Pergamon

Computers Ops Res. Vol. 21, No. 8, pp. 801–822, 1994

0305-0548/94 $7.00+0.00

OPTIMIZATION BY GHOST IMAGE PROCESSES IN NEURAL NETWORKS†

FRED GLOVER‡
US West Chair in Systems Science, Graduate School of Business, Box 419, University of Colorado, Boulder, CO 80309-0419, U.S.A.

Scope and Purpose—Neural networks come in a variety of forms and are "trained" by a variety of strategies. With a few exceptions, these forms and training processes have not produced strongly competitive approaches for optimization problems, when compared to latest methods that have evolved within the optimization field.

This paper proposes a different type of neural network conception based on "ghost image processes". The fundamental idea is to use two reinforcing types of mappings, one operating on trial solutions and one operating on idealized problem representations or target structures (called ghost images). This gives a natural basis for integrating the design and training functions, and provides an effective way to handle a variety of optimization problems that were previously not well suited to be treated by neural networks. The ghost image processes are able to incorporate specialized components to exploit problem structures in specific optimization domains, and to integrate classical optimization and search methods as part of this process. Examples show how ghost image processes can take advantage of optimization problems with diverse characteristics, and preliminary computational tests are reported for multidimensional knapsack problems that demonstrate the promise of these processes.

Abstract—We identify processes for structuring neural networks by reference to two classes of interacting mappings, one generating provisional outcomes ("trial solutions") and the other generating idealized representations, which we call ghost images. These mappings create an evolution both of the provisional outcomes and ghost images, which in turn influence a parallel evolution of the mappings themselves.

The ghost image models may be conceived as a generalization of the self-organizing neural network models of Kohonen. Alternatively, they may be viewed as a generalization of certain relaxation/restriction procedures of mathematical optimization. Hence indirectly they also generalize aspects of penalty based neural models, such as those proposed by Hopfield and Tank. Both avenues of generalization are "context free", without reliance on specialized theory, such as models of perception or mathematical duality.

From a neural network standpoint, the ghost image framework makes it possible to extend previous Kohonen-based optimization approaches to incorporate components beyond a visually oriented frame of reference. This added level of abstraction yields a basis for solving optimization problems expressed entirely in symbolic ("non-visual") mathematical formulations. At the same time it allows penalty function ideas in neural networks to be extended to encompass other concepts springing from a mathematical optimization perspective, including *parametric deformations* and *surrogate contractions*.

This paper demonstrates the efficacy of ghost image processes as a foundation for creating new optimization approaches by providing specific examples of such methods for covering, packing, generalized covering, fixed charge and multidimensional knapsack problems. Preliminary computational results for multidimensional knapsack problems are also presented.

1. INTRODUCTION

As a preliminary to characterizing the ideas of this paper more formally, we describe the orientation that lies behind them. Our approach is founded on the postulate that effective strategies for problem solving derive from the existence of fundamental idealized structures (ghost images), which operate as representational frameworks or target configurations. These structures begin in a state of

†This research is supported in part by the Joint Air Force Office of Scientific Research and Office of Naval Research Contrast No. F49620-90-C-0033, at the University of Colorado.

‡Fred Glover is the US West Chaired Professor in Systems Science at the University of Colorado, Boulder. He has authored or co-authored more than two hundred published articles in the fields of mathematical optimization, computer science and artificial intelligence, with particular emphasis on practical applications in industry and government. In addition to holding editorial posts for journals in the U.S. and abroad, Dr Glover has been featured as a National Visiting Lecturer by the Institute of Management Science and the Operations Research Society of America and has served as a host and lecturer in the US National Academy of Sciences Program of Scientific Exchange. He is co-founder of Optimization Technologies, Inc., Analysis and Research and Computation, Inc., and the nonprofit research organization Decision Analysis and Research Institute.

"imprecise adjustment" that becomes progressively more refined, and serve as a basis for interrelating and evaluating data as a way of achieving intelligent, coordinated response.

Accompanying these special structures is a set of mechanisms (mappings) for establishing new outcomes, such as inferences, behavior patterns and solutions, from the relation of current data and outcomes to the idealized structures. Also included are mechanisms that operate on these same components to produce the successively refined instances of the idealized structure. (At root the two types of mechanisms may be the same—i.e., members of a common pool capable of performing either function—but it is convenient to think of them as different.) Finally, there are processes that allow both types of transforming mechanisms to alter their form. This may be a consequence of preprogrammed evolution, perhaps guided probabilistically, or of applying the mechanisms in different basic combinations.

The *ghost image* terminology, which refers to the idealized structures that provide the raw material of these processes, is intended to convey a link to visually oriented models, and at the same time to suggest a more abstract frame of reference. We draw on the analogy suggested by popular nomenclature, where a ghost image refers to an aura, or halo, describing a distorted boundary which one seeks to bring into conformity with a figure to which it presumably corresponds. In our approach, the distortion of the ghost image is not a defect but a purposeful displacement. The underlying "true image" is not known, but rather is established by the succession of trial images (solutions) that are adjusted in coordinated association with the ghost images.

The ghost image representations are inherently fluid, able to assume many different forms. In various stages of evolution, members of a given class can embody relationships very different from those at other stages. (By this fluidity, a small number of classes can account for a wide range of functions.) We postulate that such processes underlie the creation not only of responses (i.e., solutions—emphasizing a problem solving focus), but also of additional processes. That is, the basic images and transformation mechanisms are recursively linked, leading to more advanced images and mechanisms that may be invoked to handle situations of increasing complexity.

In our application of this framework we do not confine ourselves to manipulating the most fundamental images and mappings, but are willing to incorporate more refined elements that may offer a chance to produce better outcomes. (High level processes, once generated, are considered legitimate raw magerial for producing additional processes and associated outcomes, without reverting to a building-block stage. The important issue is whether the framework for interrelating and exploiting these processes leads to fruitful consequences.)

In contrast with some neural network researchers, we avoid postulating a specific physical ("hardware") organization in which to embody our hypotheses, but wait for discoveries in areas such as neurobiology to suggest useful schemes at this level. We do not have to wait, of course, to design our own procedure to achieve the functions postulated. From this standpoint, we offer these ideas as a basis for generating new approaches to problem solving. Detailed illustrations provided in later sections show the relevance of our framework for achieving these goals, and preliminary computational results are also reported that suggest the promise for practical applications.

2. AN OPTIMIZATION FRAME OF REFERENCE

We are concerned in this paper with the solution of mathematical optimization problems, especially but not exclusively from the domain of combinatorial optimization, which may be expressed broadly in the following form.

$$\text{Minimize } f(x)$$

$$\text{subject to}$$

$$x \in X \subseteq R^n$$

The function $f(x)$ may be linear or nonlinear, and the condition $x \in X$ that defines feasible x vectors may include constraints ranging from inequalities (such as $g(x) \leqslant 0$) to discrete restrictions (such as requiring specified components of x to assume integer values).

We adopt a perspective that emphasizes a neural network orientation toward solving these problems, introducing a type of solution framework into neural networks that previously has been lacking. Our approach has a kinship to the models of Kohonen [1], and can be viewed as a

generalization of them. Alternately, it can be viewed as a generalization of certain relaxation/restriction procedures of mathematical optimization, and in this sense our approach also extends features of penalty function approaches, as represented by the models of Hopfield and Tank [2].

In the neural network context, one of the primary contributions of the proposed framework is to provide a means of introducing structures that go beyond the topologically oriented foundations of Kohonen models and their analogs, as they are currently applied to certain limited classes of optimization problems. The ghost image structures are interrelated by a pair of abstract convoluted mappings, while the mapping in the Kohonen model is one way and specialized. Likewise these structures allow penalty function concepts to be expanded to include notions that we refer to as problem deformation and surrogate contraction. The coordination of their underlying mappings leads to new types of neural network procedures for solving optimization problems, and permits a useful continuity with a variety of previous models, giving a simple way to describe them in a broader context.

3. THE GHOST IMAGE FORMULATION FOR OPTIMIZATION

To develop our approach, consider a coded information structure §, called a ghost image, related to $x \in X$ in the following manner. We posit the existence of mappings Mx and $M\S$ that generate new trial vectors x and ghost images §, denoted x' and §', by the association

$$Mx(\S, x) = x'$$

$$M\S(\S, x') = \S'.$$

More precisely, if i represents an iteration index, then the mappings embody the recursive relationships

$$Mx[\S(i), x(i)] = x(i+1)$$

$$M\S[\S(i), x(i+1)] = \S(i+1).$$

(The index i then increments by 1 and the process repeats.)

We first provide principles that describe the nature of these relationships, and then in following sections provide specific examples to further clarify their operation. (The reader may find it convenient to skim the principles on first examination, proceeding more directly to the examples, and particularly the material beginning in Section 5, to get a sense of where the principles may lead.)

Principle 1. The ghost image provides an implicit "outline" (or "shadow") of an idealized structure or target solution, which is progressively adapted to x (and which x is progressively induced to approach via the mapping Mx). From the standpoint of a neural network conceptualization, the ghost image is a collection of neurons interrelated by a special structure or set of conditions. Alternatively, the ghost image may represent an idealized problem representation (from which an idealized solution may implicitly derive). In this case, the target for x is inherent in the structure of the problem itself.

Principle 2. The ghost image may not embody a perfect representation of desired solution or problem characteristics. The recursive application of the mappings is designed to compensate for imperfection, amending § so that its idealized form moves progressively closer to the "reality" of the current problem structure. Nevertheless, it is important that § initially expresses a collection of idealized properties as faithfully as possible.

Principle 3. The ghost image generally embodies some of the constraining conditions represented by $x \in X$, but possibly not all. (This provides a connection to mathematical relaxations as elaborated subsequently.)

Principle 4. The ghost image often yields a finer grid of solution alternatives than x. In the case where § is manifested as a form of target solution, the vector § may be of a larger dimension than x (with a many-to-one association between components of § and components of x). Alternatively, values permitted to components of § may be a superset of those permitted to components of x (by the restriction $x \in X$). In the case where § is a form of problem representation, the solution vectors for this problem likewise may satisfy the properties of increased dimensionality or admissible

assignment. (Ghost image processes may be integrated with mathematical layering strategies, for example, to operate on multiple copies of selected variables.)

Principle 5. The expanded dimension (or range of value assignments) of § relative to x is subject to management by a process that alters §'s dimension adaptively, rather than adhering to a fixed grid or value set.

Principle 6. As an adjunct of Principles 3, 4 and 5, the trial solutions $x(i)$ generated by progressive mappings may not belong to X, but to corresponding sets $X(i)$ that may be larger than (or inexact approximations of) X. One of the control features of the process is to assure that these sets ultimately lie within X.

Principle 7. The mappings Mx and $M\S$ are evolutionary and time dependant, hence may be represented more accurately as $Mx = Mx(i)$ and $M\S = M\S(i)$. In the context of a Boltzmann (or simulated annealing) evolution, the form of these mappings is given by $Mx[tx(i)]$ and $M\S[t\S(i)]$, where $tx(i)$ and $t\S(i)$ are "temperature" parameters. In a tabu search evolution, the form is instead given by $Mx[Hx(i)]$ and $M\S[H\S(i)]$, where $Hx(i)$ and $H\S(i)$ are "history vectors" associated with the process. (For relevant background on simulated annealing and tabu search, see Johnson *et al.* [3] and Glover and Laguna [4], respectively.)

The inclusion of history in a neural network process (in the manner prescribed by tabu search) introduces a form of memory that does not come from the neural network itself. It may derive from some form of "external" neural network process, which interrelates and aids the performance of other neural network processes, or may represent a non-associative type of memory. (Such a use of history may be viewed as an operation designed to "change the rules of change".) The relevance of this additional memory as an extension of the customary neural network structure is an important implication of the neural network studies using tabu search be de Werra and Hertz [5] and by Chakrapani and Skorin-Kapov [6].

Principle 8. It is generally appropriate to introduce a more restrictive, and computationallly more expensive, mapping Mx^* that produces trial solutions x^* belonging to X. (A guarantee of feasibility may require some constraints to be incorporated as penalty terms in the objective function, thus enabling the condition $x \in X$ to be satisfied by a computationally tractable procedure.) Outcomes from this mapping may be incorporated into the updating of the other mappings.

Principle 9. A basic feature of ghost image processes is a policy of "narrowed focus and incremental change". In particular, the mapping Mx is typically structured to take special account of a subvector $y(i)$ of $x(i)$ in transforming $x(i)$ into $x(i+1)$. Correspondingly, $x(i+1)$ generally differs from $x(i)$ only in the components representing the transformation from $y(i)$ to $y(i+1)$ (or in components immediately affected by this transformation). The mapping Mx may therefore be viewed as consisting of two parts, first choosing the subvector $y(i)$ of $x(i)$, and then producing $x(i+1)$ from a conjunction of $y(i)$ and the ghost image $\S(i)$. Alternately, we may view the choice process and the mapping process as separate, producing a procedure that may be characterized as follows.

Step 0 (Initialization). Set $i = 1$.
Step 1 (Choice). Choose a subvector $y(i)$ of $x(i)$.
Step 2 (Map). Identify

$$x(i+1) = Mx[\S(i), x(i), y(i)]$$
$$\S(i+1) = M\S[\S(i), x(i+1)]$$

Step 3 Increment i: $= i+1$ and return to Step 1 (or terminate by a cutoff rule).

Including $y(i)$ as an additional argument of Mx, in spite of the fact that $y(i)$ already is included in $x(i)$, indicates that this component is treated in a "preferential" manner by the mapping. Analogous to the focus on $y(i)$ within $x(i)$, there typically is a corresponding subvector of § which chiefly determines the change from $\S(i)$ to $\S(i+1)$. Likewise, $y(i+1)$ typically takes a role in $M\S$ resembling the role adopted by $y(i)$ in Mx. (We do not bother to introduce additional notation to depict this.) The choice of $y(i)$ will affect the form of the mapping Mx, as is implicit in the more general formulation where this mapping is conceived to include the choice of $y(i)$ rather than being treated separately. (In simpler cases a rule may be employed that causes y to cycle through components of x in fixed order. This type of approach appears in limited instances in the literature, but we will argue that other choices are preferable.)

Although important, Principle 9 is also "dangerous" if interpreted too narowly or applied in isolation from other considerations. We will see instances, for example, where Mx or $M\S$ may be interpreted as a series of steps, each of which follows the general dictum of this principle, but which in combination produce change on a more global scale. A useful adjunct to this principle is provided by Principle 10, and critical amendments are offered by Principles 11, 12 and 13.

Principle 10. As in all neural network approaches, parallelism is a pervasive feature. This is manifested in ghost image processes in two ways. First, the restricted focus advocated by Principle 9 may be designed to allow independent treatment of multiple segments of a ghost image simultaneously. Second, a collection of separate ghost image processes may be conducted in parallel. This second type of parallelism is relevant both in tabu search and simulated annealing evolutions. Different parameters (embodied in cooling schedules in the case of *SA*) can yield different outcomes, giving parallelism a chance to contribute to greater efficiency. A parallel approach is additionally important in tabu search evolutions, where the history of the processes becomes an active determinant of future trajectories. In this case, parallelism creates an expanded historical frame of reference, permitting the history of each process to be exploited by the whole.

Principle 11. An important counterbalance to Principle 9 is the stipulation that change must not merely be focused and incremental, but must also be "influential and essential". That is, a preferred form of change normally induces a nontrivial stuctural difference (from alternatives available) and involves a subset of options that represent higher degrees of criticality for reaching specified goals. Here, the meaning of "structural difference" is related to that of information content, particularly concerning the relative magnitudes of present and future change (where the depth of the path to a future goal is preferably to be reduced). If certain elements are candidates to change location, for example, then choices should be favored that cause the resulting configuration to yield useful information about future options, and that help to close the gap between the present state and an anticipated terminal state. Further, if a subset of elements can be identified as critical, driving the behavior of others, or requiring a change in at least one of its members (while it is uncertain that this necessity applies to a larger or different subset), then such a subset should be a focus for choice. The significance of the influence concept, and of identifying *spheres of influence*, is discussed in Glover and Laguna [4]. Neural network literature generally acknowledges the merit of incremental and progressive change, and in some cases provides for change that is not incremental, but often overlooks the importance of the notions underlying this amending principle. Without them, the efficacy of adaptive procedures can become greatly reduced. (It is to be acknowledged that a neural network training/learning algorithm is not equivalent to the neural model, and our comments here apply more specifically to such training aspects. In the ghost image processes the model and its evolution are intimately linked.)

Principle 12. The use of a narrow focus advocated in Principle 9 can be defeating (barring a fortunate predetermination of x and §), unless a way is provided to create new elements (e.g., variables) out of existing ones. Certain problem structures can be analysed effectively only if composite elements are generated, and changes in these elements are monitored together with changes in the basic elements defining these problems (such as an initial set of problem variables and constraints). While the issue of dimensionality raised in Principle 4 bears on this matter, we refer more specifically here to the notion of generating new elements that involve "functional aggregates" of others. Thus, the issue is analogous to the creation of new attributes for the attribute-encoded memory structure of tabu search. It also relates to the concept of chunking in psychology (as where a chess master sees entire configurations as single units), or alternately may be viewed as creating a "vocabulary" (where certain words can summarize concepts that otherwise may require sentences or pages to express).

A useful illustration of this idea is given by a basis representation in the simplex method. In this case a restricted focus on changing the value of a single (nonbasic) variable has the effect of simultaneously changing the values of many other (basic) variables. In the present problem solving context, the special type of linking involved in generating useful vocabularies can begin from foundations of correlation and clustering, but clearly are broader. This issue is crucial to permitting the restricted focus notion of Principle 9 to apply with full effectiveness. It also constitutes a significant open research area for those interested in allowing AI mechanisms to approximate human intelligence.

Principle 13. Conditions encountered at certain boundaries or thresholds may induce $M\S$ or Mx to invoke a component that more radically changes the structure of § or x than by the incremental change element of Principle 9. This corresponds to invoking what is called a diversification step in tabu search terminology. Generally, a determining element in defining the conditions that activate such a diversification step is a memory that exploits history. Instead of an evolution by which x and § are progressively attracted toward a mutually harmonious state, the diversification operates as a form of controlled repulsion. (The word "controlled" is an essential qualification. Wildly random steps generally are not an aspect of intelligent behavior. Even repulsion is channeled.)

Conditions that trigger such departures signal a return to a preceding state, although the inclusion of history means that the state is not treated by $M\S$ in the same way as on a previous visit, hence altering the course of subsequent evolution. (The attraction mechanism that causes a return to a preceding state, or more generally causes the recovery of certain attributes of that state, is called *intensification* in tabu search terminology.)

Large-step behavior that launches a process onto a significantly different track is again a form of change not normally endorsed by neural networks. The element of history to determine conditions of repulsion, and channels for it to follow, implies a time dependent feedback loop whose circuitry in ordinary neural networks is undoubtedly subtle (if not alarmingly complex), and is not likely to be "discovered" by normal methods of training. Rather than wait for the invention of neural network circuitry to achieve this effect (or for neurobiologists to discover relevant brain structure not envisioned in present models), we recognize the importance of the effect and incorporate it directly.

4. LINKS TO PREVIOUS NEURAL NETWORK PROCESSES

Before showing how ghost image processes may be applied to solve a range of more general discrete optimization problems, we first indicate special instances of neural network procedures that fall within the preceding framework. These instances arise chiefly by adapting Kohonen types of models to treat graph problems, motivated by the visual image component of graphs (for which the visual orientation of Kohonen neural network models assumes a natural relevance). As noted earlier, our use of the ghost image terminology is intended to suggest an additional level of abstraction, in which visual linkages may usefully be viewed as a subset of broader constructions, not definable by visual concepts as currently conceived.

We will superimpose the ghost image terminology on our description of earlier processes in order to allow observations for extending (and improving) these processes within the present framework. Thus, for example, we will spead of a "ghost image point," or simply an "image point," instead of referring to a "neuron" (which in this case is the corresponding neural network counterpart).

4.1. Ring (closed curve) neural network models for TSPs

From the standpoint of visualization, a ring has a form that naturally embodies the notion of a traveling salesman tour, prompting researchers to apply the visual model of Kohonen to this setting. (See, e.g. Angeniol *et al.* [7]; Durbin and Wilshaw [8]; El Ghaziri [9]; Geraci *et al.* [10].) On a higher conceptual level, a ring may be conceived to express the sequential order constraints of a tour construction, and by this means has the useful feature of automatically satisfying the condition $x \in X$ applicable to a TSP formulation. More particularly, from our present orientation, a ring that begins in approximately circular form is an idealized tour, since a circle is an optimal closed curve in Euclidean space (of minimum length relative to the area enclosed).

This ideal property of course is disturbed least if the ring is perturbed to the least degree necessary to allow it to map onto the data points required to compensate the tour. Thus the ghost image Principle 9 directly endorses the relevance of the Kohonen dictum (derived empirically) that "visual neurons"—in this case elements of the ring—should be assigned to objects closest to them in the visual field. In the Kohonen-based approaches to TSPs, this has led to the strategy of very gradually adjusting the "closest to" relationship, by progressively moving the ring countours to fit the data points, with the outcome of producing a tour that is a reasonably good (and in some cases very good) candidate for a TSP tour.

Conceiving such a strategy within the framework of a ghost image process, there are a number

of possible mappings Mx to generate trial solutions $x(i)$, and mappings $M\S$ to produce adjusted images $\S(i)$, for this problem. This allows us at once to go beyond the neural network methods guided by the Kohonen model. Such methods produce a form of Mx which creates $x(i)$ by mapping each data point onto the element of the ring closest to it. The mapping is a "degenerate" instance of Mx, since it yields $x(i+1)=Mx[\S(i)]$, without reference to $x(i)$ as an argument of Mx.

Until more advanced algorithmic stages are reached, this mapping creates ambiguity in the tour; i.e., different points may map onto the same element of the ring. To compensate for this, the ring is allowed to have more elements (image points) than data points, which may be conceived as an application of Principle 4. This encourages a differentiation that more readily "breaks ties" in such assignments, and also facilitates the gradual adjustment feature of Principle 9. The process is initialized by a ring that is somewhat loosely located within the general region of the data points.

Conjecture 1. Advantages may be gained (applying the ghost image Principle 2) by starting from a closed curve (circle or ellipse, etc.) that is adapted to the data points, e.g., that has a center located at a center of gravity (or foci located at clustered centers of gravity), or externally molds approximately to the convex hull. (The validity of this conjecture and others to be offered is readily testable.)

Conjecture 2. A useful option (again motivated by Principle 2) is to begin with not one but a collection of rings as a ghost image, where each is designed either to circumscribe a cluster or to thread through a clustered layer. Ultimately each ring will break to connect to another.

Conjecture 3. Additional gains may be derived from the explicit mapping orientation of the present framework by introducing more refined alternatives for Mx. Specifically, a mathematical semi-assignment model can be introduced that assigns each data point to exactly one image point of the ring, while allowing each image point to receive at most one data point, to minimize the sum of assigned distances (or of distances raised to a power).

We note that the semi-assignment solution of Conjecture 3, or a simpler heuristic approximation to it, eliminates the ambiguity of assignments in the Kohonen-derived approaches, and further provides a feasible TSP trial solution (in fully dense graphs) by considering the ordering of the data points induced by the mapping. Such a mapping can also be used in a concentric ring construction, with an added rule for breaking and joining rings. In the more computationally expansive form of obtaining an optimal (as opposed to heuristic) solution of the mathematical semi-assignment problem, this may provide a basis for the Mx^* mapping of Principle 8, applied periodically.

The indicated framework provides additional links to (and extensions of) previous Kohonen-based ring models by reference to the approaches of El Ghaziri [9] and Geraci *et al.* [10]. These procedures may be interpreted in the present context as an attempt to coordinate the Mx and $M\S$ mappings by the 'narrowed focus and incremental change" format of Principle 9, employing a rule that selects a subvector of variables representing alternative assignment possibilities for a single data point, and then makes a unique assignment for that point alone. This is done by a Boltzmann type of approach in the procedures cited, although it also can be done by genetic algorithm or tabul search strategies. Control is established by operating on each data point in a fixed sequential order, and then repeating the process until convergence is established.

Conjecture 4. In contrast to examining data points (and associated subvectors) in fixed order, improvements may result by applying adaptive selection rules based on current proximities of data points to image points. This conjecture is based both on findings from tabu search and from the observation of Conjecture 5, indicated subsequently.

It is to be noted that the gradual adjustment feature of Principle 9, which is fundamental to many neural network processes (and also to many mathematical optimization processes, such as "single variable and adjacent value branching" in branch and bound), may seem to support the notion of a slowly cooling temperature, and hence may be responsible for the widespread tendency to apply Boltzmann procedures in conjunction with neural networks. However, the potential relevance of introducing a temperature parameter, and of cooling it slowly, can be interpreted as a derivative implication (not necessarily valid) of a more general Proximate Optimality Principle (POP), which says that good solutions at one stage (defined by criteria more complex than temperature) are generally close to good solutions at an adjacent stage. (See, e.g., Glover *et al.* [11].) The POP notion in fact directly motivates Principles 4, 5 and 9, and also bears on the relevance of Principles 10–13.

If the POP notion is operable in general systems in nature, and is not merely a principle of abstract combinatoric systems, then it will be applicable to behavior and organization of the visual cortex in the brain. If so, it provides a direct physical rationale for Kohonen's empirical observations. It further suggests the merit of a more refined definition of stage than embodied in a temperature parameter, and of a more refined definition of "close to" than embodied in Euclidean distance. These definitions necessarily must be implicit in the mappings Mx and $M\S$ (and can serve to motivate their construction in specific settings). Hence the use of alternative mappings and of rules for progressively amending them over time provide a basis for testing the relevance of the POP conception.

Conjecture 5. The notion of adaptive dimensionality expressed in Principle 5 offers gains through a strategy of refining a ghost image by a mapping $M\S$ that is not limited to relocating its points (in the neural network "TSP ring" approaches). Rather, the mapping may additionally alter the ghost image (ring) dimension to introduce new image points. Specifically, by considering trial points on the line segment joining two existing image points, and introducing a trial point as a new image point if it lies closer to a member of a set S of nearby data points than the two parents, the method may be expected to perform more effectively than the neural network approaches in the literature. Indeed, it is possible to specify a means of doing this (employing an approxpirate definition of S) that never alters the assignment of an image point to a data point, once made, and that produces a tour in the Euclidean plane with no crossings.

4.2. Vehicle routing (multiple salesman) problems

A first departure from the simple visual orientation that underlies the adaptation of Kohonen neural networks models to TSPs occurs in the approach proposed by El Ghaziri [12] for the VRP problem. Briefly summarized, the VRP problem requires the creation of multiple TSP tours, each initiated from a common depot. The data points represent cities, each with a specified demand (quantity of some item) to be collected by a vehicle traveling the route to which that city is assigned. Each route can contain only a collection of cities whose combined demands do not exceed the carrying capacity of the vehicle that will travel the route. The El Ghaziri approach to the VRP problem can be conveniently expressed in the ghost image framework as follows.

Initially, § consists of a collection of r rings, where r is the number of routes to be created. The starting location of these rings is not considered of great importance. (However, as an analog of Conjecture 1, it seems reasonable to hypothesize that careful testing will discover a different viewpoint is warranted. Specifically, from this perspective, better results appear likely by an initialization that generates r clusters of points, based on distance and capacity, and locates each initial ring as a distorted circle centered on the cluster center but given a tail that connects to the depot.) As in the "TSP ring" approach, data points (cities) are examined in fixed sequential order. (Again, as a counterpart of Conjectures 3 and 5, a superior choice strategy is anticipated to exist.)

The new element introduced by El Ghaziri is that the rings are no longer simple geometric or visual models, but contain components corresponding to the capacities of the tours (the associated vehicles) and to the "current weight" of a ring. The weight may be interpreted in the present setting as the result of the mapping Mx, which creates a trial solution assignment of data points of image points of the rings. A ring thus receives a weight equal to the sum of the demands of data points mapped onto it by Mx. This mapping likewise is a greedy one-at-a-time function, but instead of assigning the currently considered data point based only on its distance to image points, the evaluation includes reference to the current weights of the rings. Infeasible weight assignments are given lower probabilities of selection than feasible ones (using a ratio function).

The interpretation of this approach in the ghost image framework invites consideration of more general possibilities for the mapping Mx. In particular, the type of mapping embodied in a generalized network assignment model seems usefully relevant. Although theoretically NP hard, generalized assignment models have been treated very effectively by heuristics (see, e.g., Laguna *et al.* [13]), and yield solutions respecting capacity limitations as well as distance (cost) considerations. A generalized assignment solution can be the basis for generating feasible VRP solutions, and hence can operate in the role of Mx^*. In addition, such a solution may provide a basis for an improved determination of the $M\S$ mapping.

5. NEW GHOST IMAGE PROCESSES

We now consider how ghost image processes can be used to characterize methods that are fundamentally different from those previously described. The first class of these methods we consider is based on creating an idealized representation of problem structures by a strategy of parametric deformation. We refer to two basic kinds of parametric deformation processes: closed and open. A *closed* process has clearly specified (natural) points of origin and termination, while an *open* process lacks this feature.

5.1. Closed parametric deformations

The strategy of the closed deformation processes begins by modifying the problem parameters in a way that permits an optimal or near optimal solution to be readily determined. This idealized parameterization constitutes the initial ghost image § in this class of approaches. Progressive modification produced by the mapping M§ gradually transforms the idealized parameter values into the actual parameter values for the problem to be solved. At each step, the mapping Mx produces a solution which is near optimal for the problem encoded in §, and this solution becomes a basis for progressing to a new § and trial solution at the next stage.

We clarify the process by a series of examples that provide a more complete frame of reference.

5.1.1. Covering problem. We define the covering problem notationally by

$$\text{Minimize } cx$$

$$\text{subject to}$$

$$Ax \geqslant e$$

$$x \text{ is 0–1 (a vector of zero–one variables)}$$

where, for this formulation, we assume e is a column vector of 1s, and each column of A is a vector of 0s and 1s. The vector c is assumed nonnegative.

There are several ways to create closed parametric deformations for this problem. Let $P(A)$ denote the problem for a specified matrix A.

Deformation I. Replace A by a matrix A' consisting of all 1s (i.e., each column of A' is e). An optimal solution to the deformed (idealized) problem $P(A')$ results by setting $x_j = 1$ for the component x_j of x such that the component c_j of c is minimum. Now the matrix A' is gradually transformed to become closer to A. A simple way to do this is to define

$$A'' = A' + \lambda(A - A')$$

where λ is a scalar weight chosen to make A'' marginally different from A' ($\lambda = 1$ yields $A'' = A$). e.g., λ May be computed (approximately) to be the least positive value such that a current locally optimal solution to $P(A')$ is not locally optimal for $P(A'')$. (The meaning of "locally optimal" depends on the mapping Mx. For example, Mx may successively change the value of chosen variables from 0 to 1, or may swap the 0–1 value assignment of chosen variables, in moving from the current x for $P(A')$ to the new x for $P(A'')$.) Then A'' is redefined to be A' and the process repeats.

The problem $P(A')$ at a given interation i is in fact the ghost image §(i). The definition of A'' as a function of A' and λ (whose chosen value may depend on the solution to $P(A')$ as indicated), identifies the mapping §($i+1$) = M§[§(i), $x(i+1)$]. In this case $x(i+1)$ represents the trial solution generated for §(i), hence $P(A')$, by the mapping $x(i+1) = Mx$[§(i), $x(i)$].

Deformation II. This is the same as Deformation I, except that only subsets of columns of A' are changed to create A''. (An extreme version of Deformation II changes exactly one column of A' and selects $\lambda = 1$; hence immediately updating this column to be a column of A. This resembles the pattern of a dynamic programming approach that recursively introduces one new variable at each stage.)

Additional Deformations. "Disctete" parameterizations may be used where the transformation of A' to A'' consists of identifying selected elements A'_{ij} of A' and assigning these elements the value A_{ij}. For example, the transformation may choose a row of A' and update all elements of this row to the form of A. This is the "dual" of the extreme version of Deformation II. It resembles a successive restriction approach as sometimes used in math programming, which repeatedly imposes

a single new constraint upon resolving the problem containing previously imposed consraints. An approach that selects subsets of row or column elements to undergo change creates other types of strategies. Parameterizations of c and e of course are also possible.

5.1.2. Packing problem. We define the packing problem to be

$$\text{Maximize } cx$$

$$\text{subject to}$$

$$Ax \leqslant e$$

$$x \text{ is } 0\text{–}1$$

where A, c and e are as identified for the covering problem. An idealized starting point for this problem may be created either by setting all entries of A' to 1, as for the covering problem, or instead by setting all entries to 0. (This latter approach has a counterpart for the covering problem only if we allow e to be parameterized along with A.) In either case, an initial optimal starting solution is immediately identifiable.

The complementary directionality of moving toward A from an "all 0" or an "all 1" start invites consideration of the strategic oscillation approach as applied in tabu search. In a restricted form, strategic oscillation may be applied by reference only to one of these selected starting points. For examle, from either start, after a series of parameter changes that ultimately results in $A' = A$, the direction of these changes may be reversed to moe back toward the starting point. Such changes are repeated for a chosen number of steps, then the direction is again reversed to allow the process to return to A. Choosing good steps in each direction makes it possible to visit previous levels from different vantage points, generating different solutions, than coming from the opposite direction. (Non-duplication of solutions can be reinforced by other elements of tabu search.) Such an approach is applicable to the previously indicated parameterization of the covering problem as well.

In the more general case, strategic oscillation can be allowed to "cross the line" between the two approaches defined by the "all 0" and "all 1" starting points. Specifically, upon reaching A from the "all 1" direction, the method can proceed in the "all 0" direction, rather than reversing to return toward the "all 1" start. The distance that A' is permitted to recede from A, once attained, may not need to be large if empirical evidence from tabu search applications in other settings carries over to the processes described here.

5.1.3. Multidimensional knapsack problems. Multidimensional knapsack problems constitute a generalization of packing problems, formulated as follows.

$$\text{Maximize } cx$$

$$\text{subject to}$$

$$Ax \leqslant b$$

$$x \text{ is } 0\text{–}1$$

Here A is a general nonnegative matrix and b is a nonnegative vector. Typically we assume all data elements are integers. As before, c is nonnegative. (Other versions of the problem allow bounded integer variables, or imitate the covering formulation using a minimization objective and $Ax \geqslant b$.)

Although it can be appropriate to parameterize both b and c (as well as A) for this problem, it also remains reasonable to consider simple parameterizations that affect only A, as an extension of the two types of parameterization approaches discussed for packing problems. Specifically, in the first approach, start with each column of A' equal to b (which we may assume has all components at least as large as those of the corresponding column of A, else the column may be dropped). An optimal solution to the resulting problem $P(A')$ is given by setting $x_j = 1$ for the largest c_j. (The progressive transformation based on $A'' = A' + \lambda(A - A')$ can then be used, for example.) In the second approach, define the initial A' to be all 0 (again permitting the same form of progressive transformation). In both cases, strategic oscillation can be applied exactly as indicated for the packing problem.

For the present problem, where some elements of A' must move farther than others to reach A, it may be appropriate to use a mapping to create A'' that employs different w values for different

entires of A, allowing larger distances between A' and A to be covered more slowly than smaller ones. Such an approach should be accompanied by normalizing the constraints (e.g., dividing the row i by b_i or by the sum of the A_{ij} values of j, etc.).

5.1.4. Graph problems. Ideas similar to the preceding can be used to define closed parametric deformations for ghost image processes applied to a variety of graph problems. For example, weights and/or connectivity matrices in graph partitioning, matching, TSPs, etc., can be progressively amended by analogous procedures. (This can further be done in complementary ways, as by proceeding from large-to-small and from small-to-large, or from dense-to-sparse and from sparse-to-dense, etc.) Identifying deformations with special features, or demonstrating that one type is more valuable than another, is a topic that has some appeal.

5.2. Open parametric deformations

We now consider a type of parametric deformation for which no "natural" ending point is identifiable in advance. The exploitable characteristic of these deformations is an ability to specify a range of parameters that is assured to include an optimal set of values. This corresponds to identifying a range of ghost images, consisting of problem representations derived from these parameters. More precisely, for the classes of problems we consider, there exists a specifiable mapping Mx^* (from Principle 8), which is a function of the ghost image § alone, yielding a trial solution $x^* = Mx^*(\S)$ such that x^* not only belongs to X but is optimal for the problem of interest (when § is suitably chosen). There are no "evidently correct" mappings $M\S$ for updating $\S(i+1) = M\S[\S(i), x(i+1)]$, but updating guidelines may be determined by reference to local information. Again, to make the ideas clear, we proceed by example.

5.2.1. Fixed charge problem. The fixed charge problem, which has many well known applications in mathematical optimization, may be formulated as follows.

$$\begin{aligned} \text{FC:}\quad & \text{Minimize} \quad cs + dy \\ & \text{subject to} \\ & \qquad Ax = b \\ & \qquad x \geqslant 0 \\ & \qquad y \text{ is } 0\text{–}1 \\ & \qquad x_j \leqslant U_j y_j \text{ for each } j \in J \end{aligned}$$

Here J is the common index set for components of x and y, x is a vector of continuous variables, d is a vector of nonnegative "fixed charges", and $U = (U_j)$ is an upper bound vector where $x \leqslant U$ is directly or indirectly implied by $Ax = b$ and $x \geqslant 0$. ($x \leqslant U$ is also redundantly a consequence of $x_j \leqslant U_j y_j$ for $j \in J$.) The implication of the formulation is that $x_j > 0$ compels $y_j = 1$ (to satisfy $x_j \leqslant U_j y_j$), hence a positive valued x_j incurs the fixed charge d_j as well as a linear ("variable") cost c_j. Not all variables x_j may incur fixed charges, which corresponds to allowing $d_j = 0$ (and the variable y_j effectively may be dropped).

A straightforward nonlinear programming formulation is given by

$$\text{NFC:}\quad \text{Minimize } [cx + D(x):\ Ax = b,\ x \geqslant 0]$$

where $D(x)$ is the nonlinear function $D(x) = \sum (d_j:\ x_j > 0)$. NFC and its objective function $cx + D(x)$ play a useful role in the following development.

A least cost continuous solution to FC, where the y_j variables may take any value from 0 to 1, evidently yields $y_j = x_j / U_j$ (the smallest possible value for y_j). This motivates the classic approach of solving a relaxation of the fixed charge problem as a continuous linear program (LP), which introduces a substitution of variables to eliminate the y variables and give each x_j a cost of $c_j + d_j / U_j$.

There also evidently must exist some adjusted vector d^* which may replace d, with the effect that an optimal LP solution to the altered problem will be an optimal mixed-integer solution for the original problem (that is, each y_j will receive an optimal integer value). Specifically, if y^* represents an optimal 0–1 vector y, it suffices to choose d_j^* "large" if $y_j^* = 0$. (A range of values for d^* will work, and of course the LP objective function value will require adjustment to match the

correct value for the fixed charge problem, where the latter is given directly by $cx+D(x)$ of the NFC formulation. The same observation shows an optimal d^* exists for any mixed 0–1 ILP problem, by allowing d_j^* to be selected sufficiently negative if $y_j^*=1$.)

Consequently, we are motivated to create an open parametric deformation strategy for the fixed charge problem by manipulating a vector d^* in place of d, with the aspiration of eventually obtaining a d^* that is optimal. Considering again the form of the LP relaxation, which drops the y variables and replaces c_j by c_j+d_j/U_j, we can achieve the same result by replacing U_j with a value v_j, that is, giving x_j a coefficient of c_j+d_j/v_j, and then manipulate v_j instead. A guideline for changing v_j is the knowledge that a stronger LP relaxation will result if v_j is assigned an optimal value for x_j in a fixed charge solution. Thus, for this purpose, we define the parameterized (ghost image) relaxation based on v by

$$\text{FC(v):}\quad \text{Minimize } [(c+d/v)x\colon Ax=b,\ x\geqslant 0]$$

By convention we define the vector d/v to have components d_j/v_j. The form of our method is then as follows, noting that the determination of § [the representation FC(v)] corresponds to identifying v.

5.2.2. Outline of a fixed charge solution method.

Step 0 (Create an initial §). Set $v=U$.
Step 1 (Apply Mx). Solve the LP problem FC(v), yielding a solution x'.
Step 2 (Apply Mx^*). Starting from x', obtain an improved solution x^*.
Step 3 (Apply M§). Update v as a function of its current value and x' or x^*. Then return to Step 1.

The process is terminated after running for a selected number of iterations. To fully characterize the method, we must identify Mx^* for Step 2 and M§ for Step 3.

First we consider M§, the update of v. Let λ denote a scalar that may take values in the interval from 0 to 1, and let v_j' denote the new value of v_j to be determined in Step 3. (v_j' will depend on x_j' from Step 1, as M§ is a function of x as well as §.) We examine two simple cases.

5.2.3. Determining M§ (update of v').

Case 1. $v_j'=\lambda U_j+(1-\lambda)x_j'$.

Here, smaller values of λ shift v_j' closer to the LP solution value x_j' last obtained. Setting $\lambda=0$ yields $v_j'=x_j'$. This causes d_j/v_j' to be "infinity" if $x_j'=0$, hence compelling any variable that equals 0 in the LP solution to FC(v) also to equal 0 in the solution to FC(v') (when v' becomes the new v on the next iteration). A convergence strategy may therefore reasonably begin with λ in the vicinity of 1 and gradually reduce it to 0. If the changes in λ are monotonic, this is analogous to applying a simulated annealing cooling strategy. Alternately, the changes can be patterned to emulate a strategic oscillation approach by manipulating λ to move in both directions around "good" values encountered during the process.

Case 2. $v_j'=\lambda v_j+(1-\lambda)x_j'$.

In this instance, v_j is reassigned the value v_j' after each update, and the process is a form of exponential smoothing, modified by the manipulation of λ. The comments about λ in Case 1 also are relevant to this manipulation.

An immediate option for these two cases is to replace x_j' in the determination of v_j' by the improved trial solution value generated from Mx^*. Before examining the form of Mx^*, we first note how the preceding updates of v_j' can be improved.

A limitation of the rules for updating v_j' in Cases 1 and 2 is a tendency to reinforce characteristics of each succeeding x' too strongly. A probabilistic variant less subject to this limitation is to select the new v_j from a distribution with mean equal to the value for v_j' specified above, gradually diminishing the variance. A somewhat stronger approach, however, is to adopt a diversification strategy (again motivated by ideas from tabu search). When the trial solutions cease to show variation, a diversification step may be introduced with the goal of countering the influence of the last v' and x'. We accomplish this by adding a Step 3A as an adjunct to Step 3 in the Outline Method. The purpose of this step is to encourage the x_j' whose values are 0 to become positive, while decreasing the incentive for positive x_j' to remain positive. (It is probably not prudent to push them too strongly toward 0.)

Step 3A (Diversifying M§). When x' adopts a repeating pattern, reassign each v'_j the value:

$$0.7U_j \quad \text{if } x'_j = 0$$
$$0.3U_j \quad \text{if } x'_j > 0.$$

The foregoing evidently represents a very simple form for Step 3A. A more sophisticated version would make fuller reference to the value of x'_j in determining v'_j. Diversification based on a tabu search orientation would further incorporate history to induce v' vectors generated on different executions of Step 3A to differ significantly from each other (as by a rule approximating the criterion of maximizing the minimum distance from previous vectors, where each v'_j is given dichotomous values such as indicated above.)

Finally, we consider the mapping Mx^* of Step 2, whose goal is to obtain a trial solution x^* better than x'. We achieve this in two phases. Determining Mx^* (identifying an improved trial solution x^*).

Phase 1. By reference to x' from Step 1, identify the LP problem:

$$\text{FC}(x'): \quad \text{Minimize } (c'x: Ax = b, x \geqslant 0)$$

where $c'_j = c_j$ if $x'_j > 0$, and c'_j is "large" if $x'_j = 0$. Let x^* be an optimal LP solution to FC(x') and identify its "true FC value" $cx^* + D(x^*)$ [always as good or better than $cx' + D(x')$].

Phase 2. Consider the LP optimal basis representation yielding x^*, but restore the current objective function in this representation to the form appropriate to FC, replacing c' by the original c and adding the nonlinear component $D(x)$:

$$\text{NFC}(x^*): \quad \text{Mimimize } (c^0 + rx + D(x): Rx = f, x \geqslant 0)$$

Here $c^0 + rx$ is the result of expressing cx in the form determined by the current basis representation of x^*, where c^0 is a constant, and r is the reduced cost vector for this representation. The assignment $x = x^*$ is the current basic solution implicit in the form of $Rx = f$ (hence f identifies the values of all basic components of x^*, while $rx^* = 0$ and $c^0 = cx^*$). This problem is equivalent to FC and NFC.

(2A). Consider the primal feasible pivot steps that introduce nonbasic x_j variables (currently 0) into the basis, each causing an associated x_k to leave the basis. Let z_{jk} be the LP objective change for the pivot (equal to $r_j x^*_k / R(_{kj})$, if x is indexed so that $x^*_k = f_k$ for all basic variables x_k). Then the pivot that exchanges x_j and x_k changes the true objective value for NFC by adding the amount $z_{jk} + d_j - d_k$. (Exception for primal degeneracy: if more than one x_k is driven to 0, subtract d_k for each; while if $x^*_k = 0$, the total change is 0.)

(2B). Select and perform a pivot that yields a best (negative) improving change, as identified in (2A). Designate the resulting solution to be the new x^*, and then repeat Phase 2 until no improving pivots remain.

The application of Mx^* in this two phased approach, while based on a sequence of "small" (single exchange) pivot steps, can make a significant overall difference in the quality of trial solutions generated, by contrast to the initial LP solution step embodied in Mx of Step 1. Still better trial solutions can be obtained by extending Mx^* to incorporate a tabu search component in (2B), permitting the process to continue when no improving pivots are found (e.g., using simple tabu lists governing entry into and departure from the basis). For large problems, the number of pivot options to evaluate may usefully be reduced by means of candidate lists.

5.2.4. General mixed 0—1 ILP problems. We formulate the general mixed 0–1 ILP problem as

$$\begin{aligned} &\text{Minimize} && cx + dy \\ &\text{subject to} && \\ & && Ax + By = b \\ & && x \geqslant 0 \\ & && y \text{ is } 0\text{–}1 \end{aligned}$$

As previously noted, there exists a d^* that causes an optimal LP solution with d^* replacing d to be optimal for the 0–1 ILP problem. The ideas for manipulating the objective function and generating

improved solutions in the context of the fixed charge problem can be extended to this present more general problem, making additional use of LP postoptimality information in determining parameter updates, and using integer feasibility measures in pivoting processes. (A fuller discussion of such an approach for the mixed 0–1 ILP problem will be given elsewhere.)

6. GHOST IMAGE PROCESSES AND MATHEMATICAL DUALITY METHODS

Ghost image methods give a framework that similarly encompasses a variety of mathematical duality methods, particularly those constituting relaxation/restriction approaches in discrete optimization. Lagrangean and surrogate constraint relaxation methods are notable examples. In this case, Mx represents a mapping that generates a trial solution for a primal problem, while $M\S$ represents a mapping that updates the dual problem (as by subgradient optimization).

Beyond the convenience of providing a common notational and descriptive foundation that links such procedures to others of a substantially different nature, the ghost image framework prompts the formulation of alternative ways to exploit mathematical duality concepts. The following identifies a new discrete optimization procedure based on ghost image principles that take advantage of surrogate constraint relaxations. The procedure, which we call a "surrogate contraction" method, is specifically designed for solving classes of 0–1 covering, packing and multidimensional knapsach problems.

6.1. A surrogate contraction approach

We will focus on the following two classes of problems, defined relative to a nonnegative matrix A, and nonnegative vectors b and c.

$$\begin{aligned} &\text{(GC)} \quad \text{Minimize } cx \\ &\qquad \text{subject to} \\ &\qquad\qquad Ax \geqslant b \\ &\qquad\qquad x \text{ is } 0\text{–}1 \end{aligned}$$

$$\begin{aligned} &\text{(MK)} \quad \text{Maximize } cx \\ &\qquad \text{subject to} \\ &\qquad\qquad Ax \leqslant b \\ &\qquad\qquad x \text{ is } 0\text{–}1 \end{aligned}$$

The problem (GC) is a generalized covering problem and the problem (MK) is the multidimensional knapsack problem previously discussed. The method we now describe applies to surrogate constraint procedure to both problems using mappings Mx and $M\S$ that are nested.

A surrogate constraint (Glover [14]) consists of a nonnegative linear combination of the original problem constraints. Let w denote a nonnegative row vector, and define $a^0 = wA$ and $b^0 = wb$. Then a surrogate constraint for each of the foregoing problems may be expressed respectively as

$$a^0x \geqslant b^0 \quad \text{and} \quad a^0x \leqslant b^0.$$

These surrogate constraints give rise to associated surrogate problems (S-GC) and (S-MK) which are relaxations of the original problems (GC) and (MK):

$$\begin{aligned} &\text{(S-GC)} \quad \text{Minimize } cx \\ &\qquad \text{subject to} \\ &\qquad a^0x \geqslant b^0 \\ &\qquad x \text{ is } 0\text{–}1 \end{aligned} \qquad \begin{aligned} &\text{(S-MK)} \quad \text{Maximize } cx \\ &\qquad \text{subject to} \\ &\qquad a^0x \leqslant b^0 \\ &\qquad x \text{ is } 0\text{–}1 \end{aligned}$$

The method we propose to exploit the surrogate problems (S-GC) and (S-MK) may be viewed as a two level ghost image process. A surrogate constraint heuristic generates solutions at the first level, while a "higher level" process operates as a targeting mechanism to guide the process within it. We describe the lower level process first.

6.2. An embedded surrogate constraint method

The surrogate problems (S-GC) and (S-MK) provide the representations that correspond to the ghost image § at the first level. The mapping M§ to create the surrogate problem does so in two steps, first by disposing of direct implications and redundancies in the parent problem constraints, and then by generating w to define the associated surrogate constraint. The mapping Mx then produces a trial solution based on the surrogate problem.

Trial solutions are generated incrementally, applying Mx to assign a value to one component x_j of x at a time. As soon as an assignment occurs, M§ is immediately reapplied to generate the next surrogate problem. An overview of this procedure is as follows.

6.2.1. Outline of the embedded surrogate method (single pass)

Step 0. Introduce the link to incorporate information from the higher level process (after the first pass).

Step 1. Apply M§ to generate the current representation:

1A. Update the problem: drop currently redundant constraints and eliminate variables with implicitly assigned values. (If all variables are assigned values, terminate.)

1B. Create the current surrogate constraint and associated surrogate problem.

Step 3. Apply Mx to augment the current trial solution. By reference to the surrogate problem, select an unassigned x_j and set $x_j = 1$. Return to Step 1.

The components of the procedure that provide the explicit forms of M§ and Mx will now be elaborated.

6.2.2. Current problem representation. The operations of Step 1A are performed on a current problem representation that is progressively being reduced, not only in Step 1 but also in Step 2 as a result of selecting an unassigned x_j and setting $x_j = 1$. To represent the current problem form at each of these steps, it is convenient to draw on summation notation. Let JA denote the index set of variables currently assigned values, and let JU denote the complementary index set of variables without assigned values. Let x'_j denote the value assigned to x_j for $j \in$ JA, and let NR denote the index set of nonredundant constraints. Then the relation between the original matrix representation and the current (updated) summation representation is as follows.

Original Matrix Representation	Current Summation Representation
cx	$\sum (c_j x_j : j \in \mathrm{JU}) + c'$
Ax	$\sum (a_{ij} x_j : j \in \mathrm{JU}), i \in \mathrm{NR}$
b	$b'_i, i \in \mathrm{NR}$

The constants c' and b'_j are given by

$$c' = \sum (c_j x'_i : j \in \mathrm{JA})$$

$$b'_i = b_i - \sum (a_{ij} x'_j : j \in \mathrm{JA}).$$

The Current Summation Representation also corresponds to the original summation representation when all variables are unassigned and NR represents the index set of all constraints. Then JU is the index set of all components of x, JA is empty, and $b'_i = b_i$, $c' = 0$.

6.2.3. Applying M§: implementing Step 1A. The updating operations of Step 1A identify redundant constraints and determine implied values for problem variables by well known (and apparent) relationships, shown in the following table.

	Problem (GC)	Problem (MK)
Redundant constraint	$b'_i \leq 0$	$\sum (a_{ij} : j \in \mathrm{JU}) \leq b'_i$
Implied for $j \in$ JU:		
$x_j = 1$:	$\sum (a_{ih} : h \in \mathrm{JU} - j) < b'_i$ for some $i \in$ NC	
$x_j = 0$:		$a_{ij} > b'_i$ for some $i \in$ NC

In addition, $x_j=0$ is implied for all $j\in$JU in (GC) if the assignment $x_j=x'_j$ for $j\in$JA is feasible. For (GC) this is equivalent to stipulating all constraints are redundant. Although more advanced criteria based on dominance considerations and cutting plane analysis are also relevant, we will restrict attention to the preceding simple rules. (These can be executed with particular efficiency by organizing the non-zero a_{ij} coefficients in pre-sorted ascending or descending order for each i.)

6.2.4. Applying M§: implementing Step 1B. To create a surrogate constraint for Step 1B, we will focus on a class of approaches called normalization methods, where each component w_i of the vector w is produced as a function of the coefficients of the ith constraint. A rudimentary example of such a normalization is

$$w_i=1/b'_i \qquad \text{(N-0)}$$

where $i\in$NR. Redundant constraints, not present, may be imagined to receive a weight $w_i=0$. (N-0) has dominated implementations of surrogate constraint normalizations in the literature, perhaps because of its appealing simplicity. (Some implementations seek greater simplicity by using (N-0) only in reference to the original problem data, i.e., taking $b=b'_i$, without updating.)

We alternatively suggest normalizations for (GC) and (MK) given by

$$w_i=(1/u_i)^p \qquad \text{(N-GC)}$$

$$w_i=(u_i/(b'_i)^2)^p \qquad \text{(N-MK)}$$

The exponent p satisfies $p\geqslant 1$ and the quantity u_i is defined by

$$u_i=\sum(a_{ij}:\ j\in\text{JU}).$$

Although (N-GC) and (N-MK) appear "nonsymmetric," in fact they correspond to first redefining u_i relative to the modified form of constraint i after applying the normalization (N-0), and then respectively dividing and multiplying the modified constraint by this u_i raised to the power p. The exponent p is introduced to permit stronger differentiation of situations where one constraint may implicitly dominate or "almost dominate" another (a differentiation accentuated by larger values of p).

Many other normalizations are possible, but we have chosen these because they are straightforward and satisfy a special (highly natural) condition by which the values for w_i depend on the a_{ij} coefficients for $j\in$LU. We require this condition to be satisfied by any normalization used with the procedure we propose.

6.2.5. Applying Mx: implementing Step 2. The choice of an x_j to be assigned a value of 1 in Step 2 can take advantage of the surrogate constraint representation by dividing Step 2 into two parts, as follows.

2A. Generate a trial solution $x_j=x^0_j$, $j\in$JU, for the current surrogate problem.

2B. Select $j=j^*\in$JU by the following ratio criterion:

$$\text{(S-GC)}\quad j^*=\text{Argmin}(c_j/a^0_j:\ j\in\text{JU and } x^0_j=1)$$

$$\text{(S-MK)}\quad j^*=\text{Argmax}(c_j/a^0_j:\ j\in\text{JU and } x^0_j=1)$$

Then set $x_{j^*}=1$, transferring j^* from JU to JA.

Step 2A is often disregarded in surrogate constraint heuristics by instead applying Step 2B without reference to x^0. This causes j^* to be selected by the same rule as used in a "greedy knapsack heuristic." However, the surrogate constraint literature identifies knapsack heuristics better than the greedy one, and the inclusion of Step 2A seems preferable. Additional reasons for including Step 2A in the present context will soon be evident.

Most of the ideas presented to this point constitute special instances of those proposed for broader surrogate constraint applications (Glover [14, 15]). (An example of a popular instance is the heuristic of Chvatal [16] for covering problems, which results by applying (N-0) together with the abbreviated Step 2 that disresgards Step 2A.) We now show how the interpretation of these ideas within the ghost image framework leads to a more advanced procedure.

6.2.6. The higher level process. In the ghost image framework, the Embedded Surrogate Method also may be viewed simply as a single application of a "higher" or Master Mx, since it generates

a single completed trial solution. We are prompted accordingly to pair this Mx with a corresponding Master M§.

To characterize the larger method, we also draw on the ghost image ideas in another way. Recall that an image § can refer not only to a problem representation (as embodied in the surrogate problems) but also to a solution vector that targets the evolution of the current solution. In the present setting, these two notions may be conveniently combined. Denote the completed trial solution obtained by an application of the Embedded Surrogate Method as x''. Then x'' will be used in the next application of the embedded procedure to modify the normalization that creates the surrogate constraint, implicitly targeting the current solution to reinforce or attenuate freatures of x'' while simultaneously adjusting the current surrogate problem. (The higher level M§ draws on x'' to restructure the lower level M§). The organization of this process may be described as follows.

6.2.7. Outline of the higher process.

> **Step H1.** (Master M§), At each successive iteration (after the first), employ the previously generated solution x'' to be a target for modifying the surrogate constraint problems of the Embedded Method.
> **Step H2.** (Master Mx). Apply the Embedded Method to create a new trial solution and return to Step H1. To complete the specification of this solution process, we identify the method by which x'' is made to influence the surrogate constraint problems.

6.2.8. A surrogate contraction approach. We propose to exploit successive x'' solutions by contracting the domain over which the normalization is determined. Let JU(x'') denote the subset of JU defined by

$$\mathrm{JU}(x'') = (j \in \mathrm{JU}: x''_j = 1).$$

That is, JU(x'') is the index set of unassigned x_j in the current pass of the Embedded Method that received values of 1 in the solution x'' of the preceding pass. We conjecture that restricting consideration of unassigned variables to those with index in JU(x'') will have a special impact on a surrogate constraint generated by reference to the normalizations (N-GC) and (N-MK). As a basis for this conjecture, note that the contraction achieves the same result as if the variables with $x''_j = 1$ were the only variables in the problem. If x'' is optimal, the contraction appears appropriate ("makes sense"), which if x'' is not optimal, the resulting normalization should drive the solution elsewhere. This latter effect depends partly on the fact that constraints that are more nearly binding when $x = x''$ will tend to receive relatively larger weights under the contracted normalization than under the original. (The conjecture evidently is somewhat imprecise, and later we will indicate refinements of the normalization that influence its validity.)

To implement the contraction, we modify the form of the normalization in the Embedded Method by defining

$$u''_i = \sum (a_{ij}: j \in \mathrm{JU}(x'').$$

Correspondingly, we let w''_i denote the form of w_i when u''_i replaces u_i in (N-GC) and (N-MK), and call the resulting new normalizations (N-GC: x'') and (N-MK: x''). There are three chief ways to exploit the normalizations (N-GC: x'') and (N-MK: x''). We identify these as follows.

6.2.9. Static surrogate contraction. This is the most straightforward embodiment of the surrogate contraction approach. The rule is simply to apply the Embedded Method using (N-GC: x'') or (N-MK: x'') at each execution of Step 1B in place of (N-GC) or (N-MK). When a given pass of the Embedded Method terminates, the next pass is initiated using the previous x''. (The Higher Level Process stops either after a chosen number of steps or when x'' begins to follow a cyclic pattern.)

6.2.10. Extended surrogate contraction. The use of x'' in the static approach may be seen to resemble the incorporation of a single period short term memory in tabu search. (Although no strict tabu exists that prevents x'' from being immediately regenerated, our conjecture about the effect os x'' in the new normalizations suggests such a tabu typically will be observed.) Consequently, it becomes natural to extend the contraction approach by introducing a tabu-related memory that makes it possible to incorporate the influence of multiple previous solutions simultaneously.

To accomplish this, let $X(t)$ denote the set of solutions x'' generated on the t preceding passes of the Embedded Method. We permit t to operate as a short term "tabu list size" by assigning t

a small value, from an interval such as from 1 to 5, and then periodically altering t in Step 1 of the Higher Level Process. Associated with each t is an extended normalization identified by setting

$$w_i = \sum [w_i'': x'' \in X(t)].$$

(Instead of defining w_i by a simple sum, the w_i'' values can be weighted, as in reference to the associated cx'' values.) By the tabu search orientation, in addition to encompassing the short term effects by reference to this definition of w_i, the set $X(t)$ periodically can be replaced over the longer term by small subsets of best solutions. By selecting the subsets to differ from each other (using standard TS criteria), the process fulfills both an intensification and a diversification function.

6.2.11. Adaptive surrogate contraction. The relevance of (N-GC: x'') and (N-MK: x'') for shaping the new solution will gradually diminish as the number of variables x_j assigned values increases. As the associated index set JA grows, there is a growing likelihood that the currently generated solution x' will diverge from x''. Under these conditions, the use of x'' to guide the creation of surrogate constraints by the normalizations (N-GC: x'') and (N-MK: x'') becomes less meaningful.

To compensate for this effect, we diminish the reliance on x'' by replacing LU(x'') with a different subset of LU to define the normalizations. We propose to do this adaptively as part of the process of generating a trial solution for the surrogate problem in Step 2A of the Embedded Method.

A heuristic trial solution x^0 generated for the surrogate problem can easily be extended to provide a feasible solution for the original problem. For (GC), it suffices to augment x^0 by successively choosing additinal variables to set to 1 by the criterion of Step 2B (disregarding reference to x^0 in that criterion). For (MK), x^0 instead can be successively reduced, choosing variables such that $x_j^0 = 1$ and setting them to 0, selecting indexes for such variables by the rule that assigns $j^* = \text{Argmin}(c_j/a_j^0: j \in \text{JU and } x_j^0 = 1)$. (This is the criterion for setting variables to 1 for (GC), but we use it here to set variables to 0 for (MK).) The augmentation for (GC) or reduction for (MK) stops when feasibility is achieved.

In an adaptive approach, such an extended trial solution x^0 can then be used to replace x'' to generate a new surrogate constraint. More precisely, we replace LU(x'') by LU(x^0), thus creating the associated normalization (N-GC: $\mathbf{x}^0$) or (N-MK: x^0). The resulting surrogate constraint will induce the process to generate new solutions that depart from x^0 (appropriately, since x^0 has now been generated and can be included among candidates for the best solution). In summary, when x' begins to diverge from x'' (or at any point after a specified number of variables have been assigned values), we add the following Step 2A* to complement Step 2A of the Embedded Method.

> **Step 2A*.** Extend the surrogate constraint trial solution x^0 to be feasible for the original problem. Create the associated normalization (N-GC: x^0) or (N-MK: x^0) to generate the new surrogate constraint problem on the next interation.

With the inclusion of Step 2A*, Steps 2A and 2A* can be iterated. That is, by returning at once to Step 2A, the surrogate constraint based on x^0 can be used to produce a new x^0 (and still another new surrogate constraint at Step 2A*). (The Extended Surrogate Contraction approach also can be applied to influence this iterative implementation.)

The Adaptive Surrogate Contraction process likewise can be executed on the initial pass of the Embedded Method, before a solution x'' has been determined. Without iterative application of Steps 2A and 2A*, it represents a single pass (non-iterative) constructive heuristic that may be compared to other constructive heuristics for covering and multidimensional knapsack problems, such as proposed by Chvatal [16], Senju and Toyoda [17], and Toyoda [18]. The inclusion of an iterated component represents a miniaturization of the recursive aspect that results when the Embedded Method is incorporated into the Higher Level Process.

For convenient reference, we put the full procedure together in one place as follows.

6.2.12. Complete surrogate contraction method

The higher process

Step H1 (master M§). At each successive iteration (after the first), designate the last solution x' generated in the Embedded Method to be the solution x'', which will be incorporated to create surrogate contractions modifying the surrogate constraint problems of the Embedded Method. (Optionally employ extended surrogage contractions by reference to the set $X(t)$ of t previous x'' solutions.)

Step H2 (master Mx). Apply the following procedure to create a new trial solution.
Embedded Method

Step 1. Apply M§ to generate the current representation:
1A. Update the problem: drop currently redundant constraints and eliminate variables with implicitly assigned values. The values of currently assigned variables are denoted by x'_j, $j \in JA$. If all variables are assigned values, go to Step 3.
1B. Create the current surrogate constraint and associated surrogate problem (S-CG) or (S-MK). Use the normalization (N-GC) or (N-MK) on the first pass, and (N-GC: x'') or (N-MK: x'') on subsequent passes.
Step 2. Apply Mx to augment the current trial solution: by reference to the surrogate problem, select an unassigned x_j, $j \in JU$, and set $x_j = 1$ as follows.
2A. Generate a trial solution $x_j = x_j^0$, $j \in JU$, for the current surrogate problem.
2A*. (Optional Adaptive Contraction) Extend the surrogate constraint trial solution x^0 to be feasible for the original problem, and check it as a candidate for the best solution found. Create the associated normalization (N-GC: x^0) or (N-MK: x^0) to generate the new surrogate constraint and surrogate problem on the next iteration. (If Step 2A* is executed, Steps 2A and 2A* may be iterated, carrying forward the best x^0 to 2B.)
2B. Select $j = j^* \in JU$ by the following ratio criterion:

$$\text{(S-GC)} \quad j^* = \text{Argmin}(c_j/a_j^0 : j \in JU \text{ and } x_j^0 = 1)$$

$$\text{(S-MK)} \quad j^* = \text{Argmax}(c_j/a_j^0 : j \in JU \text{ and } x_j^0 = 1)$$

Then set $x_j^* = 1$, transferring j^* from JU to JA, and return to Step 1.
Step 3. Check the current completed trial solution x' as a candidate for the best solution found. Terminate the Embedded Method to return to Step H1 of the Higher Process. (The complete method terminates after a chosen number of iterations, or when the x' solution begin to cycle in the simpler variants.)

From an empirical standpoint, the complete procedure invites comparison with other surrogate constraint procedures for the problems (GC) and (MK), as developed by Gavish and Pirkul [19] and Freville and Plateau [20].

6.2.13. Refinements. To give a more effective procedure, we conjecture that the definition of u_i'' for determining (N-GC: x'') and (N-MK: x'') should preferably be altered to impose the influence of x'' less abruptly, allowing the "correct" form of the constraints, defined over the full set LU, to retain a degree of influence. (The incremental change notion in Principle 9 of the ghost image framework similarly suggests the relevance of imposing the surrogate contraction more gradually.) Thus, we suggest refining the definition of u_i'' to become

$$u_i'' = \lambda \sum (a_{ij} : j \in \text{LU}(x'')) + (1-\lambda) \sum (a_{ij} : j \in \text{LU})$$

where λ is a constant between 0 and 1. The value of λ determines the relative influence to be exerted by x''; i.e., $\lambda = 1$ yields the previously indicated form of (N-GC: x'') and (N-MK: x''), while smaller values of λ give more influence to the coefficients over the full set LU, yielding (N-GC) and (N-MK) when $\lambda = 0$. From a diversification standpoint, larger values of λ are likely to generate solutions that differ more significantly from each other (but may also induce cycling more quickly unless the Extended Surrogate Contraction is employed).

7. PRELIMINARY COMPUTATIONAL EXPERIENCE†

To test the relevance of the preceding ideas for developing computational algorithms, a simplified version of the ghost image procedure of Section 6.2 was applied to the multidimensional knapsack problem. The method incorporated the prescriptions of the first part of Section 6.2, but did not

†The computational testing in this section is due to F. Grange and G. Kochenberger, and summarizes initial findings of a collaborative study reported at the 1992 Joint National ORSA/TIMS Meetings, Orlando, FL.

include the extended or adaptive surrogate contractions, and also did not employ the refinements indicated at the end of the section.

7.1. *Problem generation*

Test problems were created for multidimensional knapsack models by sequentially generating objective function coefficients, constraint coefficients and knapsack capacities. Objective function coefficients were randomly sampled from an independent, uniform distribution over [0, 1], producing an assignment of the form $c_j \leftarrow U[0, 1]$.

To establish a partial correlation between the objective function and the constraint data, the constraint coefficients for a column were triangularly distributed over the interval [0, 1] with mode equal to the sampled objective coefficient for the variable, thus yielding an assignment $a_{ij} \leftarrow$ Triangular $[0, c_j, 1]$.

For each test problem, a "tightness" specification was provided as a fraction of the total knapsack requirements of the variables. Given the requirement a_{ij} of variable j for knapsack i, the capacity b_i of knapsack i was generated by the uniform distribution below:

$$b_i \leftarrow U[100 - \text{Tightness}\%, \text{Tightness}\%] \cdot \sum_j a_{ij}$$

For example, each knapsack constraint in a test problem with 60% "tightness" would have capacity uniformly distributed from 40 to 60% of the sum of its constraint coefficients.

7.2. *Test code*

The test code for the multiple-knapsack ghost image process (GIP) was written in Borland International object-oriented Turbo Pascal™ version 6.0 on a Zenith 256 microcomputer with an Intel 80286 CPU. The objective function, constraint and knapsack capacity data were stored in virtual array objects furnished in the Turbo Power Object 1.0™ library.

Test problem generation followed the procedure given above and used Turbo Pascal's own pseudo-random number generator. The test code also wrote generated problems to ASCII files in MPS format for subsequent optimization by LINDO™.

To provide a basis for comparison, the preliminary GIP code was tested against Senju and Toyoda's well known "drop" heuristic for multidimensional knapsack problems (Senju and Toyoda [17]). Each test problem was solved by both methods, and timing, objective values and solution vectors were obtained.

Twenty-five test problems were generated by the design of Section 7.1 with tightness measures ranging from 50 to 90% and with sizes (constraints × variables) consisting of (5 × 20), (10 × 20), (10 × 50), (20 × 50), and (20 × 100) (five problems in each size category).

The GIP test code ranged from five to eleven times slower than the Senju and Toyoda heuristic, but obtained solutions whose quality uniformly dominated the quality of solutions obtained by this heuristic. The improvement in objective function values ranged from 7 to 11%, which represents a substantial gain for this class of problems. Further study is required to test more sophisticated GIP implementations on wider ranges of problems and against additional alternative procedures. Nevertheless, this preliminary experimentation is an encouraging indication of the potential usefulness of the GIP approach.

8. SUMMARY CONSIDERATIONS

Ghost image processes that use even relatively simple parametric deformations offer a number of interesting options to test (e.g., relative to the manner of updating the parameter(s) we have identified by w, and choosing the precise forms of the accompanying Mx or Mx^* mappings). As a first step, the simpler implementations also yield a reasonable point of departure for gaining insights into the relevance of ghost image strategies by comparison to other attempted uses of neural networks in optimization.

The ghost image processes also create an opportunity to define new types of tabu search and simulated annealing methods, consisting of composite nested procedures. In particular, a TS or SA method can be used to define the transformations Mx and Mx^* that generate trial solutions from the current § and x, and in addition can be used at a higher level to define the transformation of

§ to its ultimate form. In the case of simulated annealing, a scalar parameter λ (as introduced in both the parametric deformation and surrogate contraction processes), can be treated as a form of temperature, where a stipulated number of subphases of SA are carried out for a given λ to generate the current x. This defines a form of temoperature that is a "data gradient" rather than an "energy gradient" (where the latter refers to objective function changes). Cooling schedules for this type of temperature may exhibit properties different from those applicable to a standard form of temperature.

Within the tabu search framework, the goal is not to move λ monotonically from initial to final state, but to use an adaptive progression (as in the use of strategic oscillation in the closed deformation processes). It should be noted that the SA treatment of λ as a temperature must be additionally altered for the mappings that use more than a single λ parameter (as by adjusting selected columns or rows of a matrix, or selected elements of these columns and rows in the closed deformation processes). Tabu search, on the other hand, adapts to such situations without altering its basic structure.

8.1. Further connections with the principle of proximate optimality

Underlying the preceding discussion is the assumption that the Proximate Optimality Principle is relevant to the ghost image processes in the parametric deformation approaches—as we have previously conjectured its relevance in other approaches. The stipulation that good solutions at one level are likely to be found close to good solutions at another leads to the goal of characterizing processes that can give worthwhile definitions of "level" and "close to" in an optimization setting. If indeed the POP notion has substance, then it offers a potential explanation for the success of various "narrowly focused and incremental" processes, in the spirit of Principle 9. Moreover, it suggests that greater success may be achieved if such processes adopt the explicit goal of obtaining good solutions relative to a given characterization of a "level"—rather than simply trusting to small and locally attractive moves to lead to desired consequences (with or without a randomizing components as in SA). The POP idea suggests that greater power will be achieved if not just the "final" solution but a set of best solutions is carried forward from one level to the next. Thus the POP provides a rationale that motivates observations contained in Principles 10–12. The ghost image processes provide a useful basis for testing the merit of this principle, and for identifying new forms in which it may be embodied.

8.2. Concluding reflection

It is probably inevitable that some AI proponents will be uncomfortable with allowing ghost image processes to use mappings Mx and M§ that represent relatively advanced problem solving machinery (such as the simplex method). We prefer to believe that intelligence involves the ability to exploit sophisticated tools as well as primitive ones. Once a mapping Mx or M§ has been created, and enters into a pool we are willing to make available to one group of intelligent processes (ourselves), there seems no reason to forbid access to other intelligent processors, or to aspiring models of such processors.

This is not to say we should exclude consideration of more modest tools to discover what can be fabricated with them. Improvements in processes and model frameworks at a rudimentary level may translate into improvements at higher levels. If one of our goals is to discover good procedures for complex problems, however, it also seems relevant to equip our models with options that already inherit a bit of ingenuity before putting them to the test. (From some of the literature, one might infer that mathematical invention does not have a legitimate place in the exercise of intelligent thought.)

We suggest the ambition of AI should not be limited to taking primitive components and making them work half decently, but also should include taking more advanced components and making them work even better. A worthwhile framework for modeling intelligent processes should have the ability to do both.

REFERENCES

1. T. Kohonen, *Self-Organization and Associative Memory*, Springer, Berlin (1988).
2. J. J. Hopfield and D. W. Tank, Neural computation of decisions in optimization problems, *Biol. Cybernetics* **52**, 141–152 (1985).

3. D. S. Johnson, C. R. Aragon, L. A. McGeoch and C. Schevon, Optimization by simulated annealing: an experimental evaluation; part 1, graph partitioning, *Ops Res.* **37**, 865–892 (1989).
4. F. Glover and M. Laguna, Tabu Search, in *Modern Heuristic Techniques for Combinatorial Problems* (Edited by C. Reeves). Blackwell (1992).
5. D. de Werra and A. Hertz, Tabu search techniques: a tutorial and an application to neural networks, *OR Spectrum*, 131–141 (1989).
6. J. Chakprani and J. Skorin-Kapov, Connectionist approaches to the quadratic assignment problem, *J. Computers Ops Res.* **19**, 287–295 (1992).
7. B. Angéniol, G. Croix Vaubois and J. Le Texier, Self-organizing feature maps and the travelling salesman problem, *Neural Networks* **1**, 289–293 (1988).
8. R. Durbin and D. Willshaw, An analogue approach to the TSP using an elastic net method. *Nature* **326**, 689–691 (1987).
9. H. El Ghaziri, *A self-organizing Map Approach for TSPs*, EPFL, DMS, Lausanne, Switzerland (1990).
10. M. Geraci, F. Sorbella and G. Vassallo, *A New Approach to the Travelling Salesman Problem Using Kohonen Maps.* CRES, Centro par la Ricerca Elettronica in Sicilia, Montreale (Palermo) (1991).
11. F. Glover, E. Taillard and D. de Werra, A user's guide to tabu search. To appear in *Ann. Pps Res.*
12. H. El Gahziri, Solving routing problems by a self-organizing map. *Artificial Neural Networks* (Edited by T. Kohonen, K. Mäkisara, O. Simula and J. Kanagas), pp. 829–834. Elsevier (1991).
13. M. Laguna, J. Kelly, J. L. Gonzales-Valarde and F. Glover, A Tabu search approach for multilevel generalized assignment problems, Graduate School of Business, University of Colorado, Boulder (1991).
14. F. Glover, A multiphase-dual algorithm for the zero–one integer programming problem, *Ops Res.* **13**, 879–919 (1965).
15. F. Glover, Heuristics for integer programming using surrogate constraints, *Decis. Sci.* **8**, 156–166 (1977).
16. V. Chvatal, A greedy heuristic for the set covering problem, *Maths Ops Res.* **4**, 233–235 (1979).
17. Senju and Y. Toyoda, An approach to linear programming with 0–1 variables, *Mgmt Sci.* **15**, 196–207 (1968).
18. Y. Toyoda, A simplified algorithm for obtaining approximate solutions to zero–one programming problems, *Mgmt Sci.* **21**, 1417–1428 (1975).
19. B. Gavish and H. Pirkul, Efficient algorithms for solving multiconstraint zero–one knapsack problems to optimality, *Mathl Program.* **31**, 78–105 (1985).
20. A, Freville and G. Plateau, An efficient preprocessing procedure for the solution of the 0–1 multiknapsack problem to optimality, *Mathl Program.* **31**, 78–105 (1991).
21. F. Glover, Future paths for integer programming and links to artificial intelligence, *Computers Ops Res.* **13**, 533–549 (1986).