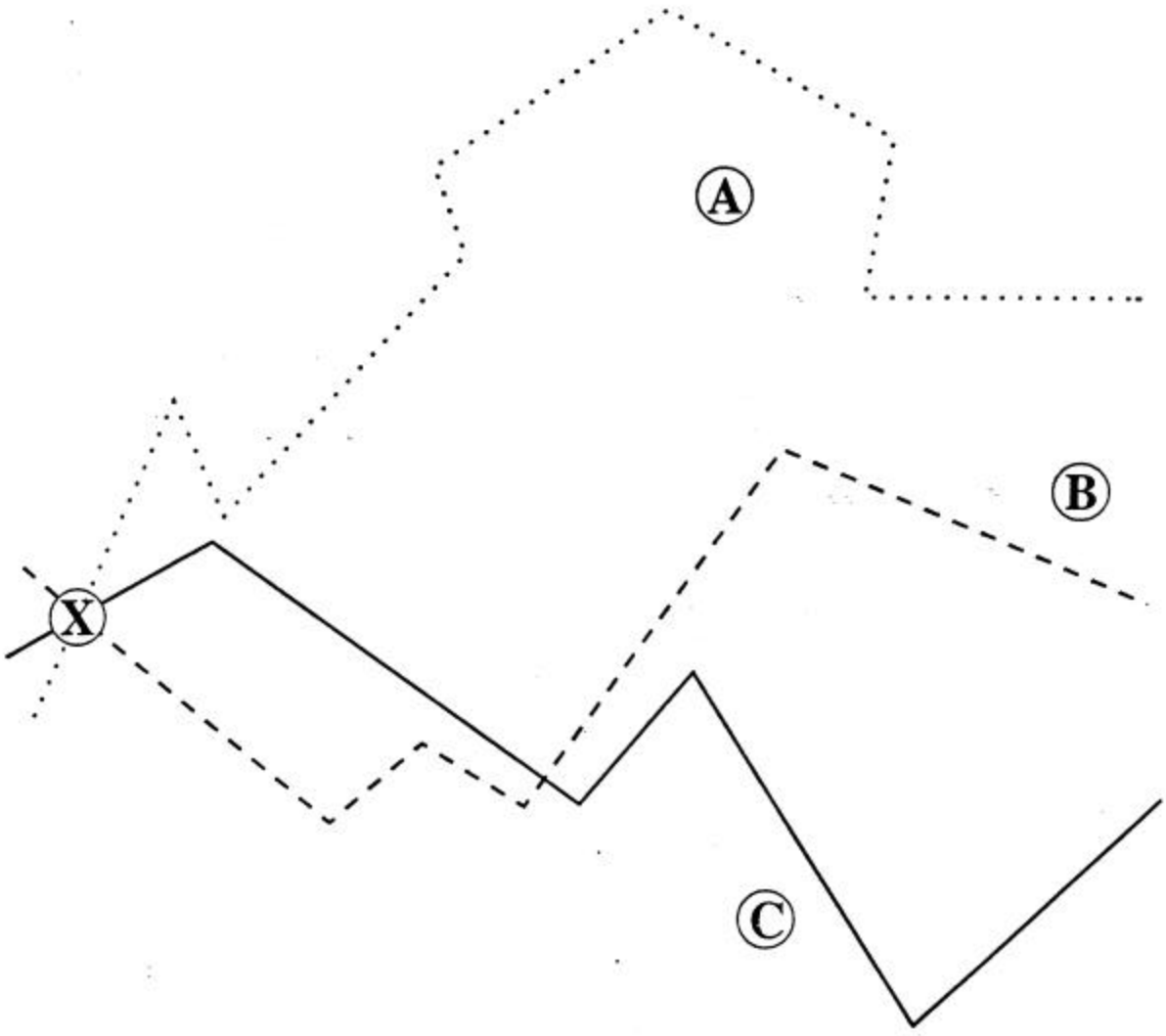
As Diagram 8 indicates, at least one path continuation is allowed beyond each initiating/guiding solution. Such a continuation can be accomplished by penalizing the inclusion of attributes dropped during a trajectory, including attributes of guiding solutions that may be compelled to be dropped in order to continue the path. (An initiating solution may also be repelled from the guiding solutions by penalizing the inclusion of their attributes from the outset.) Probabilistic TS variants operate in the path relinking setting, as in others, by translating evaluations for deterministic rules into probabilities of selection, strongly biased to favor higher evaluations.

Promising regions may be searched more thoroughly in path relinking by modifying the weights attached to attributes of the guiding solutions, and by altering the bias associated with solution quality and selected solution features. Diagram 9 depicts the type of variation that can result, where the point X represents an initiating solution and the points A, B and C represent guiding solutions. Variations of this type within a promising domain are motivated by the proximate optimality principle discussed in connection with strategic oscillation. For appropriate choices of the reference points (and neighborhoods for generating paths from them), this principle suggests that additional elite points are likely to be found in the regions traversed by the paths, upon launching new searches from high quality points on these paths. Evidence that combinatorial solution spaces often have topologies that may be usefully exploited by such an approach is provided by findings of Moscato (1993), Moscato and Tinetti (1994), and Nowicki and Smutnicki (1993, 1994). Additional aspects of path relinking are examined in Section 2.4.

2. Illustrative Tabu Search Memory Structures and Strategies

This section focuses on considerations relevant for implementing tabu search, with an emphasis on examples to illustrate main ideas.

Path Relinking by Attribute Bias



X = Solution selected to generate a relinked path.
(Different solutions may be selected to take the role of X, simultaneously or alternately.)

Neighborhood Space Paths with Different Attribute Trade-offs.

Diagram 9

2.1 Recency-Based and Frequency-Based Memory Structures

Recency-Based Memory Structures: -- We begin by indicating some commonly used recency-based memory structures for identifying attributes that are tabu-active, and for determining the tabu status of solutions containing these attributes. Let S = {1, 2,...,s} denote an index set for a collection of solution attributes. For example, the indexes $i \in S$ may correspond to indexes of zero-one variables x_i, or they may be indexes of edges that may be added or deleted from a graph. (More precisely, attributes referenced by S in these two cases consist of the specific values assigned to the variables or the specific add/drop states adopted by the edges.) In general, an index $i \in S$ can summarize more detailed information; e.g., by referring to an ordered pair (j,k) that summarizes a value assignment $x_j = k$. Hence, the index i may be viewed as a notational convenience for representing a pair or a vector, etc. (Consideration can often be limited to move representations in which only a very small number of attributes or critical attributes change at a time. E.g., a pivot step, which changes the values of many variables, can be recorded by indicating that just two variables change their states, one entering and one leaving a basis.)

For the present illustration, suppose that each $i \in S$ corresponds to a 0 - 1 variable x_i. We will not bother to write (i,0) and (i,1) to identify the two associated attributes $x_i = 0$ and $x_i = 1$ (since by knowing the current value of x_i we also know its unique alternative value). To record recency-based TS information for each variable, we keep track of iterations by an iteration counter denoted *current_iteration,* which starts at 0 and increases by 1 each time a move is made.

When a move is executed that causes a variable x_i to change its value, we record *tabu_start(i) = current_iteration* immediately after updating the iteration counter. This means that if the move has resulted in $x_i = 1$, then the attribute $x_i = 0$ becomes tabu-active at the iteration *tabu_start(i)*. Further, we stipulate that this attribute will remain tabu-active for a number of iterations equal to *tabu_tenure(i),* whose value will be determined in a manner soon to be indicated. Thus, in particular, the recency-

based tabu criterion says that the previous value of x_i is tabu-active throughout all iterations such that

$$tabu\_start(i) + tabu\_tenure(i) \quad current\_iteration.$$

Once *current_iteration* increases to the point where this inequality no longer holds, x_i will no longer be tabu-active at its previous value and hence will not be discouraged from receiving this value again.

The value *tabu_start(i)* can be set to 0 before initiating the method, as a convention to indicate no prior history exists. Then we automatically avoid assigning a tabu-active status to any variable with *tabu_start(i)* = 0 (since the starting value for variable x_i has not yet been changed).

For convenience in the following we will refer to a *variable* x_i as tabu-active with the understanding that the tabu-active condition applies to a specific associated *attribute* the attribute x_i = k where k is the last value previously assigned to x_i. If only one variable changes its value on an iteration, a move may be classified tabu whenever it changes the value of a tabu-active variable. However, if two variables change their values, as where one is set to 0 and the other to 1, then there are several choices. For example, the move can be designated tabu if:

(a) both variables are tabu-active

(b) either variable is tabu-active

(c) the variable that changes from 0 to 1 is tabu-active

The possibility that the tabu status should depend on a particular change of values, as in (c), can also be reflected by giving *tabu_tenure(i)* a different value according to the value assigned to x_i.

The choice of a preferred value for *tabu_tenure(i)* is customarily based on empirical test, starting by considering a common value for all attributes (or for all attributes in a specific class). Experience shows that options can then be quickly narrowed to a range where every value in the range gives good results, particularly if the value is treated as the center of a small interval in which *tabu_tenure(i)* is varied, either systematically or randomly. For example, the approximate outlines of such a range can be quickly inferred by investigating values that are multiples of 5 or 7.

Tabu tenure values for given classes of problems typically can be expressed as a simple function of the total number of attributes (such as fraction or a multiple of the square root of s). For increased refinement, such values then can be differentiated according to types of attributes as for example, according to assignments $x_i = 0$ or 1, and according to specific types of variables. These types of refinements can be made adaptively within the solution process itself, by monitoring the consequences of chosen alternatives. For example, Laguna et al. (1992) monitor the quality of moves associated with particular attribute changes, and vary the tabu tenure of the attributes as they participate in moves of greater or lesser attractiveness. In another type of approach, Kelly et al. (1991) keep track of patterns of objective function value changes, and modify the tabu status of moves when the pattern suggests the possibility of cycling. Battiti and Tecchioli (1992a) provide an effective method that uses a hashing function as a cycling indicator, and directly modifies an overall tabu tenure value as the search process continues, until this value is just large enough to eliminate traces of cycling. (This type of approach can be extended by taking advantage of the chunking ideas of Woodruff (1993).)

A dynamic strategy with a somewhat different foundation determines tabu status without relying on a tabu tenure at all, but by accounting for logical relationships in the sequence of attribute changes. Appropriate reference to these relationships makes it possible to determine in advance if a particular current change can produce cycling, and thus to generate tabu restrictions that are both necessary and sufficient to keep from returning to previous solutions (Glover (1990)). A small tabu tenure introduces extra vigor into the search, since the avoidance of cycling is not the only goal of recency-based memory. (In addition, a "bounded memory span" reduces overhead and provides increased flexibility, as where it may sometimes be preferable to revisit solutions previously encountered.) This means of exploiting logical interdependencies also provides information that is useful for diversification strategies. Innovative implementations have been developed by Dammeyer and Voss (1991) and Voss (1992, 1993).

While interesting opportunities exist for applying advanced forms of recency-based memory in

tabu search, it is to be noted that simpler forms often work quite well. This motivates the use of straightforward types of memory as a basis for developing initial TS implementations. Experience with such implementations can then suggest the basis for productive elaborations. This feature of tabu search, which makes it possible to introduce refinements by natural stages, is particularly useful for progressing to designs that incorporate longer term memory.

Frequency-Based Memory Structures: -- Again we consider the setting of a zero-one optimization problem, and make reference to an attribute set S = {1,...,s} that consists of indexes of 0-1 variables x_i. The form of transition memory to record the number of times x_i changes its value consists simply of keeping a counter for x_i that is incremented at each move where such a change occurs. Since x_i is a zero-one variable, such a memory also discloses the number of times x_i changes to and from each of its possible assigned values. (In more complex situations, a matrix memory can be used to determine numbers of transitions involving assignments such as $x_j = k$.) However, in using this memory, penalties and inducements are based on *relative* numbers (rather than absolute numbers) of transitions, hence requiring that recorded transition values are divided by the total number of iterations (or the total number of transitions).

Residence memory requires only slightly more effort to maintain than transition memory, by taking advantage of the recency-based memory stored in *tabu_start(i)*. The following approach can be used to track the number of solutions in which $x_i = 1$, thereby allowing the number of solutions in which $x_i = 0$ to be inferred from this. Start with *residence(i)* = 0 for all i. Then, whenever x_i changes from 1 to 0, after updating *current_iteration* but before updating *tabu_start(i)*, set

residence(i) = residence(i) + current_iteration - tabu_start(i).

Then, during iterations when $x_i = 0$, *residence(i)* correctly stores the number of earlier solutions in which $x_i = 1$. During iterations when $x_i = 1$, the "true" value of *residence(i)* is the right hand side of the preceding assignment, but the update only has to be made at the indicated points when x_i changes from

1 to 0.

As with transition memory, residence memory should be translated into a relative measure (dividing by the total number of iterations, hence solutions generated), as a basis for creating penalties and inducements. The preferred magnitude of penalties and inducements, when not preemptive, is established by empirical test. (See, for example, Laguna and Glover (1993) and Gendreau, Soriano and Salvail (1993).)

There are a number of ways of taking advantage of frequency-based memory. To illustrate some of the basic possibilities, we may consider dividing move attributes into six frequency classes, according to whether these attributes: (1) often occur in good (or very good) solutions; (2) often occur in poor solutions but rarely in good solutions; (3) often occur in moves to add the attribute to the current solution, where these moves receive evaluations that are high, but not "high enough" to be chosen; (4) often occur in moves to drop the attribute from the current solution, where these moves similarly receive evaluations insufficiently attractive to be chosen; (5) often occur in the solutions actually generated during the search process (whether good or bad); (6) often do not occur in solutions generated.

Class (1) and (2) attributes can be used to support intensification goals by selecting moves to add and drop such attributes, respectively, from solution. Class (3) and (4) attributes combine the elements of intensification and diversification by these same respective strategies. Finally, in reverse, moves that drop class (5) attributes and add class (6) attributes serve to emphasize diversification concerns. The learning approach called target analysis (see Section 3) may be used to define the thresholds implied by terms such as *often* and *good* instead of resorting to arbitrary choices or calibration efforts based on trial and error. Other types of classifications are of course possible, including those that involve conditional relationships. The use of attribute-based memory in tabu search leads naturally to the parallel concept of attribute-based evaluations, as embodied in the foregoing longer term memory strategies.

2.2 Considerations for Candidate List Strategies

Both solution speed and quality can be significantly influenced by the use of appropriate candidate list strategies. Perhaps surprisingly, the importance of such approaches is often overlooked, though they are fundamental to the TS emphasis on making judicious choices. We give examples of a few candidate list strategies that are particularly useful, and that give a basis for understanding the relevant concerns.

As already noted, memory structures to accelerate the updating of move evaluations, and to reduce the effort of finding best and near best moves, are important to support the aggressive character of TS methods. A standard precept for building candidate lists is to identify subsets of *influential* moves, such as a special collection that can be shown to contain at least one move (and preferably more) that is essential in order to reach an improved solution. The concept of influence in TS signals changes in structure or magnitude that are gauged essential to break free of an unproductive trajectory. (The connection of this idea to that of *escape distance* is elaborated in Section 3.1.)

A trivial type of candidate list strategy is to randomly sample from a given collection of moves until enough members are evaluated to give some assurance that the lot contains some decently good choices. Also an elementary level, one of the oldest (but often very useful) candidate list strategies is the *Subdivision Strategy*. This approach decomposes compound moves with the goal of isolating "good components" that are likely to be part of the best moves at the compound level. The motive for this approach is that the components can often be evaluated much more rapidly, and are typically far fewer in number, than the compound moves derived from them.

For example, "swap moves" are commonly composed of "add moves" and "drop moves," as noted earlier. The number of such swap moves generally equals the product of the numbers of their add and drop components, and hence an approach that seeks to evaluate an appreciable fraction of these swap moves can be very time consuming. On the other hand, it is usually easy to isolate a fairly small

number of the "best" add and drop moves, and to restrict attention to the swap moves composed of them. This can often identify a high quality set of moves, in spite of the fact that a swap move evaluation may not be a simple sum of the evaluations of its components. The difference in effort can be appreciable even where the number of components is not large. Improved information may be obtained by sequential evaluations, as where the evaluation of one component is conditional upon the prior (restricted) choice of another.

A somewhat different type of candidate list strategy that includes a number of interesting variants is the *Aspiration Plus* strategy. This approach establishes an aspiration threshold for the quality of move to be selected, based on the history of the search pattern, and examines moves until finding one that satisfies this threshold. At this point, an additional number of moves is examined, equal to a selected value *Plus* (for *Plus* in the interval from 20 to 100 for example), and the best move overall is selected. To assure that neither too few nor too many moves are examined in total, this rule is qualified to require that at least *Min* moves and at most *Max* moves are examined, for chosen values of *Min* and *Max*. (When the upper limit of *Max* moves is reached, before satisfying other conditions, the approach simply selects the best of the moves seen.) The values of *Min* and *Max* can be modified as a function of the number of moves required to meet the threshold.

The aspiration threshold for this approach can be determined in several ways. To illustrate, during a sequence of improving moves, the aspiration may specify that the next move chosen should likewise be improving, at a level based on other recent moves and the current objective value. During a nonimproving sequence the aspiration will typically be lower, but rise toward the improving level as the sequence lengthens. The quality of currently examined moves can shift the threshold, as by encountering moves that significantly surpass or that uniformly fall below the threshold. As an elementary option, the threshold can simply be a function of the quality of the initial *Min* moves examined on the current iteration.

This Aspiration Plus strategy includes many other strategies as special cases. For example, a "first improving" strategy results by setting *Plus* = 0 and directing the aspiration threshold to accept moves that qualify as improving, while ignoring the values of *Min* and *Max*. A slightly more advanced strategy can determine *Min* and *Max* to assure some specified additional number of moves will be examined after first satisfying an improving threshold. In general, in applying the Aspiration Plus strategy, it is important to assure that new moves are examined on each iteration that are not among those just reviewed (as by starting where the previous examination left off).

Another useful candidate list approach is the *Elite Candidate List* strategy. This approach first builds a Master List by examining all (or a relatively large number of) moves, selecting the k best moves encountered, where k is a parameter of the process. (e.g., for k = 15 to 50). (These moves themselves can be identified by another strategy, such as the Subdivision Strategy.) Then at each subsequent iteration, the current best move from the Master List is chosen to be executed, continuing until such a move falls below a given quality threshold, or until a given number of iterations have elapsed. Then a new Master List is constructed and the process repeats.

The evaluation and precise identity of a given move on the list must be appropriately monitored, since one or both may change as result of executing other moves from the list. Such an Elite Candidate List strategy can be advantageously extended by a variant of the Aspiration Plus strategy, allowing some additional number of moves outside the Master List to be examined at each iteration (where those of sufficiently high quality may replace elements of the Master List).

A *Bounded Change* candidate list strategy can also be worth considering, provided an improved solution can be found by restricting the domain of choices so that no solution component changes by more than a limited degree on any step. A bound on this degree, expressed by a distance metric appropriate to the context, is selected large enough to encompass possibilities considered strategically relevant. (The metric may allow large changes along one dimension, but limit the changes

along another so that choices can be reduced and evaluated more quickly.) Such an approach offers particular benefits as part of an intensification strategy based on decomposition, as discussed in Section 1, where the decomposition itself suggests the limits for bounding the changes considered.

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 (1988) and other refinements (beyond the tree search setting) have been suggested by Glover (1989a). 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.

In constructing candidate lists such as the foregoing, we note again that the concept of *move influence* is important to longer term considerations. Thus, for example, evaluation criteria should be periodically modified (especially where no improving moves exist) to encourage moves that create significant structural changes. A limit is required on the number of influential moves allowed in a given interval, and more particularly on their cumulative interacting effects, since moves of high influence can also be mutually incompatible as a foundation for generating solutions of the best quality. These considerations are amplified in Section 3.

2.3 Strategic Oscillation Memory Structures

Strategic oscillation offers an opportunity to make particular use of both short term and long term frequency-based memory. To illustrate, let *A(current_iteration)* denote a zero-one vector whose the component has the value 1 if attribute j is present in the current solution and has the value 0 otherwise. (The vector *A* can be treated "as if" it is the same as the solution vector for zero-one problems, though implicitly it is twice as large, since $x_j = 0$ is a different attribute from $x_j = 1$. This means that rules for operating on the full *A* must be reinterpreted for operating on the condensed form of *A*.) The sum of the A vectors over the most recent *t* iterations provides a simple memory that combines recency and frequency considerations. To maintain the sum requires remembering *A(k)*, for *k* ranging over the last *t* iterations. Then the sum vector *A** can be updated quite easily by the incremental calculation

$$A^* = A^* + A(\text{current\_iteration}) - A(\text{current\_iteration} - t + 1).$$

Associated frequency measures, as noted earlier, should be normalized, in this case by dividing *A** by the value of *t*. A long term form of *A** does not require storing the *A(k)* vectors, but simply keeps a running sum. (*A** can also be maintained by exponential smoothing.)

Such frequency-based memory is useful in strategic oscillation due to the following observation. Instead of using a customary recency-based TS memory at each step of an oscillating pattern, greater flexibility results by disregarding tabu restrictions until reaching the turning point. At this point, assume a choice rule is applied to introduce an attribute that was not contained in any recent solution at the critical level. If this attribute is maintained in the solution by making it tabu to be dropped, then upon eventually reaching the critical level the solution will be different from any seen over the horizon of the last *t* iterations. Thus, instead of updating *A** at each step, the updating is done only for critical level solutions, while simultaneously enhancing the flexibility of making choices.

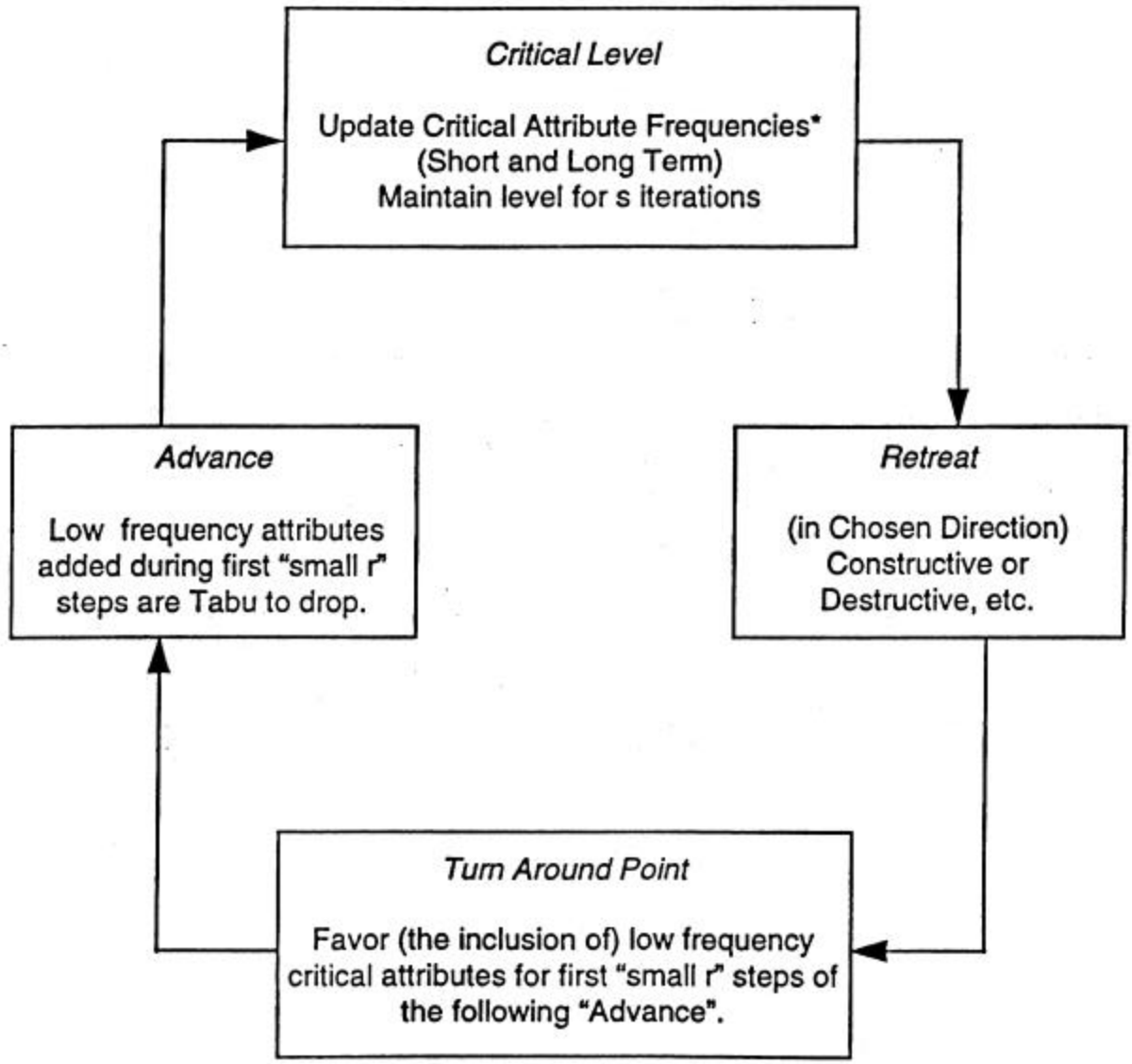
In general, the possibility occurs that no attribute exists that allows this process to be

implemented in the form stated. That is, every attribute may already have a positive associated entry in A^*. Thus, at the turn around point, the rule instead is to choose a move that introduces attributes which are least frequently used. (Note, "infrequently used" can mean either "infrequently present" or "infrequently absent," depending upon the current direction of oscillation.)

For greater diversification, this rule can be applied for r steps after reaching the turn around point. Normally r should be a small number, e.g., with a baseline value of 1 or 2, which is periodically increased in a standard diversification pattern. Shifting from a short term A^* to a long term A^* creates a global diversification effect.

This type of memory has proved remarkably effective for solving multidimensional knapsack and covering problems, especially when using choice rules based on surrogate constraint evaluations (Glover and Kochenberger (1995)). A template for this approach is given in Diagram 10.

Strategic Oscillation - Illustrative Memory



* For selected part of critical level iterations: e.g., for first and best solutions of current block.

Diagram 10

The approach of Diagram 10 is not symmetric. An alternative form of control is to seek immediately to introduce a low frequency attribute upon leaving the critical level, to increase the likelihood that the solution at the next turn around will not duplicate a solution previously visited at that point. Such a control can likewise enhance diversity, though duplication at the turn around will already be inhibited by starting from different solutions at the critical level, and when such duplication nevertheless occurs it may not always be undesirable.

2.4 Path Relinking Considerations

Path relinking strategies in tabu search can occasionally profit by employing different neighborhoods and attribute definitions than used by the heuristics for generating the reference solutions. For example, it is sometimes convenient to use a constructive neighborhood for path relinking as in generating a sequence of jobs to be processed on specified machines. In this case an elite initiating solution can be used to give a *beginning partial construction*, by specifying particular attributes (such as jobs in particular relative or absolute sequence positions) as a basis for remaining constructive steps. Our comments about constructive neighborhoods in this section can also readily be made to apply to destructive neighborhoods, where an initial solution is "overloaded" with attributes donated by the guiding solutions, and such attributes are progressively stripped away or modified until reaching a set with an appropriate composition.

When path relinking is based on constructive neighborhoods, the guiding solution(s) provide the attribute relationships that give options for subsequent stages of construction. At an extreme, a full construction can be produced, by making the initiating solution a *null solution*. (The destructive extreme starts from a "complete set" of solution elements.) Constructive and destructive approaches produce only a single new solution, rather than a sequence of solutions, on each "path" that leads from the initiating solution toward the others. In this case the path will never reach the others unless a transition neighborhood is used to extend the constructive neighborhood.) A characterization of such

processes, and illustrative rules for implementing them, are indicated in Glover (1991).

Constructive neighborhoods can be viewed as a special case of *feasibility restoring* neighborhoods, since a null or partially constructed solution does not satisfy all conditions to qualify as feasible. A variety of methods been devised to restore infeasible solutions to feasibility, as exemplified by flow augmentation methods in network problems, subtour elimination methods in traveling salesman and vehicle routing problems, alternating chain procedures in degree-constrained subgraph problems, and value incrementing and decrementing methods in covering and multidimensional knapsack problems. Using neighborhoods that permit restricted forms of infeasibilities to be generated, and then using associated neighborhoods to remove these infeasibilities, provides a form of path relinking with useful diversification features. Upon further introducing transition neighborhoods, with the ability to generate successive solutions with changed attribute mixes, the mechanism of path relinking also gives a way to *tunnel through* infeasible regions. Application of such processes within a probabilistic TS framework, translating evaluations from deterministic rules into probabilities of selection, offer further opportunities for variation.

A summary of the components of path relinking that embodies these ideas (in abbreviated form) is given in Table 3.

PATH RELINKING SUMMARY

Step 1. Identify the neighborhood structure and associated solution attributes for path relinking (possibly different from those of other TS strategies applied to the problem).

Step 2. Select a collection of two or more reference solutions, and identify which members will serve as the initiating solution and the guiding solution(s). (For a constructive neighborhood, identify the portion of the initiating solution, possibly null, to start the construction.)

Step 3. Move from the initiating solution toward (or beyond) the guiding solution(s), generating one or more intermediate solutions as candidates to initiate subsequent problem solving efforts. (If the first phase of this step creates an infeasible solution, apply an associated second phase with a feasibility restoring neighborhood.)

Table 3

Connections to Other Approaches: Path relinking derives from a population based approach called *scatter search*, which generates new solutions by creating modified linear combinations of the reference points (Glover (1977)). The reference points for scatter search, as for path relinking, consist of elite solutions produced by other search processes, and the best combined solutions are used to re-initiate the processes in a repeating cycle. From one perspective, the modified linear combinations produced by scatter search can be viewed as generating paths in Euclidean vector space. Such a view leads by natural extension to the notion of replacing Euclidean space with neighborhood space, thus giving the basis for the path relinking approach.

By reverse analogy, the solutions produced by path relinking may be viewed as "combinations" of their reference solutions. This provides an interesting connection between proposals of tabu search and proposals of genetic algorithms. In fact, many recently developed "crossover operators" in GA strategies, with no apparent relation between them in the GA setting, can be shown to arise as instances

of path relinking, by restricting attention to two reference points (taken as parents in GAs), and by replacing strategic selection with a reliance on randomization.

Path Relinking Roles in Intensification and Diversification: Path relinking, in common with strategic oscillation, gives a natural foundation for developing intensification and diversification strategies. Intensification strategies in such applications typically choose reference solutions to be elite solutions that lie in a common region or that share common features. Similarly, diversification strategies based on path relinking characteristically select reference solutions that come from different regions or that exhibit contrasting features. Diversification strategies may also place more emphasis on paths that go beyond the reference points. Collections of reference points that embody such conditions can be usefully determined by clustering methods.

These alternative forms of path relinking also offer a convenient basis for parallel processing, contributing to the approaches for incorporating intensification and diversification tradeoffs into the design of parallel solution processes generally.

3. Advanced Solution Capabilities: Fundamental Issues for Improved Implementations.

This section describes concepts and issues that are important for effective application and that merit fuller investigation. We begin by examining the notion of *influence*, followed by considering the generation of compound moves, with particular reference to procedures called *ejection chain strategies*. Then we introduce a series of principles that motivate tabu search strategies in general and that are relevant for designing better solution procedures. Probabilistic tabu search is discussed next, with a sketch of recent findings and potential implications for parallel processing. Finally, we consider the learning approach called *target analysis*, and indicate its uses with tabu search.

3.1 Influence and Measures of Distance and Diversity.

The notion of *influence*, and of *influential moves*, has several dimensions in tabu search. This

notion is particularly relevant upon encountering an *entrenched regionality* phenomenon, where local optima or regions encompassing a particular collection of local optima are "mini black holes" that can be left behind, once visited, only by particularly strong effort. Viewed from a minimization perspective, these regions are marked by the presence of humps which can only be crossed by choosing moves with significantly inferior evaluations, or alternately by the presence of long valleys, where the path to a better solution can only be found by a long (and possibly erratic) climb. In such cases, a faster and more direct withdrawal may be desirable.

A strategy of seeking influential moves, or an *influential series* of moves, becomes important in such situations Glover (1989a, 1990b). The notion of influence does not simply refer to anything that creates a "large change," however, but rather integrates the two key aspects of diversification and intensification in tabu search by seeking change that *holds indication of promise*. This requires reference to memory and/or strategic uses of probabilities while paying careful attention to evaluations.

Diversification in its "pure" form, which solely strives to reach a destination that is markedly different from all others encountered, is incomplete as a basis for an effective search strategy. (It is nevertheless important to characterize how such a pure form would operate, in order to overlay it with balancing considerations of intensification. The essential elements of pure diversification, and their differences from randomization, are discussed in Glover and Laguna (1993).) The notion of influence enters into this by conceiving *influential diversity* to result when a new solution is not only different from (or far from) others seen, but also has a notably attractive structure or objective function value. A variant of this notion has also surfaced more recently in "large step" optimization approaches, though without reference to memory. (See Johnson (1990), Martin et al. (1992), Lourenço (1993).)

From a probability standpoint, solutions that satisfy such requirements of attractiveness are much rarer than those that meet the conditions of pure diversification, and hence in this sense involve a stronger form of diversity (Kelly, Laguna and Glover (1991)). In particular, search spaces commonly

have the property that solutions with progressively better objective function values are distributed with a "diminishing tail," so the likelihood of encountering the better representatives of such solutions is relatively small. Where this is not the case, the problems are generally somewhat easier. A strategy of treating high quality solutions as "improbable" is not a liability in any event. Consequently, the notion of influence focuses on bringing about change that is simultaneously *significant* and *good*.

One way to do this is to create a measure of distance that identifies the magnitude of change in structure or "location" of a solution. Distance can refer to change induced by a single move or by a collection of moves (e.g., viewed as a compound move). Natural measures of distance in different contexts, for example, may refer to weights of elements displaced by a move, costs of elements added or deleted, degrees of smoothness or irregularity created in a pattern, shifts in levels of aggregation or disaggregation, variation in step sizes, alterations in levels of a hierarchy, degrees of satisfying or violating critical constraints, and so forth.

Given a particular distance measure, the tradeoffs between change in distance and change in quality embodied in the notion of influence can be addressed by partitioning distances into different classes. The word "class" is employed to reflect the fact that a measure may encompass more than one of the elements illustrated above, and different combinations invite categorical distinctions. Even where a measure is unidimensional, the effects of different levels of distance may not be proportional to their magnitudes, which again suggests the relevance of differentiation by class.

Under conditions of entrenched regionality, where moves that involve greater distances are likely to involve greater deterioration in solution quality, the goal is to determine when an evaluation for a given distance should in fact be regarded attractive, although superficially such an evaluation may appear less attractive than an evaluation for a smaller distance. Such a determination of relative attractiveness is highly dynamic, since it depends on the extent to which the current solution is affected by the entrenched regionality phenomenon hence, for example, by the distance it has already moved away from a local

optimum. The importance of accounting for the quality of solutions produced when retreating from a local optimum is illustrated by the study of market niche clusters by Kelly (1995). Using a form of strategic oscillation that periodically induces a sequence of steps which progressively degrades the objective function, selecting moves of least degradation was far more effective than selecting moves of greater degradation. Thus, while the notion of influence suggests that moves that create greater changes are to be favored, provided they represent alternatives of comparable quality, it remains important not to be lured by change for the sake of change alone.

Among pitfalls to be avoided, a common mistake made in diversification strategies is to overlook the need for diversifying steps that are *mutually compatible* (and thus which do not propel a solution into an unproductive region). This is typically reflected in the fact that once a large distance move is made, the tradeoffs embodied in selecting influential moves change, so that a higher degree of quality must be demanded of a move of a given distance (or within a given distance class) in order for it to qualify as attractive. Another common mistake is to overlook the phenomenon where some forms of diversifying moves require a series of simpler supporting moves before their effects can be reasonably determined. Often look-ahead analysis is important to exploit this phenomenon, deferring the choice of a diversifying move until such extended effects have been determined for several candidates.

Empirical studies are called for to identify the degree of look-ahead and the number of candidates that should be used in applying such analysis in various settings. A strategy that allows previous solutions to be revisited if a threshold of quality is not soon achieved can serve as an approximate form of look-ahead.

Empirical studies are also called for to identify tradeoffs between quality and distance for particular problem classes and at particular stages of diversification (whether or not look-ahead is used). Recency-based and frequency-based memory can be used to uncover and characterize situations in which evaluations for large distance moves should be preferable to those of smaller distance moves.

The learning approach of target analysis, discussed in Section 3.8, has particular bearing on this issue.

3.2 Compound Moves, Variable Depth and Ejection Chains

The issues of influence, and their relevance for combining the goals of intensification and diversification, are not simply manifested in isolated choices of moves with particular features, but rather in coordinated choices of moves with interlinking properties. The theme of making such coordinated moves leads to consideration of *compound moves*, fabricated from a series of simpler components.

Procedures that incorporate compound moves are often called *variable depth methods* (Papadimitrou and Steiglitz (1982)), based on the fact that the number of components of a compound move generally varies from step to step. One of the simpler approaches, for example, is to generate a string of component moves whose elements (such as edges in a graph or jobs in a schedule) are allowed to be used or "repositioned" only once. Then, when the string cannot be grown any larger, or deteriorates in quality below a certain limit, the best portion of the string (from the start to a selected end point) provides the compound move chosen to be executed. This simple design constitutes the usual conception of a variable depth strategy, but the TS perspective suggests the merit of a somewhat broader view, permitting the string to be generated by a more flexible process. For example, by using TS memory it is possible to avoid the narrowly constrained progression that disallows a particular type of element from being re-used.

Within the class of variable depth procedures, broadly defined, a special subclass called *ejection chain procedures* has recently proved useful. Early forms of ejection chain procedures are illustrated by alternating path methods for matching and degree-constrained problems in graph theory (see, e.g., Berge (1962)). A compound move in this setting, which consists of adding and dropping successive edges in an alternating path, not only has a variable depth but also exhibits another fundamental feature. Some components of the compound move create conditions that must be "resolved" by other components. Accordingly, the move is generated by complementary stages that

introduce certain elements and eject others. One step of the move creates a disturbance (such as violating a node degree by adding an edge) which must be removed by a complementary step (restoring the node balance by dropping an edge).

The theme of such approaches generalizes naturally to a variety of settings more complex then that of adding and dropping edges in graphs. The key principle is that a strategic collection of partial moves generates a critical (or fertile) condition to be exploited by an answering collection of other partial moves. Typically, as in alternating paths, this occurs in stages that trigger the ejection of elements (or allocations, assignments, etc.) and hence reinforces the ejection chain terminology. In such cases, intermediate stages of construction fail to satisfy usual conditions of feasibility, such as fulfilling structural requirements in a graph or resource requirements in a schedule.

A prototypical example of the alteration between a critical condition and a triggered response comes from network flows, in the classical out-of-kilter algorithm (Ford and Fulkerson (1962)). A linked sequence of probing and adjustment steps is executed until achieving a "breakthrough," which triggers a chain of flow changes, and this alternation is repeated until optimality is attained. Another example, again coming from classical methods, is provided by cutting plane procedures for integer programming. In this case the addition of a cutting plane inequality destroys feasibility conditions, which are restored by an answering series of reoptimization steps, carried out in repeated alternation until reaching convergence. (Here, improvements are measured as reductions of duality gaps.) In contrast to the approaches considered here, however, such examples involve macro strategies rather than embedded strategies. More importantly, they do not encompass the freedom of choices for intermediate steps allowed in heuristic procedures. Above all, they do not involve special memory or probabilistic links between successive phases to overcome local optimality conditions when a compound move no longer generates an improvement. (The original characterization of variable depth methods also gave no provision for a means to proceed when a compound move failed to improve the

current solution.)

Within the heuristic setting, ejection chain approaches have recently come to be applied with considerable success in several problem areas, such as generalized assignment, clustering, planar graph problems, traveling salesman problems and vehicle routing. (See, for example, Laguna et al. (1991), Dorndorf and Pesch (1994), Pesch and Glover (1995), Rego and Roucairol (1995).) Such strategies for generating compound moves, coupled with TS processes both to control the construction of the moves and to guide the master procedure that incorporates them, offer a basis for many additional heuristics.

3.3 The Proximate Optimality Principle

The Proximate Optimality Principle (POP), which applies to both simple and compound moves, is the notion that good solutions at one level are likely to be found "close to" good solutions at an adjacent level. (The challenge is to define levels and moves that make this rather loose statement usefully exploitable.) An important part of the idea is the following intuition. In a constructive or destructive process as in generating new starting solutions, or as in applying strategic oscillation it can be highly worthwhile to seek improvements at a given level before going to the next level.

The basis for this intuition is as follows. Moves that involve (or can be interpreted as) passing from one level to another are based chiefly on knowledge about the solution and the level from which the move is initiated, but rely on an inadequate picture of interactions at the new level. Consequently, features can become incorporated into the solution being generated that introduce distortions or undesirable sub-assemblies. Moreover, if these are not rectified they can build on themselves -- since each level sets the stage for the next, i.e., a wrong move at one level changes the identity of moves that look attractive at the next level. Consequently, there will be a tendency to make additional wrong moves, each one reinforcing those made earlier. Eventually, after several levels of such a process, there may be no way to alter earlier improper choices without greatly disrupting the entire construction. As a

result, even the application of an improvement method to the resulting solution may find it very hard to correct for the previous bad decisions.

This supports the idea of applying restarting strategies and strategic oscillation approaches by pausing at periodic intervening levels of construction and destruction, in order to "clean up" the solution at those levels. Such an approach is not limited in application to constructive and destructive processes, of course, but can also be applied to other forms of strategic oscillation. Further, the process of "pausing" at a particular level can consist of performing a tight series of strategic oscillations at this level.

To date, there do not seem to be any studies that have examined this type of approach conscientiously, to answer questions such as: (a) how often (at what levels) should clean up efforts be applied? (b) how much work should be devoted at different levels? (Presumably, if a clean up phase is applied at every level, then less total work may be needed because the result at the start of these levels will already be close to what is desired. On the other hand, the resulting solutions may become "too tightly improved," contrary to the notion of congenial structures discussed in the next section.) (c) how can "attractiveness" be appropriately measured at a given level, since the solution is not yet complete? (d) what memory is useful when repeated re-starts or repeated oscillation waves are used, to help guide the process? (e) what role should probabilities have in these decisions? (f) is it valuable to carry not just one but a collection of several good solutions forward at each step, as in a sequential fan candidate list strategy? (An interesting question arises in a parallel application, related to the sequential fan candidate list strategy: what kinds of diversity among solutions at a given level are desirable as a base for going to the next level?)

Answers to the foregoing questions are relevant for providing improved procedures for problems in scheduling, graph partitioning, maximum weighted cliques, p-median applications and many others. The next sections raise considerations that yield avenues for further improvement.

3.4 The Principle of Congenial Structures

An important supplement to the POP notion is provided by the Principle of Congenial Structures. The key idea is that there often exist particular types of solution structures that provide *greater accessibility to solutions of highest quality* and, as an accompaniment, there also frequently exist special evaluation functions (or "auxiliary objective functions") that can guide a search process to produce solutions with these structures.

This principle is usefully illustrated by an application of tabu search in work force scheduling (Glover and McMillan (1986)), where improved solutions were found by modifying a standard objective function evaluation to include a "smoothing" evaluation. The smoothing evaluation in this case was allowed to dominate during early-to-middle phases of generating starting solutions, and then was gradually phased out. However, the objective function itself was also modified by replacing an original linear formulation with a quadratic formulation (in particular, replacing absolute deviations from targets by squared deviations). The use of quadratic evaluations reinforced the "smoothness" structure in this setting and, contrary to conventional expectation, produced solutions generally better for the linear objective than those obtained when this objective was used as an evaluation function.

A more recent application disclosing the importance of congenial structures occurred in multiprocessor scheduling (Hubscher and Glover (1994)). The notion of a congenial structure in this instance was used to guide phases of *influential diversification*, which made it possible to effectively "unlock" structures that hindered the ability to find better solutions, with the result of ultimately providing improved outcomes.

This issue of appropriately characterizing the nature of congenial structures for different problem settings, and of identifying evaluation functions (and associated procedures) to realize these structures, deserves fuller attention. Specific aspects of this issue are examined next.

Congenial Structures Based on Influence

The influence concept, discussed in Section 3.1, can play an important role in identifying (and

creating) congenial structures. This concept is manifested in a number of settings where solution components can be viewed as falling roughly into two categories, consisting of *foundation components* and *crack fillers*. The crack fillers are those that can relatively easily be handled (such as jobs that are easily assigned good positions or variables that are easily assigned good values) once an appropriate way of treating foundation components has been determined.

Typically, crack fillers represent components of relatively small "sizes," such as elements with small weights in bin packing problems, edges with small lengths in routing problems, jobs with small processing times in scheduling problems, variables with small constraint coefficients in knapsack problems, etc. Hypothetically, an approach that first focuses on creating good (or balanced) assignments of foundation elements, as by biasing moves in favor of those that introduce larger elements into the solution, affords an improved likelihood of generating a congenial structure. For example, among competing exchange moves within a given interval of objective function change, those that involve larger elements (or that bring such elements into the solution), may be considered preferable during phases that seek productive forms of diversity. Such moves tend to establish structures that allow more effective "endgames," which are played by assigning or redistributing the crack fillers. (The periodic endgames arise figuratively in extended search with transition neighborhoods, and arise literally in multistart methods and strategic oscillation.)

Approaches of this type, which provide a simple approximation to methods that seek to characterize congenial structures in more advanced ways, have some appeal due to their relatively straightforward design. For example, at a particularly simple level, if an improving move exists, choices may be restricted to selecting such a move with a greatest level of influence. More generally, a set of thresholds can be introduced, each representing an interval of evaluations. Then a move of greatest influence can be selected from those that lie in the highest nonempty evaluation interval. Such approaches motivate a quest for appropriate thresholds of objective function change versus influence

change, particularly in different regions or phases of search. Studies that establish such thresholds can make a valuable contribution.

Congenial Structures Based on Improving Signatures

Another way to generate congenial structures arises by making use of an *improving signature of a solution*. This approach has particular application to searches that are organized as a series of improving phases that terminate in local optimality, coupled with intervening phases that drive the search to new vantage points from which to initiate such phases. (The improving phases can be as simple as local search procedures, or can consist of tabu search methods that use aspiration criteria to permit each sequence of improving moves to reach a local optimum.)

As a first approximation, we may conceive the improving signature IS(x) of a solution x to be the number of solutions $x' \in N(x)$ that are better than x, i.e., that yield $f(x') > f(x)$ for a maximization objective. We conjecture that, in the process of tracing an improving path from x, the probability of reaching a solution significantly better than x is a function of IS(x). More precisely, the probability of finding a (near) global optimum on an improving path from x is a function of IS(x) and the objective function value f(x). (Our comments are intended to apply to "typical" search spaces, since it is clearly possible to identify spaces where such a relationship does not hold.)

An evident refinement occurs by stipulating that the probability of finding a global optimum depends on the distribution of $f(x')$ as x' ranges over the improving solutions in N(x). Additional refinements result by incorporating deeper look-ahead information, as from a sequential fan candidate list strategy. From a practical standpoint, we stipulate that the definition of the improving signature IS(x) should be based on the level of refinement that is convenient in a given context.

With this practical orientation, the first observation is that N(x) may be too large to allow all its improving solutions to be routinely identified. Consequently, we immediately replace N(x) by a subset C(x) determined by a suitable candidate list strategy, and define IS(x) relative to C(x). If we restrict

C(x) to contain only improving solutions, this requires identifying values (or bounds on) f(x′) for x′ ∈ C(x). Consequently, in such candidate list approaches, knowledge of these values (as well as the size of C(x)) is automatically available as a basis for characterizing IS(x).

We follow the convention that larger values of IS(x) are those we associate with higher probabilities of reaching a global optimum from x. (Hence for example, in a maximization setting, IS(x) may be expressed as an increasing function of f(x) and the size of C(x), or as a weighted sum of the values f(x′) - f(x) for x′ ∈ C(x).) By this design, explicit TS memory can be used to keep a record of solutions with largest IS(x) values, permitting these solutions to be used as a basis for launching additional improving searches. Attributive TS memory (or probabilistic TS rules) can then be applied, as an accompaniment, to induce appropriate variation in the paths examined.

Identifying and exploiting congenial structures by reference to improving signatures has significant potential to benefit from the application of target analysis, discussed in Section 3.8. In addition, approaches based on these notions are directly relevant to the Pyramid Principle, discussed next.

3.5 The Pyramid Principle

A natural goal of search is to maximize the percentage of time devoted to exploring the most profitable terrain. The lack of knowledge about what such terrain consists of leads to *diffused* strategies such as those of simulated annealing, which by design spend large periods of time in unattractive regions, and such as those of random restarting which (also by design) "aimlessly" jump to a new point after each improving phase is completed.

A somewhat different type of strategy is motivated by the Pyramid Principle of improving search paths, which rests on the following observation. Consider an arbitrary improving path from a given starting solution to a local optimum. As the search gets closer to the local optimum, the tributary improving paths to other local optima become fewer, since all such paths at any given level are

contained among those at each level farther from the local optimum.

To formulate this observation and its consequences more precisely, let LO(x) denote the set of local optima that can be reached on the union of all improving paths starting from x. Also let IN(x) the *improving neighborhood* of x, i.e., IN(x) = {x ∈ N(x): f(x) > f(x)} (assuming a maximization objective). Finally, let IP(x) denote the collection of improving paths from x to a local optimum. Then LO(x) is the union of the sets LO(x) for x ∈ IN(x), and IP(x) is the union of the sets IP(x), each augmented by the link from x to x , for x ∈ IN(x). Further,

$$LO(x) \leq \sum(LO(x) : x \in IN(x)),$$

$$IP(x) \geq \sum(LO(x) : x \in IN(x)).$$

The second of these inequalities is strict whenever the first is strict (which usually may be expected), and in general, the number of elements of IP(x) can be greatly larger than that of LO(x), a discrepancy that grows the farther x is from any given local optimum.

The relevant relationships are completed by defining the *length* of an improving path to be the number of its links, the *distance* D(x,x") from x to a local optimum x" to be the length of the longest improving path from x to x" (under conditions where at least one such path exists), and the *level* LEV(x), to be the greatest distance D(x, x") as x" ranges over all local optima accessible by an improving path from x. Then

LEV(x) = 1 + Max (LEV(x): x ∈ IN(x)), and for any given improving path, the value f(x) is strictly decreasing function of LEV(x). In addition, IP(X) is nondecreasing function of LEV(x) and a nonincreasing function of f(x).

The Pyramid Principle then can be expressed by saying that the total number of improving paths decreases as f(x) moves closer to a global optimum. If we view the number of such paths as the width of a band that corresponds to different intervals of f(x) values, the band becomes progressively narrower as f(x) approaches its global maximum, hence roughly resembling the shape of a pyramid.

Adopting the convention that

IP(x) = {x*} for each globally optimal solution x, where x* is a dummy "root" solution for these global optima, the apex of the pyramid consists of the point x*. For many search spaces (such as those with moderate connectivity, and where f(x) takes on multiple values), the rate at which the pyramid narrows as f(x) grows can be dramatic.

Mild assumptions about the structure of improving paths causes this pyramid to generate an analogous pyramidal shape, but inverted, for the probability of finding improving paths to a global optimum as f(x) increases. (The base of the inverted pyramid corresponds to the point where f(x) achieves its maximum value, and the width of this base corresponds to the maximum probability of 1.) Thus, for example, if the size of IN(x) is approximately randomly distributed, or falls randomly within a particular range for each x at any given f(x) value, then the inverted pyramid structure may be expected to emerge. Under such circumstances, the search can be significantly more productive by a strategy that undertakes to "keep close" to the globally optimum value of f(x). (Such a strategy, of course, stands in marked contrast to the strategies of simulated annealing and random restarting.)

The foregoing analysis is somewhat pessimistic, and potentially myopic, for it implicitly supposes no information exists to gauge the merit of any improving move relative to any other (starting from given level of f(x)). Hence, according to its assumptions, all improving moves from a "current" solution x should be given the same evaluation and the same probability of selection. However, it is reasonable to expect that the search space is not so devoid of information, and better strategies can be designed if a sensible means can be identified to extract such information. In particular, the Pyramid Principle is likely to benefit significantly when applied together with the Principle of Congenial Structures. In combination these two principles clearly have implications for the design of parallel processing strategies. Additional considerations relevant to such strategies derive from probabilistic TS implementations, examined next.

3.6 Probabilistic Tabu Search And Parallel Processing Uses

Probabilistic tabu search, which is a direct extension of deterministic tabu search, is based on the principle that appropriately designed probabilities can substitute for certain functions of memory as a means for guiding search (Glover (1989a)). The basic approach can be summarized as follows.

(A) Create move evaluations that include reference to tabu status and other relevant biases from TS strategies using penalties and inducements to modify an underlying "standard" evaluation.

(B) Map these evaluations into positive weights, to obtain probabilities by dividing by the sum of weights. The highest evaluations receive weights that disproportionately favor their selection.

Memory continues to exert a pivotal influence through its role in generating penalties and inducements. However, this influence is modified (and supplemented) by the incorporation of probabilities, in some cases allowing the degree of reliance on such memory to be reduced.

As in other applications of tabu search, the use of an intelligent candidate list strategy to isolate an appropriate subset of moves for consideration is particularly important in the probabilistic TS approach. Although a variety of ways of mapping TS evaluations into probabilities are possible, the following instance of the approach has recently been found to perform quite well.

(1) Select the "r best" moves from the candidate list, for a chosen value of r, and order them from best to worst (where "evaluation ties" are broken randomly).

(2) Assign a probability p to selecting each move as it is encountered in the ordered sequence, stopping as soon as a move is chosen. (Thus, the first move is selected with probability p, the second best with probability (1-p)p, and so forth.) Finally, choose the first move if no other moves are chosen.

The effect of the approach can be illustrated for the choice $p = 1/3$. Except for the small additional probability for choosing move 1, the probabilities for choosing moves 1 through k are

implicitly:

$$1/3, 2/9, 4/27, 8/81, ..., 2^{(k-1)}/3^k.$$

The probability of not choosing one of the first k moves is $(1 - p)^k$, and hence the value p = 1/3 gives a high probability of picking one of the top moves: about .87 for picking one of the top 5 moves, and about .98 for picking one of the top 10 moves.

Experimentation with a TS method for solving 0-1 mixed integer programming problems (Glover and Lokketangen (1994)) has found that values for p close to 1/3, in the range from .3 to .4, appear to work very well. In this application, values less than .3 resulted in choosing "poorer" moves too often, while values greater than. 4 resulted in concentrating too heavily on the moves with highest evaluations. Presumably, basing probabilities on relative differences in evaluations can be important as a general rule, but the simplicity of the ranking approach, which does not depend on any "deep formula," is appealing. (It still can be appropriate, however, to vary the value of p. For example, in procedures where the number of moves available to be evaluated may vary according to the stage of search, the value of p should typically grow as the alternatives diminish. In addition, making p a function of the proximity of an evaluation to a current ideal shows promise of being an effective variant (Xu, Chiu and Glover (1995)).)

Conjectures about why this approach has performed well suggest an interesting possibility. It may be supposed that evaluations have a certain "*noise level*" that causes them to be imperfect so that a "best evaluation" may not correspond to a "best move." Yet the imperfection is not complete, or else there would be no need to consider evaluations at all (except perhaps from a thoroughly local standpoint, keeping in mind that the use of memory takes the evaluations beyond the "local" context). The issue then is to find a way to assign probabilities that appropriately compensates for the noise level.

A potential germ for theory is suggested by the challenge of identifying an ideal assignment of probabilities for an assumed level of noise (appropriately defined). Alternative assumptions about noise

levels may then lead to predictions about expected numbers of evaluations (and moves) required to find an optimal solution under various response scenarios (e.g., as a basis for suggesting how long a method should be allowed to run).

Parallel Implementations

Probabilistic TS has several potential roles in parallel solution approaches, which may be briefly sketched as follows.

(1) The use of probabilities can produce a situation where one processor may get a good solution somewhat earlier than other processors. The information from this solution can be used at once to implement intensification strategies at various levels on other processors to improve their performance. Simple examples consist of taking the solution as a new starting solution for other processors, and of biasing moves of other processors to favor attributes of this solution. (This is a situation where parallel approaches can get better than linear improvement over serial approaches.) In general, just as multiple good solutions can give valuable information for intensification strategies in serial implementations, pools of such solutions assembled from different processors can likewise be taken as the basis for these strategies in parallel environments.

(2) Different processors can apply different probability assignments to embody different types of strategies as where some processors are "highly aggressive" (with probabilities that strongly favor the best evaluations), some are more moderate, and some use varying probabilities. (Probabilities can also be assigned to different choice rules, as in some variants of strategic oscillation.) A solution created from one strategy may be expected to have a somewhat different "structure" than a solution created from another strategy. Thus, allowing a processor to work on a solution created by the contrasting strategy of another processor may yield an implicit diversifying feature that leads to robust outcomes.

(3) Solution efforts are sometimes influenced materially by the initial solutions used to launch them. Embedding probabilistic TS within methods for generating starting solutions allows a range of

initial solutions to be created, and the probabilistic TS choice rules may have a beneficial influence on this range. Similarly, using such rules to generate starting solutions can be the basis for diversification strategies based on restarting, where given processors are allowed to restart after an unproductive period.

At present, only the simplest instances of such ideas have been examined, and many potential applications of probabilistic TS in parallel processing remain to be explored.

3.7 The Space/Time Principle

The Space/Time Principle is based on the observation that the manner in which space is searched should affect the measure of time. This principle depends on the connectivity of neighborhood space, and more precisely on the connectivity of regions that successively become the focus of the search effort.

The idea can be illustrated by considering the hypothetical use of a simple TS approach for the traveling salesman problem, which is restricted to relying on a short term recency-based memory while applying a candidate list strategy that successively looks at different tour segments. (A "segment" may include more that one subpath of the tour and its composition may vary systematically or probabilistically.) In this approach, it may well be that the search activity will stay away from a particular portion of a tour for an extended duration that is, once a move has been made, the search can become focused for a period in regions that lie beyond the *sphere of influence* of that move (i.e., regions that have no effective interaction with the move). Then, when the search comes back to a region within the move's sphere of influence, the tabu tenure associated with the move may have expired! Accordingly, since no changes have occurred in this region, the moves that were blocked by this tabu tenure become available (as if they had never been forbidden). Under such circumstances, the recency-based tabu memory evidently becomes ineffective. It is not hard to see that more complex scenarios can likewise exert an erratic influence on memory, creating effects that similarly distort its

function and decrease its effectiveness.

In problem settings like that of the TSP, where a form of *spatial decomposition* or *loose coupling* may accompany certain natural search strategies, the foregoing observations suggest that measures of time and space should be interrelated. This space/time dependency has two aspects: (1) The clock should only "tick" for a particular solution attribute if changes occur that affect the attribute. (2) On a larger scale, longer term forms of memory are required to bridge events dispersed in time and space. This includes explicit memory that does not simply record best solutions, but also records best "partial" (or regional) solutions.

For example, in the TSP, after obtaining a good local optimum for a tour segment that spans a particular region, the procedure may continue with the outcome of producing a less attractive solution (tour segment) in that region. Then, when improvement is subsequently obtained in another region, the current solution that includes the current partial solution for the first region is not as good as otherwise would be possible. (This graphically shows the defect of considering only short term memory and of ignoring compound attributes.)

This same sort of "loose coupling" has been observed in forestry problems by Lokketangen (1995) who proposes similar policies for handling it. Quite likely this structure is characteristic of many large problems, and gains may be expected by recognizing and taking advantage of it.

3.8 Target Analysis

Target analysis is a learning procedure that can be used to provide more intelligent applications of the foregoing principles. The approach is based on a preliminary study of representative problems from a given class, consisting of integrated phases designed to uncover information to produce improved decisions (Glover (1986), Glover and Greenberg (1989), Laguna and Glover (1993)). In this sense, target analysis is a *global* or *class-based* process of learning and inference. It can also provide a framework for improved applications of *local* or *individual-problem-based* learning approaches,

which seek to adaptively modify their decision rules according to information recorded and processed during the solution of a particular problem. In general, target analysis can be applied in many procedures, such as branch and bound and even simple local search. Its main features may be briefly sketched by viewing the approach as a three-phase procedure, as follows.

Phase 1 of target analysis is devoted to applying currently established methods to determine optimal or near optimal solutions to representative problems from a given class. This phase is straightforward in its execution, although a high level of effort may be expended to assure the solutions are of the specified quality.

Phase 2 is the major phase of the procedure, and can be conceived as divided into three overlapping parts. The first part uses the solutions produced by Phase 1 as targets, which become the focus of a new set of solution passes. During these passes, each problem is solved again, this time scoring all available moves (or a high-ranking subset) on the basis of their ability to progress effectively toward the target solution. (The scoring can be a simple classification, such as "good" or "bad," or it may capture more refined gradations. In the case where multiple best or near best solutions may reasonably qualify as targets, the scores may be based on the target that is "closest to" the current solution.) In some implementations, choices during this phase are biased to select moves that have high scores, thereby leading to a target solution more quickly than the customary choice rules. In other implementations, the method is simply allowed to make its regular moves. In either case, the goal is to generate information during this solution effort which may be useful in inferring the solution scores. That is, the scores provide a basis for creating modified evaluations and more generally, for creating new rules to generate such evaluations in order to more closely match them with the measures that represent "true goodness" (for reaching the targets).

In the case of tabu search intensification strategies such as the elite solution recovery approaches described in Section 1.2, scores can be assigned to parameterized rules for determining the

types of solutions to be saved. For example, such rules may take account of characteristics of clustering and dispersion among elite solutions. In environments where data bases can be maintained of solutions to related problems previously encountered, the scores may be assigned to rules for recovering and exploiting particular instances of these past solutions, and for determining which new solutions will be added to the data bases as additional problems are solved. (The latter step, which is part of the target analysis and not part of a solution effort, is reserved to be performed "off line.") Such an approach is relevant, for example, in applications of linear and nonlinear optimization based on simplex method subroutines, to identify sets of variables to provide crash-basis starting solutions.

In path relinking strategies, scores can be applied to rules for matching initiating solutions with guiding solutions. As with other types of decision rules produced by target analysis, these will preferably include reference to parameters that distinguish different problem instances. The parameter-based rules similarly can be used select initiating and guiding solutions from pre-existing solution pools. Tunneling applications of path relinking, which allow traversal of infeasible regions, and strategic oscillation designs that purposely drive the search into and out of such regions, are natural accompaniments for handling recovered solutions that may be infeasible.

The second part of Phase 2, closely linked with the first part, constructs parameterized functions of the information generated, with the goal of finding values of the parameters to create a *master decision rule*. This rule is designed to choose moves and decision processes that score highly, in order to achieve the goal that underlies the first part of Phase 2. It should be noted that the parameters available for constructing a master decision rule depend on the search method employed. Thus, for example, tabu search may include parameters that embody various elements of recency-based and frequency-based memory, together with measures of influence linked to different classes of attributes or to different regions from which elite solutions have been derived.

The final part of Phase 2 transforms the general design of the master decision rule into a specific

design by applying a model to determine effective values for its parameters. This model can be a simple set of relationships based on intuition, or can be a more rigorous formulation based on mathematics or statistics (such as a goal programming or discriminant analysis model, or even a "connectionist" model based on neural networks).

The components of Phase 2 are not entirely distinct, and may be iterative. On the basis of the outcomes of this phase, the master decision rule becomes the rule that drives the method applied to the current problem of interest. In the case of tabu search, this rule may naturally be evolutionary, i.e., it may use feedback of outcomes obtained during the solution process to modify its parameters for the problem being solved.

Phase 3 concludes the process by applying the master decision rule to the original representative problems and to other problems from the chosen solution class to confirm its merit. The process can be repeated and nested to achieve further refinement.

Target analysis has an additional important function. On the basis of the information generated during its application, and particularly during its confirmation phases, the method produces empirical frequency measures for the probabilities that decisions with high evaluations will lead to an optimal (or near-optimal) solution within a certain number of steps. These decisions are not only at tactical levels but also at strategic levels, such as when to initiate alternative solution phases, and which sources of information to use for guiding these phases (e.g., whether from processes for tracking solution trajectories or for recovering and analyzing solutions). By this means, target analysis can provide inferences concerning expected solution behavior, as a supplement to classical "worst case" complexity analysis. These inferences can aid the practitioner by indicating how long to run a solution method to achieve a solution of desired quality (and with a specified empirical probability).

One of the useful features of target analysis is its capacity for taking advantage of human interaction. The determination of key parameters, and the rules for connecting them, can draw directly

on the insight of the observer as well as on supplementary analytical techniques. The ability to derive inferences from pre-established knowledge of optimal or near optimal solutions, instead of manipulating parameters blindly (without information about the relation of decisions to targeted outcomes), can save significant investment in time and energy. The key, of course, is to coordinate the phases of solution and guided re-solution to obtain knowledge that has the greatest utility. Many potential applications of target analysis exist, and recent applications suggest the approach holds considerable promise for developing improved tactical and strategic decision rules for difficult optimization problems.

3.9 Vocabulary Building

Vocabulary building, in common with scatter search and path relinking, can be interpreted as a strategy for combining solutions. Vocabulary building inherits the scatter search orientation of allowing multiple vectors to be united simultaneously, but is distinguished by a concern with components of solution vectors, rather than with complete solutions.

The vocabulary building process joins elementary solution attributes to yield more complex attributes, thereby effectively representing an approach for creating *solution fragments.* These fragments typically (though not exclusively) represent attribute combinations shared in common by elite solutions. From this standpoint, vocabulary building provides a mechanism for supplementing TS intensification strategies. However, it also provides a means of diversification, by generating large numbers of solution fragments which may be joined in many different ways. The challenge of exploiting these numerous alternatives is effectively handled by applying exact and heuristic procedures for determining the combinations to be generated (Glover and Laguna (1993)).

One of the significant aspects of vocabulary building is that strategies for integrating solution fragments can often be based on a somewhat different type of problem than the one currently under consideration. For example, in the context of traveling salesman and routing problems, a vocabulary building approach can be applied to transform various subtour fragments into complete tours by

specialized shortest path procedures (Glover (1992)). Ejection chain strategies, as discussed in Section 3.2, provide one of the useful ways for generating the fragments to be assembled. A notable benefit of vocabulary building based on solving optimization models is the fact that the optimization can yield combined vectors that dominate exponential numbers of alternatives, an outcome that is sometimes called the *combinatorial leverage phenomenon*.

A recent example of the use of optimization models for vocabulary building occurs in the work of Rochat and Taillard (1995), who use a partitioning model to assemble component tours of a vehicle routing problem into a complete VRP solution. A related application is provided by the work of Kelly and Xu (1995), who use a covering model for assembling components of more general delivery and routing problems. Telecommunication bandwidth packing problems as studied by Ryan and Parker (1994) and Laguna and Glover (1995) offer another significant application, where solution fragments consisting of routed calls can be integrated into a complete solution by a multidimensional knapsack model.

Vocabulary building, in common with other approaches for explicitly and implicitly exploiting memory based designs, raises important strategic considerations whose applications appear to hold significant promise.

4. Conclusion

The practical successes of tabu search have promoted useful research into ways to exploit its underlying ideas more fully. At the same time, many facets of these ideas remain to be explored. The issues of identifying best combinations of short and long term memory and best balances of intensification and diversification strategies still contain many unexamined corners, and some of them undoubtedly harbor important discoveries for developing more powerful solution methods in the future.

There are evident contrasts between TS perspectives and the views currently favored by the artificial intelligence and neural network communities, particularly concerning the role of memory in

search. However, there are also useful complementarities among these views, which raise the possibility of creating systems that integrate their fundamental concerns. Advances are already underway in this realm, with the creation of *tabu training and learning* models (de Werra and Hertz (1989), Beyer and Orgier (1991), Battiti and Tecchioli (1993), Gee and Prager (1994)), *tabu machines* (Chakrapani and Skorin-Kapov (1993), Nemati and Sun (1994)) and *tabu design* procedures (Kelly and Gordon (1994)). The outcomes from this work have shown promising consequences for supplementing customary connectionist models and paradigms as by yielding levels of performance notably superior to that of models based on *Boltzmann machines*, and by yielding processes for modifying network linkages that give more reliable mappings of inputs to outputs.

Recent years have undeniably witnessed significant gains in solving difficult optimization problems, but it must also be acknowledged that a great deal remains to be learned. Research in these areas is full of uncharted and inviting landscapes.

References

Battiti, R. and G. Tecchiolli (1992a). "The Reactive Tabu Search," IRST Technical Report 9303-13, to appear in *ORSA Journal on Computing*.

Battiti, R. and G. Tecchiolli (1992b). "Parallel Biased Search for Combinatorial Optimization: Genetic Algorithms and TABU," *Microprocessors and Microsystems* 16, 351-367.

Battiti, R. and G. Tecchioli (1993). "Training Neural Nets with the Reactive Tabu Search," Technical Report UTM 421, Univ. of Trento, Italy, November.

Berge, C. (1962). *Theory of Graphs and its Applications*, Methuen, London

Beyer, D. and R. Ogier (1991). "Tabu Learning: A Neural Network Search Method for Solving Nonconvex Optimization Problems," *Proceedings of the International Joint Conference on Neural Networks*, IEEE and INNS, Singapore.

Chakrapani, J. and Skorin-Kapov (1991). "Massively Parallel Tabu Search for the Quadratic Assignment Problem," Working Paper Harriman School for Management and Policy, State University of New York at Stony Brook.

Chakrapani, J. and Skorin-Kapov, J. (1993). "Connection Machine Implementation of a Tabu Search Algorithm for the Traveling Salesman Problem," *Journal of Computing and Information Technology* - CIT 1, 1, 29-36.

Crainic, T.G., M. Gendreau, P. Soriano, and M. Toulouse (1993). "A Tabu Search Procedure for Multicommodity Location/Allocation with Balancing Requirements," *Annals of Operations Research*, 41(1-4): 359-383.

Crainic, T.G., M. Toulouse, and M. Gendreau (1993a). "A Study of Synchronous Parallelization Strategies for Tabu Search," Publication 934, Centre de recherche sur les transports, Universite de Montreal, 1993.

Crainic, T.G., M. Toulouse, and M. Gendreau (1993b). "Appraisal of Asynchronous Parallelization Approaches for Tabu Search Algorithms," Publication 935, Centre de recherche sur les transports, Universite de Montreal, 1993.

Dammeyer, F. and S. Voss (1993). "Dynamic Tabu List Management Using the Reverse Elimination Method," *Annals of Operations Research* 41, 31-46.

Daniels, R.L. and J.B. Mazzola (1993). "A Tabu Search Heuristic for the Flexible-Resource Flow Shop Scheduling Problem," *Annals of Operations Research*, Vol. 41, 207-230.

Dell'Amico, M. and M. Trubian (1993). "Applying Tabu Search to the Job-Shop Scheduling Problem," *Annals of Operations Research*, Vol. 41, 231-252.

Dorndorf, U. and E. Pesch (1994). "Fast Clustering Algorithms," *ORSA Journal on Computing* 6,

141-153.

Ford, L.R. and D.R. Fulkerson (1962). *Flows in Networks,* Princeton University Press.

Freville, A. and G. Plateau (1986). "Heuristics and Reduction Methods for Multiple Constraint 0-1 Linear Programming Problems," *European Journal of Operational Research*, 24, 206-215.

Freville, A. and G. Plateau (1982). "Methodes Heuristiques Performantes Pour les Problemes in Variables 0-1 a Plussieurs Constraintes en Inegalite," Publication ANO-91, Universite des Sciences et Techniques de Lille.

Gee, A. H. and R.W. Prager (1994). "Polyhedral Combinatorics and Neural Networks," *Neural Computation 6*, 161-180.

Gendreau, M. A., Hertz, and G. Laporte (1991). "A Tabu Search Heuristic for Vehicle Routing," CRT-777, Centre de Recherche sur les transports, Universite de Montreal, to appear in *Management Science*.

Gendreau, M., P. Soriano, and L. Salvail (1993). "Solving the Maximum Clique Problem Using a Tabu Search Approach," *Annals of Operations Research*, Vol. 41, 385-404.

Glover, F. (1977). "Heuristics for Integer Programming Using Surrogate Constraints," *Decision Sciences*, Vol. 8, No. 1, January, 156-166.

Glover, F. (1986). "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computers and Operations Research*, 13, 533-549.

Glover, F. (1989a). "Tabu Search - Part I," *ORSA Journal on Computing*, 1(3), 190-206.

Glover, F. (1989b). "Candidate List Strategies and Tabu Search," CAAI Research Report, University of Colorado, Boulder, July.

Glover, F. (1990a). "Tabu Search-Part II," *ORSA Journal on Computing*, 2, 4-32.

Glover, F. (1990b). "Tabu Search: A Tutorial," *Interfaces*, Vol. 20, No. 1, 74-94.

Glover, F. (1992). "New Ejection Chain and Alternating Path Methods for Traveling Salesman Problems," Graduate School of Business and Administration, University of Colorado, Boulder, pp. 449-509.

Glover, F. (1994). "Tabu Search for Nonlinear and Parametric Optimization (with Links to Genetic Algorithms)," *Discrete Applied Mathematics*, 49, 231-255.

Glover, F. (1993) "Tabu Thresholding: Improved Search by Nonmonotonic Trajectories," to appear in *ORSA Journal on Computing*.

Glover, F. and H. J. Greenberg (1989). "New approaches for heuristic search: A bilateral linkage with artificial intelligence," *European Journal of Operational Research*, 39, 119-130.

Glover, F. and M. Laguna (1993). "Tabu Search," *Modern Heuristic Techniques for Combinatorial Problems*, C. Reeves, ed., Blackwell Scientific Publishing, 70-141.

Glover, F., M. Laguna, E. Taillard, and D. de Werra, eds. (1993) "*Tabu Search*," special issue of the *Annals of Operations Research*, Vol. 41, J. C. Baltzer.

Glover, F. and A. Lokketangen (1994). "Probabilistic Tabu Search for Zero-One Mixed Integer Programming Problems," University of Colorado, Boulder.

Glover, F. and G. Kochenberger (1995). "Critical Event Tabu Search for Multidimensional Knapsack Problems," University of Colorado, Boulder.

Glover, F., E. Taillard and D. de Werra (1993). "A Users Guide to Tabu Search," *Annals of Operations Research*, Vol. 41, 3-28.

Hansen, P. (1986). "The Steepest Ascent, Mildest Descent Heuristic for Combinatorial Programming," presented at the Congress on Numerical Methods in Combinatorial Optimization, Capri, Italy.

Hansen, P. and B. Jaumard (1990). "Algorithms for the Maximum Satisfiability Problem," *Computing*, Vol. 44, 279-303.

Hansen, P., B. Jaumard, and Da Silva (1992). "Average Linkage Divisive Hierarchical Clustering," to appear in *Journal of Classification*.

Hertz, A. and D. de Werra (1991). "The Tabu Search Metaheuristic: How We Used It," *Annals of Mathematics and Artificial Intelligence*, 1, 111-121.

Hubscher, R. and F. Glover (1994). "Applying Tabu Search with Influential Diversification to Multiprocessor Scheduling," *Computers and Operations Research*, Vol. 21, No. 8, pp. 877-884.

Johnson, D.S. (1990). "Local Optimization and the Traveling Salesman Problem," *In Proc. 17th Colloquium on Automata, Languages and Programming*, pages 446-461. Springer-Verlag.

Kelly, J. P. (1995). "Determination of Market Niches Using Tabu Search-Based Cluster Analysis," Graduate School of Business, University of Colorado, Boulder.

Kelly, J. P., B. L. Golden, A. A. Assad (1993). "Large-Scale Controlled Rounding Using Tabu Search with Strategic Oscillation," *Annals of Operations Research*, Vol. 41, 69-84.

Kelly, J. P. and K. Gordon (1994). "Predicting the Rescheduling of World Debt: A Neural Network-based Approach that Introduces New Construction and Evaluation Techniques." Working Paper, College of Business and Administration, University of Colorado, Boulder, CO 80309.

Kelly, J. P., M. Laguna and F. Glover (1991). "A Study of Diversification Strategies for the Quadratic Assignment Problem," to appear in *Computers and Operations Research*.

Kelly, J. and J. Xu (1995). "Tabu Search and Vocabulary Building for Routing Problems," Graduate School of Business and Administration, University of Colorado at Boulder.

Laguna, M. and F. Glover (1993). "Integrating Target Analysis and Tabu Search for Improved Scheduling Systems," *Expert Systems with Applications*, Vol. 6, 287-297.

Laguna, M., J. P. Kelly, J. L. Gonzalez-Velarde, and F. Glover (1995). "Tabu Search for the Multilevel Generalized Assignment Problem," *European Journal of Operations Research*, 82, 176-189.

Laguna, M. and F. Glover (1993). "Bandwidth Packing: A Tabu Search Approach," *Management Science*, Vol. 39, No. 4, pp. 492-500.

Lokketangen, A. (1995). "Tabu Search for Forestry Problems," University of Molde, Norway.

Lokketangen, A. and F. Glover (1995). "Tabu Search for Zero-One Mixed Integer Programming Problems With Advanced Level Strategies and Learning," University of Colorado, Boulder.

Lourenço, H. (1993). "Local Optimization and The Job-Shop Scheduling Problem," Faculdade de Ciências, Universidade de Lisboa, Portugal.

Martin, O., S.W. Otto, and E. W. Felten (1992). "Large-step markov chains for TSP incorporating local search heuristics," *Operations Research Letters*, 11:219-224.

Moscato, P. (1993). "An Introduction to Population Approaches for Optimization and Hierarchical Objective Functions: A Discussion on the Role of Tabu Search," *Annals of Operations Research*, Vol 41, 85-122.

Moscato, P. and F. Tinetti (1994). "Blending Heuristics with a Population-Based Approach: A "Memetic" Algorithm for the Traveling Salesman Problem," to appear in *Discrete Applied Mathematics*.

Nemati, H. and M. Sun (1994). "A Tabu Machine for Connectionist Methods," Joint National ORSA/TIMS Meeting, Boston, MA.

Nowicki, E. and C. Smutnicki (1993). "A Fast Taboo Search Algorithm for the Job Shop Problem," Report 8/93, Institute of Engineering Cybernetics, Technical University of Wroclaw.

Nowicki, E. and C. Smutnicki (1994). "A Fast Tabu Search Algorithm for the Flow Shop Problem," Institute of Engineering Cybernetics, Technical University of Wroclaw.

Osman, I.H. (1993). "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem," *Annals of Operations Research*, 41:421-451.

Osman, I.H. and N. Christofides (1993). "Capacitated Clustering Problems by Hybrid Simulated Annealing and Tabu Search," Report No. UKC/IMS/OR93/5. Institute of Mathematics and Statistics, University of Kent, Canterbury, UK. Forthcoming in: *International Transactions*

in Operational Research, 1994.

Ow, P.S. and C. Morton (1988). "Filtered Beam Search in Scheduling," *Int. J. Prod. Res.*, Vol. 26, No. 1, 35-62.

Pesch, E. and F. Glover (1995). "TSP Ejection Chains," Graduate School of Business, University of Colorado, Boulder, to appear in *Discrete Applied Mathematics*.

Porto, S.C. and C. Ribeiro (1993). "A Tabu Search Approach to Task Scheduling on Heterogeneous Processors Under Precedence Constraints," Monographia em Ciêcia da Computaç o, No. 03/93, Pontificia Universidade Católica do Rio de Janeiro.

Reeves, C.R. (1993). "Diversification in Genetic Algorthms: Some Connections with Tabu Search," Conventry University, U.K.

Rego, C. and C. Roucairol (1994). "An Efficient Implementation of Ejection Chain Procedures for the Vehicle Routing Problem," Research Report RR-94/44, PRISM Laboratory, University of Versailles.

Rochat, V. and A. Semet (1992). "Tabu Search Approach for Delivering Pet Food and Flour in Switzerland," ORWP 92/9, Departement de Mathematiques, Ecole Polytechnique Federale de Lausanne.

Rochat, Y. and E. Taillard (1995). "Probabilistic Diversification and Intensification in Local Search for Vehicle Routing," Centre de Recherche sur les Transports, Universite de Montreal, to appear in *Journal of Heuristics*.

Ryan, J. and M. Parker (1994) "A Column Generation Algorithm for Bandwidth Packing," *Telecommunications Systems*, 185-195.

Ryan, J., C. Anderson and K. Jones (1993). "A Permutation-Based Tabu Search for Path Assignment," *Annals of Operations Research*, 299-312.

Soriano, P. and M. Gendreau (1993). "Diversification Strategies in Tabu Search Algorithms for the Maximum Clique Problem," Publication #940, Centre de Recherche sur les Transports, Universite de Montreal.

Taillard, E. (1991). "Parallel Tabu Search Technique for the Job Shop Scheduling Problem," Research Report ORWP 91/10, Departement de Mathematiques, Ecole Polytechnique Federale de Lausanne.

Taillard, E. (1993). "Parallel Iterative Search Methods for Vehicle Routing Problems," Networks, Vol. 23, 661-673.

Vaessens, R., E. Aarts and J.K. Lenstra (1994). "Job Shop Scheduling by Local Search," Eindhoven University of Technology, the Netherlands.

Verdejo, V. V., R. M. Cunquero and P. Sarli (1993). "An Application of the Tabu Thresholding

Technique: Minimization of the Number of Arc Crossings in an Acyclic Digraph," Departamento de Estadistica e Investigacion Operative, Universidad de Valencia, Spain.

Voss, S. (1992). "Tabu Search: Applications and Prospects," Technical report, Technische Hochshule Darmstadt, 1992.

Voss, S. (1993). "Solving Quadratic Assignment Problems Using the Reverse Elimination Method," Technische Hochschule Darmstadt, Germany.

Voss, S. (1994). "Concepts for Parallel Tabu Search," Technische Hochschule Dormstadt, Germany.

de Werra, D. and A. Hertz (1989). "Tabu Search Techniques: A Tutorial and an Applications to Neural Networks," *OR Spectrum*, 11, 131-141.

Woodruff, D. L. (1993). "Tabu Search and Chunking," working paper, University of California, Davis.

Woodruff, D.L. and E. Zemel (1993). "Hashing Vectors for Tabu Search," *Annals of Operations Research*, Vol. 41, 123-138.

Xu, J., S. Chiu and F Glover (1995). "Probabilistic Tabu Search for Telecom- munications Network Design," Graduate School of Business, University of Colorado, Boulder.

Xu, J. and J. P. Kelly (1995). "A Robust Network Flow-Based Tabu Search Approach for the Vehicle Routing Problem," Graduate School of Business, University of Colorado, Boulder.

Filename: Tabu Search Fundamental & Uses.doc
Directory: G:\wpc\WPCSTAFF\GLOVER\TABU
Template: C:\Program Files\Microsoft Office\Templates\Normal.dot
Title:
Subject:
Author: Bartley
Keywords:
Comments:
Creation Date: 02/22/00 11:23 AM
Change Number: 51
Last Saved On: 03/28/00 8:33 AM
Last Saved By: Winnie Bartley
Total Editing Time: 520 Minutes
Last Printed On: 04/10/00 11:34 AM
As of Last Complete Printing
Number of Pages: 84
Number of Words: 21,925 (approx.)
Number of Characters: 124,978 (approx.)