
Journal of Heuristics, 11: 89–108, 2005

A GRASP and Path Relinking Heuristic for Rural Road Network Development

MARIA P. SCAPARRA
Kent Business School, University of Kent, Canterbury CT2 7PE, UK

RICHARD L. CHURCH*
Department of Geography, University of California, Santa Barbara, Santa Barbara, California 93106-4060, USA
email: church@geog.ucsb.edu

Submitted in December 2003 and accepted by Steve Chiu in February 2005 after 1 revision

Abstract

This paper presents a model for rural road network design that involves two objectives: maximize all season road connectivity among villages in a region and maximize route efficiency, while allocating a fix budget among a number of possible road projects. The problem is modeled as a bicriterion optimization problem and solved heuristically through a greedy randomized adaptive search procedure (GRASP) in conjunction with a path relinking procedure. The implementation of GRASP and path relinking includes two novel modifications, a new form of reactive GRASP and a new form of path relinking. Overall, the heuristic approach is streamlined through the incorporation of advanced network flow reoptimization techniques. Results indicate that this implementation outperforms both GRASP as well as a straightforward form of GRASP with path relinking. For small problem instances, for which optimality could be verified, this new, modified form of GRASP with path relinking solved all but one known instance optimally.

Key Words: network design, metaheuristics, GRASP, path-relinking

Introduction

Road network design and investment planning has been the subject of considerable research in the past three decades (see, for example, Magnanti and Wong, 1984; Yang and Bell, 1998, for comprehensive surveys). In the most general form, the network design problem (NDP) can be stated as the problem of optimally allocating a limited budget among road projects in order to maximize efficiency. Road investments include the construction of new links as well as the upgrade and widening of existing road segments.

The wide range of network design models proposed in the literature reflects the need for planning efficient road systems in very diverse socioeconomic and environmental contexts. Depending upon the field of application, in fact, different operational characteristics, technical requirements, problem objectives and system performance measures arise which need to be captured within an optimization model. As an example, planning for a

*Author to whom all correspondence should be addressed.

high-volume urban transportation network must account for traffic congestion, elastic demand and equilibrium flow patterns (Sheffi, 1985). On the other side of the spectrum, the design of low-volume, dispersed rural roads in developing countries is not concerned with congestion effects but requires encompassing other system perspectives, such as connectivity and accessibility (Antunes, Seco, and Pinto, 2003). The objective of this paper is to address the latter type of problem, planning rural road development in developing countries.

The transportation systems in many developing countries are often substandard, and represent a major barrier to people and goods mobility, agricultural development, and economic growth. Village connectivity is usually very poor and mainly relies on earthen tracks, which become impassable during the rainy season. As an example in Uganda 20 percent of the National Roads are paved and 85 percent of the community access roads are in a poor condition. In Sudan, asphalted all-weather roads amount to roughly 10–15 percent of the total road system while most of the commercial flows occur on substandard earth or gravel roads. A common objective of development is to provide all-weather road connectivity to towns and villages. The objective of this paper is to propose an optimization model that is capable of identifying the most cost-effective way of developing an all-weather road system through the selection of road segments to be upgraded and paved. The primary goal of the model is to provide efficient connections among villages while maximizing the connectivity level achieved.

Yang and Bell (1998) identify four major components characterizing network design problem variants: design variables, routing behavior, demand characteristics and design objectives. The model we propose for planning rural road development uses discrete design variables to represent the choice of the links to be upgraded and paved. The existing system of gravel and earthen roads are treated as the available links for upgrading and paving. Commonly, discrete models are applied when planning for the formation of new road systems, whereas models using continuous variables are more applicable to road capacity expansion planning in congested networks (Abdulaal and LeBlanc, 1979). In terms of routing behavior, we assume that the optimal routing of trips among villages is via the shortest paths on the all-weather network. This is a reasonable assumption for those models where congestion is not a major concern and providing efficient all-weather transportation access is one of the principal objectives. When dealing with traffic flow among villages in developing countries, the basic travel demand can be assumed largely inelastic. Within this context, we do not consider the effect on improved mobility due to network improvements, although this could be easily done. We assume a fixed matrix of trips among villages and use a gravity model as a proxy for generating the trip distribution.

The prevailing objective function in the design of transportation networks has been the minimization of the total network travel costs. However, many other objectives have also been considered, which include accessibility (Antunes, Seco, and Pinto, 2003), equity (Feng and Wu, 2003), social and economic benefits (Meng and Yang, 2002) among others. Furthermore, in recognition of the inherent multi-objective nature of transportation planning problems, many authors have investigated the possibility of identifying best compromise solutions with respect to several development objectives (see Current and Min, 1986; Yang and Bell, 1998, for reviews on multi-objective models). The problem treated in this paper is a bicriterion optimization problem aimed at minimizing the sum of trip-weighted shortest path distances between all pairs of villages connected by all-weather roads (efficiency criterion)

and maximizing the traffic flow volume provided by all-weather paved roads (connectivity criterion). Network design problems that only consider the first objective have received considerable attention in the literature (Boyce, Farhi, and Weischedel, 1973; Johnson et al., 1978; Dionne and Florian, 1979) and are referred to by Magnanti and Wong (1984) as budget design problems. However, these formulations assume that it is both necessary and possible to provide a fully connected network passable in all seasons. In practice, planning situations exist where satisfying the entire traffic flow is either impossible because of the resource constraints, or not efficient, meaning that enforcing full connectivity may result in a lengthy and inefficient network. Our scope is to develop a model that can yield reasonable compromise solutions in view of both efficiency and connectivity. This problem has never been formally treated in the literature.

This bicriterion network investment problem is addressed by combining the two objectives into one through the use of weighting coefficients. Depending upon the weights assigned to each criterion, different pavement strategies are generated that reflect the relative importance of the two goals. In the following, we will refer to the bicriterion problem stated above as connectivity maximizing budget design problem (COMBDP).

The connectivity maximizing BDP is a generalized model of the BDP. Since the BDP has been classified as NP-hard (Johnson, Lenstra, and Rinnooy Kan, 1978), the COMBDP can also be so classified. Since only small instances of this problem have been solved optimally, our research has concentrated on the development of a heuristic strategy for solving COMBDP. Our proposed methodology is based upon an adaptation of the greedy randomized adaptive search procedure, GRASP (Feo and Resende, 1989), used in conjunction with the recently introduced path relinking technique, PR (Glover and Laguna, 1997).

The remainder of the paper is organized as follows. In the next section, we introduce some useful notation, which is used to define COMBDP more formally. Section 2 gives a brief overview of the solution methodology. In Section 3, we discuss the implementation details of the greedy randomized adaptive search procedure, while the path relinking approach is described in Section 4. Some computational experience follows in Section 5. We conclude with a summary and recommendations for future work.

1. Notation and problem statement

Rural road network systems can be represented as an undirected graph $G = (N, A)$ where the set N of nodes represents the villages of the rural region under study and the set of arcs A represents the existing earth-tracks which connect the villages as well as possible new connections. Each link (i, j) in A has two numerical values, d_{ij} and c_{ij}, associated with it, which represent respectively its length and the cost for upgrading it to a paved, all-weather road connection. For each pair of nodes i and j, we denote by a_{ij} the volume of traffic flow between the corresponding villages i and j. We assume here that the traffic volume is static over time and that such volume will not increase with improved routes, although this can be easily handled by the heuristic. For instance, if we assume that the traffic volume between two villages i and j increases from a_{ij} to a'_{ij} if an all-weather connection is established between i and j, the only modification required in the approach consists in using the new value a'_{ij} instead of the initial a_{ij}. This is straightforward based upon our assumption that

both the connectivity and the efficiency objectives are measured only in terms of the traffic flow on the all-weather road network.

A solution to the network investment problem is a subgraph $G' = (N', A')$ of G, where the set of arcs A' corresponds to the road segments selected for pavement, and the set of nodes N' corresponds to the set of villages that are served by at least one paved road.

A solution G' is feasible if the total cost of the selected arcs does not exceed a prespecified budget B, that is to say:

$$\sum_{(i,j)\in A'} c_{ij} \leq B \tag{1}$$

Note that G' is not necessarily connected but it can be represented as the union of a number K of disjoint subgraphs $G'_1, \ldots, G'_k, \ldots, G'_K$, each one representing a maximal set N'_k of connected nodes. A subgraph G'_k represents a set of villages which are pairwise connected by at least one all-season paved route. The value of each connected subgraph G'_k, $k = 1, \ldots, K$, is measured in terms of: (1) the traffic volume that can be served by all-season paved roads, and (2) the weighted distance associated with traffic being efficiently routed on all-season paved roads. The total value, $P(G'_k)$, of a subgraph G'_k can be expressed as the weighted combination of these two performance measures. Formally:

$$P(G'_k) = w_1 \sum_{i\in N'_k} \sum_{j\in N'_k} a_{ij} - w_2 \sum_{i\in N'_k} \sum_{j\in N'_k} a_{ij} L_{ij} \tag{2}$$

where L_{ij} is the length of the shortest all-season route between i and j in subgraph G'_k.

The first term in the right hand side of formula (2) represents the total traffic volume among the villages in G'_k. This first term represents inter-village traffic volume served by all season paved routes. The second term of the right hand side of formula (2) is the sum of trip-weighted shortest path distances (using all season paved roads) between all pairs of villages in G'_k. The two terms are weighted by w_1 and w_2 which represent the relative importance of each objective. Overall, we want to maximize the value of the first term (i.e. maximize connectivity) and minimize the value of the second term (i.e. minimize total weighted travel distance on all season paved roads). These two objectives can be combined by including a minus sign for the second term (when $P(G'_k)$is to maximized).

The total combined value of a solution G' over all subgraphs k is then:

$$P(G') = \sum_{k=1}^{K} P(G'_k) \tag{3}$$

The COMBDP seeks an investment strategy that maximizes $P(G')$, i.e., the strategy that optimizes the weighted combination of traffic flow volume served by paved roads and weighted traveled distance associated with those traffic volumes.

2. Solution method

In order to find good approximate solutions to COMBDP, we propose a methodology based on the combined use of GRASP and path relinking. GRASP, which first appeared in Feo and Resende (1989) and was later formalized in Feo and Resende (1995), is an iterative two-phase metaheuristic. At each iteration, the first phase produces a solution through the use of a randomized greedy construction scheme. Local search is then applied in the second phase to this solution, in order to obtain a local optimum in its neighborhood. Enhanced versions of the basic GRASP metaheuristic have been applied to a wide range of combinatorial optimization problems (the reader is referred to Resende and Ribeiro (2002) for an extensive treatment of the methodology and its applications, and Festa and Resende (2001) for an annotated bibliography).

Path-relinking was first introduced as a tool to compound intensification and diversification strategies in the context of tabu search (Glover, 1996; Glover and Laguna, 1997). The strategy is formulated on the principles of evolutionary approaches but unlike conventional evolutionary techniques (e.g., genetic algorithms), it does not employ randomization to generate new solutions. Instead, it constructs them through a methodical exploration of trajectories that connect previously generated high quality solutions. An in-depth description of path relinking can be found in Glover (1999) and Glover, Laguna and Martí (2000).

The first application of GRASP and path relinking was undertaken by Laguna and Martí (1999). Since then, a few other applications have appeared that combine the two methodologies. Some applications use path relinking as an intensification strategy within the GRASP procedure (see, for example, Resende and Ribeiro, 2003); others apply path relinking as a post-optimization step after the execution of GRASP (Ribeiro, Uchoa, and Werneck, 2002). Some authors considered both utilizations of the path relinking strategy (Aiex et al., 2003; Resende and Werneck, 2002). According to the survey work on GRASP by Resende and Ribeiro (2002), path relinking is more effective when used as an intensification phase. In our implementation, we have chosen to use it at the end of each GRASP iteration in order to intensify the search around local optima.

3. GRASP implementation

When implementing a GRASP procedure, several different issues need to be addressed and tailored to the structural characteristics of the problem under study. First, an adaptive greedy function needs to be defined to guide the iterative construction phase, which builds the solution by adding one element at the time. The greedy function is adaptive in the sense that it must be updated after the insertion of each new element in the partial solution under construction. Second, a restriction mechanism must be defined to build the restricted candidate list (RCL), that is the list from which to select the next element to be added to the solution. A probabilistic selection strategy must then be specified to select an element from the RCL. Finally, the essential constituents of the local search procedure (i.e., the neighborhood structure and the search strategy) must be defined.

3.1. Constructive phase

The constructive phase for COMBDP starts with an empty solution and iteratively adds arcs to it according to a randomized greedy scheme. When no more arcs are affordable for pavement, the constructive phase terminates and the local search procedure is initiated.

The greedy function guiding the arc selection is defined as the incremental unit value due to the addition of an unselected arc to the solution. Formally, for each arc (i, j) which is not part of the current partial solution G', the greedy function is given by:

$$p_{ij} = \frac{\Delta P_{ij}}{c_{ij}} = \frac{w_1 \Delta A_{ij} - w_2 \Delta D_{ij}}{c_{ij}} \tag{4}$$

In formula (4), ΔP_{ij} denotes the change of the objective function value due to the insertion of arc (i, j) in the solution, and is obtained as the weighted combination of the change in covered demand flow (ΔA_{ij}) and the change in weighted traveled distance (ΔD_{ij}). We provide a detailed approach to how these two terms (ΔA_{ij} and ΔD_{ij}) can be calculated efficiently in Section 3.3.

At each construction iteration, the greedy function is used to determine a restricted list of candidate arcs for selection (RCL). The RCL is built by means of an achievement level restriction mechanism (Resende, 1998). Namely, the unselected arcs are ranked according to the adaptive greedy function (4) and inserted in the RCL if the following two conditions are met: (i) their incorporation in the solution does not violate the budget constraint (1); and (ii) their greedy function value is in the range $[\alpha p_{\max}, p_{\max}]$, where $p_{\max}$ is the greedy function value of the best possible insertion at that construction iteration. The restricted candidate parameter α varies between 0 and 1, and its value strongly conditions the kind of solutions generated. In particular, values of α close to 0 increase the amount of randomness of the algorithm. This generally results in poor solution but greater diversification among the solutions generated in the constructive phase. On the other side, values of α close to 1 increase the amount of greediness, thus producing solutions of higher quality but less diversified. In our computational investigation we will show the results obtained for different values of α ranging between 0.1 and 0.9. We also investigate the use of a self-tuning mechanism for automating the choice of the parameter α. Such a calibration process, known as Reactive GRASP (Prais and Ribeiro, 2000), uses a probability distribution to select at each iteration a particular value for α from a set of candidate values. As the algorithm execution proceeds, the selection probabilities are biased to reflect the goodness of each alpha choice in producing high quality solutions.

Once the restricted candidate list is defined, a candidate arc is randomly selected from it and added to the partial solution under construction. The unit values p_{ij} associated with each arc not in the solution are then updated according to formula (4) to reflect the changes based upon the inclusion of the last added arc. The constructive phase terminates when no arc can be added to the solution without exceeding the residual available budget. The obtained solution is then subjected to the local search phase.

3.2. *Local search*

The neighborhood local search phase is based upon add, drop and swap moves. Namely, the neighbors of the current solution are all the solutions that can be obtained either by adding a new arc to the set of already selected arcs (add), or by removing a selected arc (drop), or by substituting a selected arc with an unselected arc (swap). A move is legal only if it maintains the feasibility of the solution with respect to the budget constraint. The worth of a move is calculated as the change in the weighted objective function resulting from it. Since the evaluations of all neighboring solutions are computationally expensive, we use a first improvement strategy in our local search phase, meaning that the current solution is replaced by the first improved solution detected in its neighborhood.

The search of an improving neighbor is performed by storing the arcs in the current solution and the arcs not in the current solutions in two lists, L_{in} and L_{out}, respectively. Each arc in L_{in} is first evaluated for dropping. If the arc removal is profitable, the move is performed and the search resumes from the next arc in the list; otherwise the arcs in L_{out} are examined in order until an arc is detected which can be profitably swapped with the arc in L_{in} under examination. If neither profitable drops nor profitable swaps can be detected after a whole pass of the elements in L_{in}, the local search enters the add phase in which add moves are attempted for the elements in L_{out}. If a profitable add move can be performed the search is restarted by evaluating the arcs in L_{in} for dropping and swapping. The search ends after a full pass of both lists with no improvement of the solution. Different ways of alternating add, drop and swap moves have been experimented in the initial testing. However, the particular choice just described was the one that provides the best results, both in terms of solution quality and execution time.

3.3. *Add and drop operations*

As already observed in the previous sections, arc insertions and removals imply a change in both the demand flow served by paved roads and in the weighted distance of flow of the current solution. Computing the changes in the values of the objective terms can be computationally extensive, since it requires checking the graph connectivity and recomputing the shortest path distances among all pairs of connected nodes. Furthermore, these operations need to be performed a considerable number of times in each GRASP iteration, both in the construction phase for updating and adding the selected arcs, and in the local search phase for each attempted move. To reduce the computational effort, the add and drop move implementations can be streamlined by exploiting the structural characteristics of the partial solution graph and by incorporating in the solution approach efficient methods for shortest path reoptimization. We explain this in detail next.

In the following, we assume that, throughout the algorithm execution, we keep track of all the connected subgraphs that form the partial solution G' as well as the shortest path distance matrix, L, which stores the shortest distances, L_{ij}, between each pair of nodes i and j in G'. After each add and drop move, the data structures storing the solution subgraphs and the shortest path distance matrix are updated to reflect the changes just made to the solution. For each node i in the current solution (i.e., $i \in N'$), we denote by $G'(i)$ the

connected subgraph that contains i and by $N'(i)$ its node set. We next describe how changes in the two objective function terms can be efficiently computed when accounting for the insertion of a new arc in the solution.

Let (i, j) be the arc to be added to a partial solution graph $G' = (N', A')$. The change in value of the two objective function terms due to an add move can be best defined by distinguishing four possible cases.

(i) Neither node i nor node j belong to N'. In this case, adding arc (i, j) does not affect the connectivity among the other nodes in the solution, and the change of the two objective terms are simply:

$$\Delta A_{ij} = a_{ij},$$
$$\Delta D_{ij} = a_{ij} d_{ij}.$$

(ii) Only one of the arc endpoints belongs to N'. Without loss of generality, assume, for example, that node j belongs to the subgraph $G'(j)$ of G', while $i \notin N'$. In this case, adding arc (i, j) allows connecting node i to all the nodes h in $N'(j)$. The changes in the objective terms are easily computed as:

$$\Delta A_{ij} = \sum_{h \in N'(j)} a_{ih},$$
$$\Delta D_{ij} = \sum_{h \in N'(j)} a_{ih}(d_{ij} + L_{hj}).$$

Note that the values L_{hj} are known and no shortest path recalculation is needed.

(iii) Both node i and node j are in N' and they belong to two different subgraphs, $G'(i)$ and $G'(j)$. When arc (i, j) is added, all the nodes in $G'(i)$ and $G'(j)$ can be connected only through link (i, j). This means that we can compute the change in the values of the two objective terms as:

$$\Delta A_{ij} = \sum_{l \in N'(i)} \sum_{h \in N'(j)} a_{lh},$$
$$\Delta D_{ij} = \sum_{l \in N'(i)} \sum_{h \in N'(j)} a_{lh}(L_{li} + d_{ij} + L_{jh}).$$

Also in this case, there is no need for shortest path recalculations.

(iv) Both node i and node j are in G' and they belong to the same subgraph, G'_k. Only in this case, does evaluating the change in weighted traveled distance, ΔD_{ij}, require recomputing the shortest path distances between the nodes in G'_k, since some of the traffic flow might best be served through a path which uses link (i, j). The change in

served traffic volume is 0, since the newly inserted arc does not affect the total traffic volume that can be provided by paved roads. Hence:

$$\Delta A_{ij} = 0, \tag{5}$$

$$\Delta D_{ij} = \sum_{l \in N'_k} \sum_{h \in N'_k} a_{lh}(L'_{lh} - L_{lh}). \tag{6}$$

In formula (6), L'_{lh} denotes the new shortest path distance from node l to node h after the insertion of arc (i, j) in G'_k.

Computing formula (6) requires reoptimizing the shortest path tree rooted at each node in G'_k. In order to perform the reoptimization efficiently, we use one of the algorithmic approaches described in Pallottino and Scutellà (2003) and based on the pioneering work of Gallo (1980). The procedure is designed for finding the new shortest path tree when the cost of one arc is reduced. To adapt the procedure to our problem, we assume that the cost of the arc to be added is initially infinite and that, as result of the add move, it is decreased to its actual length (in practice, for each connection to be added, we insert into the graph two directed arcs, (i, j) and (j, i), with equal lengths). The basic premise of this method is to exploit the knowledge of the existing shortest path tree in computing the new shortest path tree. In the initial shortest path tree, the arc reduced costs associated with a linear programming model of the shortest path tree are all positive, as stated by the optimality conditions of linear programming. If the reduced costs of the newly inserted arcs are also positive, the tree is still optimal and no reoptimization is needed. Otherwise, if one of the reduced costs is negative, a Dijkstra shortest path procedure, which uses the reduced costs instead of the actual costs, is applied to propagate the negative value associated with that arc. It is important to note that when the Dijkstra procedure, based on the reduced costs, terminates, the label associated with each node measures the contraction of the shortest path tree from the root to that node. In our specific case, this value is exactly the shortest path distance variation $L'_{lh} - L_{lh}$ needed in formula (6), for a given root node l and a generic node h in G'_k. For more details on the reoptimization methodology the reader is referred to Pallottino and Scutellà (2003) and Gallo (1980). The use of this procedure within our solution approach proved to be very effective in practice and significantly reduced the computational time required for recomputing shortest paths.

We now describe how to efficiently compute a drop move. Let (i, j) be the arc to be dropped from the current solution and G'_k the subgraph currently containing node i and node j. We can identify two possible cases.

(i) The removal of arc (i, j) disconnects node i and node j in terms of all-season paved road travel, i.e., it generates a cut $(N'_k(i), N'_k(j))$ in G'_k, where $N'_k(i)$ is the subset of nodes connected to node i after the removal, and $N'_k(j)$ is its complement. The change in values of the two objective terms must reflect the loss of connectivity among the

nodes in $N'_k(i)$ and $N'_k(j)$ due to the removal of (i, j). Formally,

$$\Delta A_{ij} = -\sum_{l \in N'_k(i)} \sum_{h \in N'_k(j)} a_{lh}$$

$$\Delta D_{ij} = -\sum_{l \in N'_k(i)} \sum_{h \in N'_k(j)} a_{lh} L_{lh}$$

(ii) The removal of arc (i, j) does not disconnect node i and node j in terms of all-season paved road travel. If the connectivity of the nodes in G'_k is preserved, the shortest paths among the nodes in G'_k need to be recomputed, since the loss of the dropped arc might cause an increase in the shortest path distances. ΔA_{ij} and ΔD_{ij} are defined as in formulas (5) and (6), but this time we have $L'_{lh} \geq L_{lh}$.

In order to recompute the shortest paths after the removal of one arc, we use the dual ascent approach for shortest path reoptimization as described in Pallottino e Scutellà (2003). The procedure is an adaptation of the "dual-hanging" approach described in Pallottino and Scutellà (1997) to the problem of finding the new shortest path tree when the cost of one arc is augmented. The choice of this procedure is motivated by the fact that the dual feasibility of the shortest path tree is preserved when the cost of an arc is increased (or equivalently, when an arc is removed and its cost is set to a very large value). If the primal feasibility is also preserved, meaning that the removed arc is not in the optimal tree, no reoptimization is needed. Otherwise, suitably designed *hanging* operations can be used to optimally reconnect the fragment of the optimal tree that might have been disconnected by the arc removal (i.e., to restore the primal feasibility). Preliminary results showed that the use of the dual ascent algorithm appreciably improved the computational time of the overall methodology as compared to the use of standard label-setting or label-correcting shortest path algorithms (Dreyfus, 1969). Also in this case, the reoptimization algorithm must be repeated to recalculate the shortest path tree rooted at each node in G'_k.

4. Path relinking implementation

At the end of each GRASP iteration, we apply path-relinking as an intensification strategy in the attempt to detect value-improving solutions around the local optima just generated. The relinking is performed by investigating the trajectory that starts from a locally optimal solution obtained after local search (the initiating solution) and directed towards a solution selected from a pool of high quality solutions previously found (the guiding solution). The trajectory is built by gradually transforming the current solution into the guiding solution through appropriate arc additions, deletions or swaps. To define the set of moves which should be applied, we first compute the symmetric difference between the initiating and guiding solutions, i.e., the set of arcs which are selected for pavement in one of them but not in the other. At each stage of the trajectory construction, a new solution is generated by performing a move that reduces the symmetric difference. The next move to be performed is

always chosen in a greedy fashion, i.e., it is the add, drop or swap move that results in a best possible improvement or a least possible deterioration of the objective function value. After the generation of each new solution, the benefits of all the remaining moves are evaluated again and the next move is selected, until the guiding solution is finally attained. The best solution found along the trajectory is returned by the procedure and evaluated for possible insertion in the pool of elite solutions.

The pool of elite solutions is progressively built and updated in the following way. Each solution obtained at the end of a GRASP iteration or returned by the path-relinking procedure is a candidate to be inserted in the pool, unless already present. Such a solution is automatically inserted if the maximum pool size has not yet been reached. If the pool is already full, the candidate is inserted only if it is better than the worst elite solution. In this case, the worst solution is removed from the elite set.

As for the selection of the guiding solutions, a simple and commonly used strategy is to select it at random from the pool of elite solutions. In our empirical investigation, we provide results for this strategy. Additionally, we tested an alternative scheme which favors solution diversity. Namely, we select the solution that is most different from the incumbent, breaking ties randomly. The importance of guaranteeing diversity among the solutions subjected to the path-relinking process was already emphasized by other authors (see for example Resende and Werneck, 2002). Solution diversity can be enforced when dealing with the pool by inserting only solutions which are sufficiently diverse from the other elite solutions, or, as in our approach, when selecting the guiding solution for a given initiating solution.

5. Computational experiments

In this section, we report on the computational results obtained with different implementations of the GRASP and path relinking approach. All of the algorithm implementations were coded in Visual Basic 6.0 and run on a PC with a Pentium 4/2.0GHz processor and 1GB of RAM. The proposed heuristics were tested on a set of 15 networks, generated as part of a project supported by the U.S.D.O.T. The village coordinates were uniformly generated within a bounding box, while the populations at each village were generated from a uniform distribution in [1,1000]. The Euclidean distances among nodes and the node population sizes were then used as input to a gravity model in order to estimate the traffic flow among villages. The rural road connections among villages were established through the use of a Graphical User Interface developed within the project framework. The pavement costs of the arcs were assumed to be proportional to the arc lengths. The problem set includes 5 classes of 3 problems each. Each problem in the same class has the same dimension, although different in layout. For example, the problems in the first class have 50 villages and 80 rural road segments; the problems in the remaining four classes have dimensions (75, 120), (100, 160), (125, 200), (150, 240), respectively. Each problem was solved for 3 budget levels and for 2 pairs of objective weighting coefficients. We hence conducted experiments on 18 instances for each class, resulting in a total number of 90 different problems. The budget levels were chosen to be equal to constant proportions of the total network construction costs. More specifically, proportions equal to 15%, 30% and 45% of

the total pavement cost were chosen. Based upon preliminary experiments, the weighting coefficient of the efficiency objective, w_2, was fixed to 1 while the weighting coefficient related to the connectivity objective, w_1, was fixed to 50 in one case and 100 in the other.

5.1. Parameter testing

A calibration phase was conducted to determine appropriate values of the key parameters and implementation options of the GRASP + PR approach. In particular, we experimented with different schemes for selecting the achievement level parameter α, with different path-relinking strategies and various pool size values. Two hundred independent runs were executed for each algorithm variant.

Tables 1 and 2 show the results obtained by varying the parameter α, when the pool size is fixed to 10 and the path-relinking is performed by using as a guiding solution, the most distant solution from the incumbent. Table 1 lists for each of the five classes the average deviation from the value of the best-known solution. The average is computed over the 18 instances of each class. For the first class (namely, for the problems with 50 nodes),

Table 1. Tests on the GRASP parameter α: Average percent deviation from best known solutions.

	Fixed α					Reactive GRASP			
Pr. Set	0.1	0.3	0.5	0.7	0.9	25	50	75*	100*
Pr50	0.007	0.022	0.047	0.089	0.146	0.039	0.001	0.001	0.001
Pr75	0.034	0.040	0.125	0.108	0.368	0.046	0.018	0.038	0.042
Pr100	0.055	0.134	0.098	0.154	0.559	0.141	0.120	0.067	0.072
Pr125	0.067	0.047	0.068	0.193	0.375	0.038	0.060	0.062	0.057
Pr150	0.171	0.165	0.259	0.413	0.597	0.115	0.120	0.093	0.110
Avg.	0.067	0.082	0.119	0.191	0.409	0.076	0.064	0.052	0.056

Options: DPR, $ps = 10$.

Table 2. Tests on the GRASP parameter α: Execution time (sec.).

	Fixed α					Reactive GRASP			
Pr. Set	0.1	0.3	0.5	0.7	0.9	25	50	75*	100*
Pr50	33	22	19	16	14	18	19	19	18
Pr75	143	79	60	47	38	65	65	66	65
Pr100	351	189	156	158	175	168	215	174	172
Pr125	765	376	300	271	245	339	418	336	336
Pr150	1796	751	608	595	465	709	897	688	672
Avg.	617	283	228	217	187	260	323	257	253

Options: DPR, $ps = 10$.

the best-known solution corresponds to the optimal solution found by CPLEX (for the IP formulation of COMBDP the reader is referred to Church and Scaparra, 2004). The larger problems in the following four sets could not be solved by CPLEX. Hence, the best-known solution for these problems is the best solution found in our computational investigation by some of our algorithmic variants.

In Table 1, the first column gives the problem set name, the following five columns list the average percent deviation when different fixed values of the parameter α were used, while the last section of four columns reports the percent deviation obtained with four different implementations of Reactive GRASP. The header of each of the last four columns refers to the parameter *block_iterations* used to implement the Reactive GRASP procedure (Prais and Ribeiro, 2000). The *block-iterations* parameter represents the number of iterations after which the probabilities, which guide the selection of a particular α value, are updated. For the update operation we use the same absolute qualification rule as described in Prais and Ribeiro (2000), with the suitable changes to reflect the maximization nature of our problem. We then introduce a variation to the general reactive scheme, which consists in using a variable *block-iterations* parameter. Namely, in the initial runs, we use a large value for *block-iterations* in order to have a better estimate of the goodness of each α choice. As the algorithm proceeds, we systematically reduce its value so as to intensify the use of good α values. Several different ways of scaling down the value of *block_iterations* were attempted. We only report the results for two implementations (indicated with an asterisk in the column header): in the first case (75*), we set the initial value to 75 and reduce it by 25 units every time we update the probability distribution; in the second case (100*), we initially set it to 100 and halve its value after each update, up to a minimum value of 25. The results in the first two Reactive GRASP columns refer to a standard implementation of Reactive GRASP with a fixed *block-iterations* set up to 25 and 50, respectively.

Table 2 has a similar structure, but displays the average computational times required by the different implementations for each problem set.

From the analyses of the two tables it can be noticed that when GRASP is implemented with a fixed α, the percent deviation clearly increases as bigger α values are used, meaning that in terms of solution quality the variants which favor randomness are to be preferred to the more greedy implementations. However, small values of α require a significantly higher computational time, which is especially spent by the local search phase to reach local optima from highly random solutions.

The variants using Reactive GRASP are clearly superior to the ones with fixed α. The dominance is not as noticeable for fixed *block-iterations* (columns indexed by 25 and 50). This is mainly due to the limited number of iterations (200) we could run without incurring an excessive amount of computational time and, consequently, to the lack of sufficient information to set-up a reliable self-adjusting selection mechanism. This limitation, however, can be partially bypassed by iteratively reducing the value of block-iterations, as previously described. Reactive GRASP with variable *block-iterations* (last two columns in Tables 1 and 2) turned out to be a good trade-off between computational time and solution quality. In particular, the option denoted by 75* yields the best average percent deviation, performing considerably well on the largest problems (set Pr150), and it is very competitive in terms of computational time. Furthermore, it is the implementation that finds the largest number

Table 3. Results for different path relinking strategies.

	% Deviation			Time (sec.)		
Pr. Set	NPR	RPR	DPR	NPR	RPR	DPR
Pr50	0.004	0.001	0.001	17	18	19
Pr75	0.127	0.050	0.038	59	62	66
Pr100	0.135	0.065	0.067	154	159	174
Pr125	0.251	0.101	0.062	300	321	336
Pr150	0.346	0.144	0.093	630	661	688
Avg.	0.173	0.072	0.052	232	244	257

Options: Reactive 75*, $ps = 10$.

of best-known solutions (55 out of the 90 problems) among all the variants tested. Hence, we selected the option 75* as the base case to test the other implementation options.

The second experiment was conducted to investigate the behavior of the path-relinking strategy. As already mentioned in Section 4, we consider two path-relinking variants, which re-link the incumbent solution to a solution randomly selected from the pool and to the most different solution, respectively. The pure GRASP variant without path relinking was also tested to provide evidence of the effective gain deriving from the use of path-relinking. In Table 3, the three algorithmic variants are referred to respectively as NPR (for no path-relinking), RPR (for random path-relinking), and DPR (for path-relinking based on maximal distance). It can be noted that there is a clear benefit in using path-relinking: it systematically leads to improvements in the solutions found and the computational effort associated with it is negligible when compared to a pure GRASP strategy. As for the selection of the guiding solution, the exploration of longer trajectories among solutions enforced by the DPR strategy generally produces better solutions at little extra computational effort, in accordance with the theoretical expectation.

Finally to study the sensitivity of the overall methodology to the dimension of the pool of elite solutions, we report the results obtained for different pool size (ps) values. Among the various values tested, we provide the results for pool sizes equal to 3, 10 and 20. Table 4 shows that a size of 10 is a good trade-off. With a smaller pool size ($ps = 3$), the average deviation deteriorates for each problem set and the saving in terms of computational time is minor. Handling a larger pool ($ps = 20$) requires additional computation effort (especially on large problems) but does not pay off in terms of solution quality (it leads to a marginal improvement only on the second set, but it is inferior in all of the other cases).

5.2. Final results

Detailed results for the best parameter combination (i.e., Reactive 75*, DPR and $ps = 10$) are provided in Tables 5 and 6. For each of the 3 problems in each problem set and for each combination of the budget values and weighting coefficients, Table 5 reports the percent deviation from the best known solutions and the execution time to run the 200

Table 4. Results for various pool size values.

	% Deviation			Time (sec.)		
Pr. Set	$ps = 3$	$ps = 10$	$ps = 20$	$ps = 3$	$ps = 10$	$ps = 20$
Pr50	0.004	0.001	0.001	18	19	19
Pr75	0.052	0.038	0.023	66	66	68
Pr100	0.092	0.067	0.078	169	174	176
Pr125	0.080	0.062	0.089	336	336	357
Pr150	0.138	0.093	0.139	694	688	810
Avg.	0.073	0.052	0.066	256	257	286

Options: Reactive 75*, *DPR*.

Table 5. Percent deviation from best known and execution time (sec.) for different budgets and objective weights.

			Pr50		Pr75		Pr100		Pr125		Pr150	
Pr. no.	B	w1	%Dev	Time	%Dev	Time	%Dev	Time	%Dev	Time	%Dev	Time
1	15	100	0.000	6	0.000	2	0.000	2	0.482	9	0.000	9
1	15	50	0.000	3	0.142	2	0.103	2	0.000	9	0.000	7
1	30	100	0.000	38	0.000	64	0.106	84	0.138	144	0.366	358
1	30	50	0.000	8	0.000	35	0.092	39	0.138	66	0.000	124
1	45	100	0.000	70	0.020	456	0.000	858	0.237	1508	0.000	6025
1	45	50	0.000	14	0.000	42	0.029	148	0.024	202	0.018	875
2	15	100	0.000	1	0.000	2	0.000	3	0.039	11	0.000	6
2	15	50	0.000	1	0.000	2	0.000	4	0.000	16	0.058	8
2	30	100	0.000	11	0.000	50	0.141	189	0.000	127	0.079	315
2	30	50	0.000	6	0.000	18	0.218	34	0.000	64	0.000	99
2	45	100	0.012	55	0.108	200	0.000	912	0.000	903	0.314	1580
2	45	50	0.000	12	0.000	35	0.000	108	0.008	219	0.033	205
3	15	100	0.000	0	0.338	2	0.000	6	0.000	13	0.073	11
3	15	50	0.000	1	0.000	1	0.000	8	0.000	9	0.485	7
3	30	100	0.000	26	0.051	38	0.000	87	0.000	617	0.079	260
3	30	50	0.000	6	0.000	24	0.000	36	0.000	76	0.046	130
3	45	100	0.000	70	0.021	174	0.236	538	0.000	1893	0.112	1983
3	45	50	0.000	11	0.000	35	0.280	67	0.048	170	0.004	383
Avg.			0.001	19	0.038	66	0.067	174	0.062	336	0.093	688

Table 6. Connectivity (%cov) and efficiency (vmt) objective values for different budgets and objective weights.

			Pr50		Pr75		Pr100		Pr125		Pr150	
Pr. no.	B	w1	%cov	vmt	%cov	vmt	%cov	vmt	%cov	vmt	%cov	vmt
1	15	100	82.42	9,532	59.86	12,684	44.39	15,494	50.14	28,337	45.87	28,954
1	15	50	79.92	7,665	58.63	10,480	42.18	11,542	49.05	24,230	45.61	27,892
1	30	100	95.94	16,819	85.50	31,815	77.69	50,802	70.66	55,825	77.04	120,571
1	30	50	87.71	10,325	78.40	20,813	68.05	30,126	64.90	35,706	67.81	65,297
1	45	100	98.85	17,701	99.55	45,483	96.41	72,545	92.72	114,241	97.15	179,959
1	45	50	89.57	10,850	83.35	22,514	80.13	38,831	76.52	51,942	80.25	89,780
2	15	100	69.95	3,700	58.47	7,213	55.43	15,500	50.45	21,974	52.36	23,988
2	15	50	67.75	2,987	58.47	7,213	52.79	10,510	49.41	18,486	51.19	20,274
2	30	100	86.95	8,083	81.95	17,980	82.47	40,015	75.44	54,250	73.22	58,242
2	30	50	78.57	4,801	76.09	12,743	69.28	18,954	69.37	36,817	66.53	35,682
2	45	100	93.42	9,510	99.61	32,073	96.01	52,351	89.87	75,246	89.65	96,880
2	45	50	84.93	6,058	80.47	13,823	76.07	22,080	78.41	44,962	75.40	47,749
3	15	100	55.73	3,965	58.54	9,551	51.33	14,339	62.73	34,367	51.46	23,493
3	15	50	54.47	3,515	56.63	7,782	48.54	10,848	59.79	26,130	48.49	17,285
3	30	100	88.94	12,924	81.68	22,218	77.11	32,363	89.01	81,108	75.20	53,090
3	30	50	72.31	6,132	73.59	13,398	67.57	19,279	75.93	42,255	66.98	32,564
3	45	100	96.21	13,471	91.40	28,269	95.74	51,680	96.30	87,499	93.35	87,493
3	45	50	75.58	6,393	79.86	15,950	76.15	23,478	81.59	47,266	76.02	40,660

The vmt measure is expressed in 000's units.

iterations, whereas Table 6 lists the values of the two objectives, for the best solution of each problem.

As shown in Table 5, all but one problem in the first set are solved to optimality by our approach with the chosen parameter setting. The problem complexity increases as more importance is given to the connectivity measure ($w_1 = 100$ vs. $w_1 = 50$), and, especially, as the budget level is increased. This observation suggests an alternative way of implementing the GRASP constructive phase when high budget levels are considered. Namely, we could start the constructive phase with a completely paved network and delete arcs according to a randomized greedy scheme, until the budget constraint is satisfied. Since the time-consumption of the algorithm is largely dependent on the number of network evaluations needed every time an arc is added or removed, for high budget availability this approach could reduce the running time associated to the GRASP phase.

The results in Table 6 demonstrate how connectivity can be traded for greater operational efficiency. As an example, consider the first problem in the set Pr100. With more importance placed on the first objective ($w_1 = 100$) and a budget of 15%, the traffic flow volume covered by paved roads equals around 44% of the total traffic flow on the network. By contrast, the solution obtained when more importance is assigned to the second objective ($w_1 = 50$),

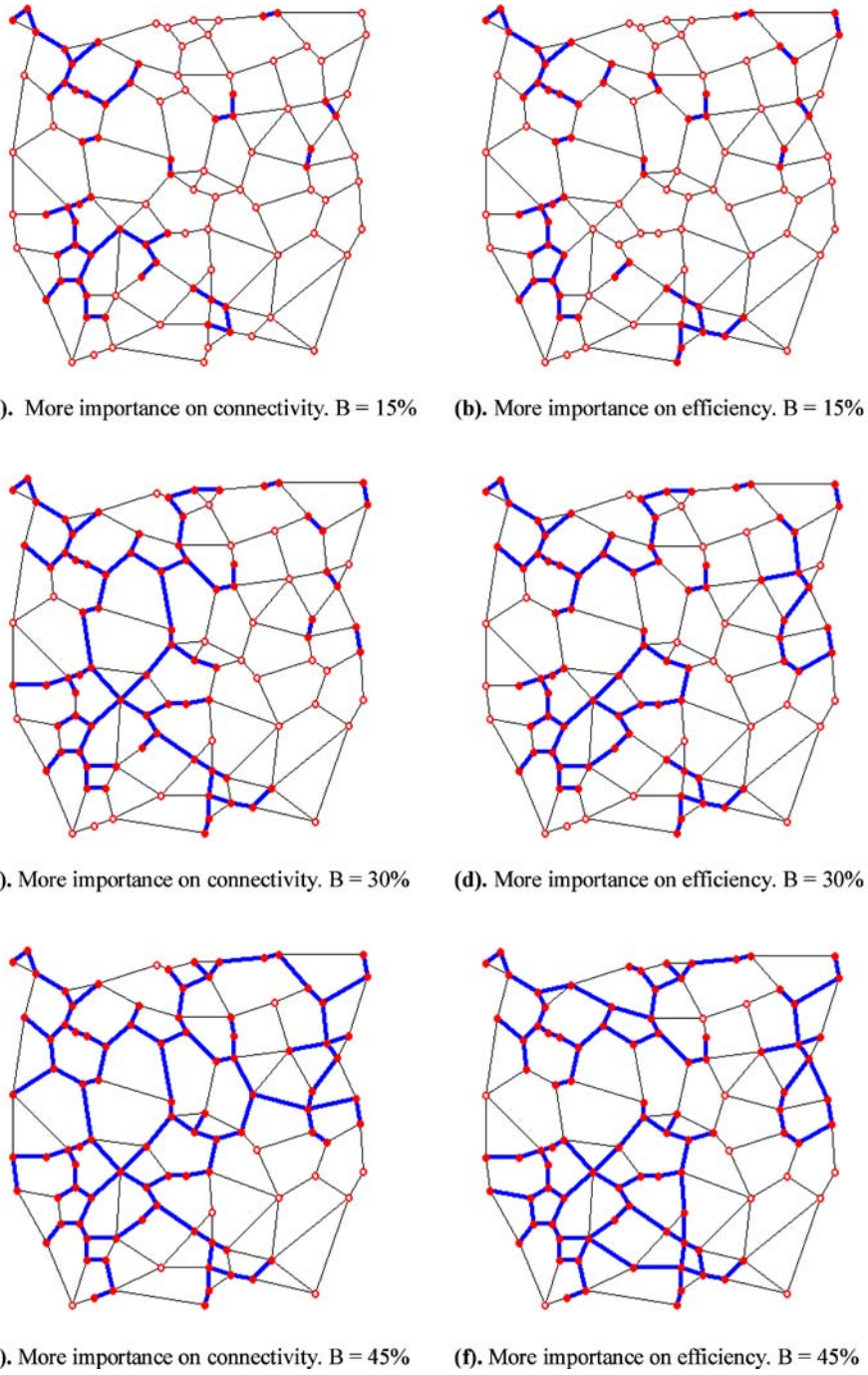

Figure 1. Connectivity vs. efficiency for different budget values for the first problem in Pr100.

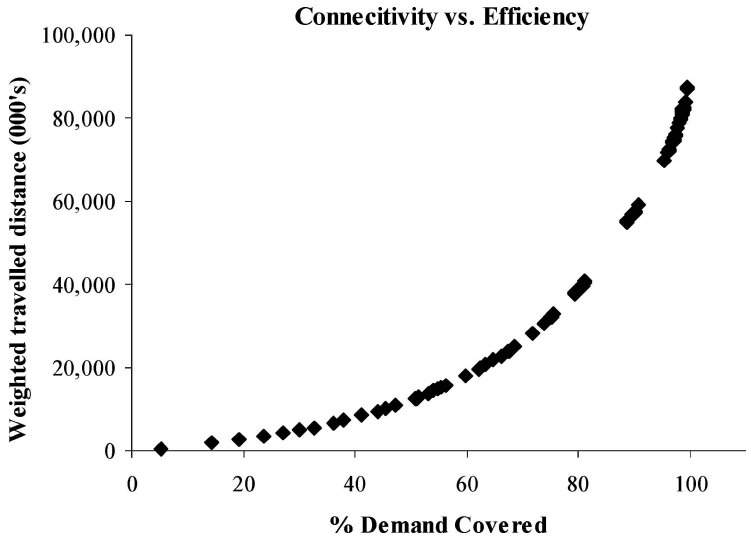

Figure 2. Pareto-near-optimal frontier for the first problem in Pr100 ($B = 45\%$)

covers only 42% of the traffic flow volume but improves on the efficiency measure by 25%. This trade-off is even more noticeable for increasing budget availability. For the same problem and a budget of 30%, a drop in the covered traffic flow from 78% to 68% yields a 40% improvement in network efficiency. With a budget of 45%, the network efficiency can be improved by almost 50% by sacrificing 15% of the traffic coverage.

The solutions corresponding to the 6 situations just described for the first problem in the set Pr100 are depicted in figure 1. In the picture, the dots represent villages. The thin lines depict dirt roads while the heavy lines show candidate paved sections. It can be noticed that when greater importance is placed on the efficiency measure (figures b, d and f), the approach tends to generate more disjoint components in the resulting road configuration. However, even if fewer interconnections are provided by paved roads, each cluster represents a more efficient road sub-network.

Finally, for the same problem, we display in figure 2 the Pareto near-optimal front, obtained by applying our algorithm a number of times with equi-spaced values of w_1 ranging from 5 to 200. In spite of a recognized deficiency of the weighted sum approach in generating unsupported non-dominated solution (see for example Koski, 1985), the method seems to work well for solving COMBDP. As evidenced in figure 2, the weighting method fails to populate only small sections of the Pareto boundary while producing efficient network solutions.

6. Conclusions and future research

In this paper we have presented a GRASP and Path-relinking strategy for finding approximate solutions to a budget network design problem when complete coverage of all the traffic

flow volume by all-season roads is not strictly required. We have shown how the computational efficiency of the approach can be improved by exploiting the structural characteristics of the solutions as well as by using specialized routines for shortest path reoptimization. Furthermore, we have enhanced the performance of a standard GRASP and Path Relinking implementation by introducing a new form of reactive GRASP which allows a more efficient parameter calibration, and by relinking the incumbent solution with the most different elite solution. Results on 90 problems with up to 150 nodes and 240 undirected links validate the effectiveness of these variants: all of the smaller problems for which optimality could be verified were solved to optimality; for larger problems, which cannot be solved by CPLEX, the enhanced implementation never exceeded a 0.5% deviation from the best known solutions in reasonable computational time.

Even though the model has been primarily devised for solving the rural road network design problem, its applicability is not confined to this problem. Other application settings exist where it is realistic, and beneficial, to sacrifice service provision to a segment of users in favor of more economical, operational and higher quality service to others.

A possible extension of the model includes an added objective that accounts for all traffic, in addition to all season traffic flow. For some developing countries it is adequate to consider efficiency as provided by good roads only, since road segments which are not paved or upgraded could be completely impassable or inaccessible during some parts of the year. However, for other road systems it could be reasonable to consider the efficiency of the overall network for each season. Finally, the plan for future research includes an extension of the methodology to account for multi-period planning.

Acknowledgment

We wish to acknowledge the support of the Research and Special Programs Administration of the U.S.D.O.T. (Contract DTRS56-00-T-0002). We wish also to thank our colleagues within the National Consortium on Remote Sensing in Transportation, Infrastructure for their helpful comments.

References

Abdulaal, M. and L.J. LeBlanc. (1979). "Continuous equilibrium network design models." *Transportation Research B* 13B, 19–32.

Aiex, R.M., M.G.C. Resende, P.M. Pardalos, and G. Toraldo. (2003). "GRASP with path relinking for the three-index assignment problem." *INFORMS J. on Computing* (to appear).

Antunes, A., A. Seco, and N. Pinto. (2003). "An accessibility-maximization approach to road network planning." *Computer-Aided Civil and Infrastructure Engineering* 18, 224–240.

Boyce, D.E., A. Farhi, and R. Weischedel. (1973). "Optimal network problem: A branch-and-bound algorithm." *Environmental and Planning* 5, 519–533.

Church, R.L. and M.P. Scaparra. (2003). A mixed integer programming formulation for a bi-objective rural road development problem. Working paper.

Current, J. and H. Min. (1986). "Multiobjective design of transportation networks: Taxonomy and annotation." *European Journal of Operational Research* 26, 187–201.

Dionne, R. and M. Florian. (1979). "Exact and approximate algorithms for optimal network design." *Networks* 9, 37–59.

Dreyfus, S.E. (1969). "An appraisal of some shortest path algorithms." *Operations Research* 17, 395–412.

Feng, C. and J.Y. Wu. (2003). "Highway investment planning model for equity issues." *Journal of Urban Planning and Development* 129(3), 161–176.

Feo, T.A. and M.G.C. Resende. (1989). "A probabilistic heuristic for a computationally difficult set covering problem." *Operations Research* Letters 8, 67–71.

Feo, T.A. and M.G.C. Resende. (1995). "Greedy randomized adaptive search procedures." *Journal of Global Optimization* 6, 109–133.

Festa, P. and M.G.C. Resende. (2001). "GRASP: An annotated bibliography." In C.C. Ribeiro and P. Hansen (eds.), *Essays and Surveys in Metaheuristics*, Kluwer Academic Publishers, pp. 325–367.

Gallo, G. (1980). "Reoptimization procedures in shortest paths problems." *Riv. Mat. Sci. Econom. Social.* 3, 3–13.

Glover, F. (1996). "Tabu search and adaptive memory programming—Advances, applications and challenges." In R.S. Barr, R.V. Helgason, and J.L. Kennington (eds.), *Interfaces in Computer Science and Operations Research*, Boston: Kluwer, pp. 1–24.

Glover, F. (1999). "Scatter search and path relinking." In D. Corne, M. Dorigo, and F. Glover (eds.), *New Ideas in Optimisation.* Wiley.

Glover, F. and M. Laguna. (1997). *Tabu Search*. Boston: Kluwer Academic Publishers.

Glover, F., M. Laguna, and R. Martí. (2000). "Fundamentals of scatter search and path relinking". *Control and Cybernetics* 39, 653–684.

Johnson, D.S., J.K. Lenstra, and A.H.G. Rinnooy Kan. (1978). "The complexity of the network design problem." *Networks* 8, 279–285.

Koski, J. (1985). "Defectiveness of weighting method in multicriteria optimization of structures." *Communications in Applied Numerical Methods* 1, 333–337.

Laguna, M. and R. Martí. (1999). "GRASP and path relinking for the 2-layer straight line crossing minimization." *INFORMS Journal on Computing* 11, 44–52.

Magnanti, T.L. and R.T. Wong. (1984). "Network design and transportation planning: Models and algorithms." *Transportation Science* 18(1), 1–55.

Meng, Q. and H. Yang. (2002). "Benefit distribution and equity in road network design." *Transportation Research Part B* 36, 19–35.

Pallottino, S. and M.G. Scutellà. (1997). "Dual algorithms for the shortest path tree problem." *Networks* 29, 125–133.

Pallottino, S. and M.G. Scutellà. (2003). "A new algorithm for reoptimizing shortest paths when the arc costs change." *Operations Research Letters* 31, 149–160.

Prais, M. and C.C. Ribeiro. (2000). "Reactive GRASP: An application to a Matrix Decomposition Problem in TDMA traffic assignment." *INFORMS Journal on Computing* 12(3), 164–176.

Resende, M.G.C. (1998). "Computing approximate solutions of the maximum covering problem using GRASP." *Journal of Heuristics* 4, 161–171.

Resende, M.G.C. and C.C. Ribeiro. (2002). "Greedy randomized adaptive search procedures." In F. Glover and G. Kochenberger (eds.), *State-of-the-Art Handbook of Metaheuristics*, Kluwer Academic Publishers, pp. 219–249.

Resende, M.G.C. and C.C. Ribeiro. (2003). "GRASP with path-relinking for private virtual circuit routing." *Networks* 41, 104–114.

Resende, M.G.C. and R.F. Werneck. (2002). "A GRASP with path-relinking for the p-median problem." Technical Report, AT&T Labs Research, Florham Park, NJ.

Ribeiro, C.C., E. Uchoa, and R.F. Werneck. (2002). "A hybrid GRASP with perturbations for the Steiner problem in graphs." *INFORMS Journal on Computing* 14, 228–246.

Sheffi, Y. (1985). Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods, Englewood Cliffs, NJ: Prentice-Hall.

Yang, H. and M.G.H. Bell. (1998). "Models and algorithms for road network design: A review and some new development." *Transportation Reviews* 18(3), 257–258.