Ann Oper Res (2010) 181: 661–681
DOI 10.1007/s10479-010-0800-4

RAMP for the capacitated minimum spanning tree problem

Cesar Rego · Frank Mathew · Fred Glover

Published online: 19 October 2010

Abstract This paper introduces dual and primal-dual RAMP algorithms for the solution of the capacitated minimum spanning tree problem (CMST). A surrogate constraint relaxation incorporating cutting planes is proposed to explore the dual solution space. In the dual RAMP approach, primal-feasible solutions are obtained by simple tabu searches that project dual solutions onto primal feasible space. A primal-dual approach is achieved by including a scatter search procedure that further exploits the adaptive memory framework. Computational results from applying the methods to a standard set of benchmark problems disclose that the dual RAMP algorithm finds high quality solutions very efficiently and that its primal-dual enhancement is still more effective.

Keywords Minimum spanning tree · Heuristics · Surrogate constraints · Scatter search · Tabu search · RAMP

1 Introduction

The capacitated minimum spanning tree (CMST) problem is fundamental to the design of communication networks, and has been widely studied for its importance in practical applications. A classical application consists of finding a minimum cost design of a capacitated centralized processing network where a *central node* of limited capacity must be linked via a tree topology to a number of *remote terminals* with a specified demand. The CMST also finds application in a variety of other settings in distribution, transportation and logistics

C. Rego (✉) · F. Mathew
School of Business Administration, University of Mississippi, University, MS 38677, USA
e-mail: crego@bus.olemiss.edu

F. Mathew
e-mail: fmathew@bus.olemiss.edu

F. Glover
University of Colorado, Boulder, CO 80309-0419, USA
e-mail: fred.glover@colorado.edu

(see Gavish 1982, 1991). The problem also provides a relaxation of the classical capacitated vehicle routing problem, which is central to many other complex problems, including the design of communications networks with topological ring structures (Klincewicz et al. 1998).

The classical CMST problem can be stated in reference to a complete undirected graph $G = (V_0, A)$, whose vertex (node) set is represented by $V_0 = \{0, 1, \ldots, n\}$ and whose arc set is represented by $A = \{(i, j) | i, j \in V; i \neq j\}$. Node 0 denotes a special node named *root* with demand $d_0 = 0$, and each node $i \in V = V_0 \backslash \{0\}$ has a specified integer demand $d_i > 0$. A matrix $C = (c_{ij})$ is associated with A, where c_{ij} is a non-negative weight (distance or cost) for arc (i, j) if there is link between nodes i and j. Otherwise c_{ij} is infinity. The CMST problem consists in finding a minimum cost of a tree T spanning all nodes of G, so that the sum of the demands in each subtree off the root node does not exceed a fixed integer arc capacity Q. When all the nodes $i \in V$ have the same demand the problem is referred to as the homogeneous demand CMST problem. Similarly, when the nodes have different demand the problem is referred to as the heterogeneous demand CMST problem.

In this paper, we address the CMST with heterogeneous demand. The CMST problem has been shown to be NP-Complete (Papadimitriou 1978). Several formulations have been developed for the problem. Similar to Gavish (1982) the CMST can be formulated as:

$$(P_1) \quad \text{Minimize} \quad \sum_{i=0}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{i=0}^{n} x_{ij} = 1, \quad j = 1, \ldots, n \tag{2}$$

$$\sum_{i=0}^{n} y_{ij} - \sum_{i=1}^{n} y_{ji} = d_j, \quad j = 1, \ldots, n \tag{3}$$

$$d_j x_{ij} \leq y_{ij} \leq (Q - d_i) x_{ij}, \quad i = 0, \ldots, n; \; j = 1, \ldots, n \tag{4}$$

$$y_{ij} \geq 0, \quad i = 0, \ldots, n; \; j = 1, \ldots, n \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad i = 0, \ldots, n; \; j = 1, \ldots, n \tag{6}$$

which takes the form of a single-commodity flow problem. Binary variables x_{ij} equal 1 if the arc (i, j) is in the solution, and 0 otherwise. Integer variables y_{ij} represent the amount of flow through the arc $(i, j) \in A$. The objective (1) is to select a CMST having minimum cost, or equivalently that minimizes the sum of the costs of arcs included in the solution. The constraints of (2) ensure that each node $j \in V$ is sourced by exactly one arc (i, j) from some node $i \in V_0$. Conservation of flow within the network is accounted for by (3) while (4) assures that flow on an arc cannot exceed its capacity if the arc is open (chosen to belong to the solution) and equals 0 if the arc is closed. In the case of unitary demands, a formulation for the homogeneous-demand CMST is obtained by setting $d_j = 1$ in problem P_1.

Although numerous heuristic algorithms have been proposed for the CMST, the best methods to date still have difficulty finding solutions of highest quality. Currently, the most effective heuristic algorithms for the CMST are due to Amberg et al. (1996), Sharaiha et al. (1997), and notably Ahuja et al. (2003). The next section gives a brief review of heuristic algorithms for the CMST and a comprehensive review can be found in Mathew and Rego (2006).

The relaxation adaptive memory programming (RAMP) approach is a relatively new metaheuristic that has proved effective in the solution of a variety of difficult combinatorial optimization problems. The method incorporates principles of adaptive memory programming as introduced in tabu search and takes advantage of mathematical relaxation approaches to exploit primal and dual relationships. A description of the method and its applications appears in Rego (2005).

Just as Memetic Algorithms refer to hybrid methods composed of evolutionary methods and local search, RAMP refers to hybrid methods composed of mathematical relaxation methods and adaptive memory programming. In the same way that numerous challenges arise in establishing an effective interaction between evolutionary processes and local search, a variety of intriguing challenges present themselves in establishing an effective combination of mathematical relaxation and adaptive memory. Grünert (2002) extends the classical Lagrangian-based heuristic framework with tabu search adaptive memory with the purpose of restricting the neighborhood space to some constructive projection method. With the same purpose of reducing the neighborhood size, Yagiura et al. (2006) use information from Lagrangian relaxation for fixing variables in an adaptive memory local search algorithm. A more elaborate integration of mathematical relaxation and adaptive memory is developed in Caserta (2007), which explores primal-dual interactions by cross-linking tabu search and Lagrangian-based subgradient search within a RAMP framework.

In this paper we identify ways we have been able to exploit the RAMP framework to produce a method that proves particularly useful for solving CMST problems. Specifically, we propose two RAMP algorithms for the CMST problem aimed at exploring different levels of sophistication. The simpler approach considers a dual-based RAMP variant that employs a surrogate constraint relaxation to create and project dual solutions onto the primal feasible space. A more advanced RAMP variant incorporates a scatter search procedure to create a primal-dual RAMP algorithm (PD-RAMP). Adaptive memory is used in integrating the components of both methods.

The remainder of this paper is organized as follows. In Sect. 2, we briefly review algorithms for CMST. Section 3 gives an overview of the RAMP method. Sections 4 to 7 describe the various component methods of the proposed dual and primal-dual RAMP algorithms, followed by their implementation details in Sect. 8. Section 9 provides a computational analysis. Summary and concluding remarks are presented in Sect. 10.

2 Previous approaches for CMST

The CMST problem has been widely studied over the last four decades with proposals of different formulations and a variety of specialized algorithms. Greedy *constructive methods* are the first to appear in the literature. For about thirty years a simple savings heuristic proposed by Esau and Williams (1966) had been a standard for providing approximate solutions to the CMST. Starting from a solution with all nodes directly connected to the root, the method operates by disconnecting a node from the root at each step and reconnecting it to another node so as to obtain a maximum possible savings in cost. The process is repeated until no further improvement is possible. By contrast, the constructive method of Elias and Ferguson (1974) begins with a solution for the MST relaxation of the CMST and iteratively attempts to satisfy each of the capacity constraints using the minimum increase in cost. This heuristic was later used by Gavish (1983) as a heuristic projection method for a Lagrangian relaxation approach.

A *branch-and-bound* algorithm with Lagrangian relaxation was presented by Malik and Yu (1993), which makes use of a stronger set of valid inequalities aimed at getting better

bounds during the optimization process. An interesting formulation for the homogeneous demand CMST was proposed by Gouveia (1995). For a problem with n nodes, the formulation is bounded by $O(n)$ constraints, making it more suitable for relaxation procedures than more traditional formulations (such as (P_1)) involving $O(n^2)$ constraints. Gouveia also proposed Lagrangian relaxation schemes that improved bounds by Gavish (1985) across several problem instances, especially those having smaller levels of arc capacity Q. Hall (1996) applied polyhedral methods to CMST and improved lower bounds for problem instances with root node in the center. A more powerful approach that undertakes to generate improved lower bounds has recently been considered by Uchoa et al. (2008).

In the latter half of the 90's *simulated annealing* and *tabu search* metaheuristics came into play to provide significantly better results than previous approaches. The neighborhood structure used during the search has a major influence on the performance of these algorithms. Amberg et al. (1996) had surprising success with two very basic neighborhoods: a *shift* neighborhood that transfers a node from one sub-tree to another, and a *swap neighborhood* that interchanges nodes between subtrees. The authors proposed and tested these two neighborhoods within a simulated annealing approach and a tabu search approach that were used to improve initial feasible solutions provided by the Esau-Williams heuristic. Sharaiha et al. (1997) proposed another tabu search approach based on a *subtree neighborhood* structure, which splits the current spanning tree into two subtrees and reconnects them by adding an arc different from the one that had been deleted in the original tree. The reconstruction of the new spanning tree may involve some arc reversals in order to maintain proper directions of flow. These neighborhoods that modify at most two subtrees are generally called two-exchange neighborhoods.

Patterson et al. (1999) proposed an *adaptive reasoning technique* drawing on principles of adaptive memory programming of the type used in tabu search and constructive neighborhood search processes. In their approach, *constructive neighborhoods* are obtained by iteratively executing Esau-Williams heuristic subjected to current tabu restrictions, which are probabilistically modified at each iteration of the method.

The state-of-the-art in the literature is provided by a sequence of two papers by Ahuja et al. (2001, 2003). In the initial paper, the authors proposed two *multi-exchange* neighborhood structures by generalizing the node-based and tree-based *two-exchange* neighborhood structures previously proposed by Amberg et al. (1996) and Sharaiha et al. (1997), respectively. The generalization of these methods considers larger neighborhoods that may propagate over all sub-trees in the solution, the size of which grows exponentially with problem size, making them extremely costly to evaluate even for problems of modest size. To overcome this limitation, the authors consider a reduced neighborhood using the concept of an *improvement graph*. An arc in the improvement graph with respect to a feasible solution represents an elementary move that transfers a node (or subtree) to join another subtree, with the cost of the arc being the change in cost associated with the move. Hence, a sequence of arcs tracing a node-simple path or cycle in the improvement graph can represent multi-exchanges of nodes or subtrees of the original graph. To evaluate the best multi-exchange move a technique based on shortest path constructions is used. The authors implemented tabu search as well as a greedy randomized adaptive search procedure (GRASP) for both neighborhoods and report the results from all four algorithms. Between the four different approaches, the authors obtained the best known solutions for all the benchmark problems. In the second paper, the authors proposed a unified neighborhood that integrates the node-based and tree-based multi-exchange neighborhoods to form a composite neighborhood along with a GRASP procedure. This approach further improved solutions for 36% of the standard benchmark problems.

3 The RAMP method: conceptual foundations and overview

Broadly speaking, adaptive memory programming (AMP) refers to processes using flexible memory structures and associated strategies for exploiting them as originally in the tabu search method (see, e.g., Glover 1989a, 1989b, 1996). AMP has been the key to many important algorithmic developments and appears as a major strategy in the creation of enhanced hybrid approaches utilizing tabu search components. Notable instances of such methods are provided by recent developments in scatter search and its generalization, the path-relinking approach. Together with RAMP, these algorithms form a class of methods generally called *adaptive memory metaheuristics*. While scatter search and path-relinking are typically primal methods that explore the solution space of the original problem, the RAMP method focuses on exploring the dual space associated with a relaxed problem and establishing primal-dual connections to overcome the duality gap.

Conceptually, the RAMP method is founded on the following premises:

(P1) The solution of appropriate relaxation dual problems affords relevant insights for the creation of adaptive memory structures by gathering information that cannot be obtained by primal based approaches.
(P2) The use of adaptive memory strategies that affect both sides of the primal-dual connection provides a useful means for bridging the duality gap that exists in combinatorial optimization.
(P3) A method that effectively gathers information from the primal and dual sides fulfills the adaptive memory programming concept of a method that learns as it progresses and may suitably provide a unified framework for solving difficult combinatorial optimization problems.

Operationally, the RAMP method comprises two fundamental components, a dual search that explores the solution space of an associated relaxation problem and a projection method that projects a dual feasible solution onto the primal solution space. The dual component includes a relaxation technique and a method for determining new dual values (weights or multipliers) for the relaxation. The projection method is designed to incorporate a heuristic improvement process to create enhanced primal feasible solutions. Adaptive memory is employed to take advantage of information generated from the dual and primal components. On one hand, the dual approach coupled with the associated projection method provides a means to create primal solutions by generating paths from the dual to the primal feasible space. On the other hand, the primal approach influences the search for new dual solutions by producing the feasible solutions and associated information for launching this search. Any time a primal search is completed a new relaxation problem is created to initiate another dual search. The structure of each relaxation problem depends on the current state of the search and is devised by appropriate memory structures that leverage the oscillation process between primal and dual spaces. The method alternates between the primal and the dual components until a specified stopping criterion is met. In advanced forms of the method, primal and dual solutions are combined to generate offspring solutions in an evolutionary fashion, providing another device to create adaptive memory structures.

From an implementation perspective, the RAMP method is organized around four main component methods that can be briefly described as follows:

An Adaptive Memory Relaxation Method—Create a relaxation problem based on the current relaxation parameters and the selected constraints set and solve it to obtain a dual solution and associated bound. Depending on the complexity of the relaxation problem, the underlying solution method can be either exact or heuristic. Within these options, the

method may encompass multiple levels of relaxation as exemplified by the cross-parametric relaxation method (Rego 2005).

An Adaptive Memory Projection Method—Apply a projection method to transform infeasible solutions into feasible and enhanced solutions. The method may be used to project dual solutions onto the primal solution space or to achieve feasibility for infeasible solutions encountered in search paths originating from primal feasible solutions. A variety of possibilities exist to create projection methods that take advantage of adaptive memory—see Rego (2005) and Glover (2005) for examples of those options.

An Adaptive Weighting Update Method—Compute weights for the dual search based upon memory structures that appropriately account for the effect of primal-dual interactions.

An Adaptive Solution Combination Method—Combine solutions originating from primal and dual searches in order to generate new solutions containing information from both search spaces.

The specific design of these methods for the capacitated minimum spanning tree problem is given in the following sections.

4 Adaptive memory relaxation method

In our design for the CMST the adaptive memory relaxation method uses surrogate constraint relaxation combined with cutting planes. The $2n$-constraint formulation proposed by Gouveia (1995) for the homogeneous-demand CMST is convenient for a surrogate constraint approach due to its reduced number of constraints. We first generalize this formulation in Sect. 4.1 to solve the general CMST with heterogeneous demand. In Sect 4.2 we discuss important properties of a basic surrogate relaxation that led us to the development of the enhanced surrogate relaxation with cutting planes developed in Sect. 4.3.

4.1 Generalized $2n$-constraint formulation for CMST

As shown earlier the classic formulation for the CMST problem (Gavish 1982) involves n^2 variables and $2n^2$ constraints (not counting the basic assignment and bounding constraints), which leads to rather large integer programming problems even for moderate values of n. Gouveia (1995) introduced an alternative formulation that involves n^2Q variables but only $2n$ constraints. This feature, combined with the ability to produce the same lower bound as the LP relaxation of classical CMST formulations, makes Gouveia's formulation advantageous for surrogate relaxation. While originally specialized for the homogeneous-demand variant of the CMST problem, we developed a generalization of this formulation to tackle the more general heterogeneous-demand CMST problem:

$$(P_2) \quad \text{Minimize} \quad \sum_{i=0}^{n}\sum_{j=1}^{n}\sum_{q=d_j}^{Q-d_i} c_{ij} Z_{ijq} \tag{7}$$

$$\text{subject to} \quad \sum_{i=0}^{n}\sum_{q=d_j}^{Q-d_i} Z_{ijq} = 1, \quad j = 1,\ldots,n \tag{8}$$

$$\sum_{i=0}^{n}\sum_{q=d_j}^{Q-d_i} q Z_{ijq} - \sum_{i=1}^{n}\sum_{q=d_i}^{Q-d_j} q Z_{jiq} = d_j, \quad j = 1,\ldots,n \tag{9}$$

$$Z_{ijq} \in \{0,1\}, \quad i = 0,\ldots,n;$$

$$j = 1, \ldots, n;\ q = d_j, \ldots, (Q - d_i) \tag{10}$$

The three dimensional binary variable Z_{ijq} indicates the existence ($Z_{ijq} = 1$) or absence ($Z_{ijq} = 0$) of a specific quantity of flow q on an arc (i, j) associated with the solution for CMST. Note that in the original formulation q ranges from 1 to $(Q - d_i)$, where d_i is the demand of node i. In our general case q is allowed to take any value from d_j to $(Q - d_i)$, as the minimum amount of flow on an arc has to equal the demand of the node it serves. The constraints of (8) ensure that each node $j \in V$ is sourced by exactly one arc (i, j) from some node $i \in V_0$, while conservation of flow is ensured by the constraints of (9). Also note in (9) that the RHS is modified to be d_j instead of 1.

This formulation forms the basis of the surrogate relaxation problem discussed next.

4.2 The basic surrogate constraint relaxation

We begin by discussing some special properties of the basic surrogate relaxation that are relevant for an effective design of a dual RAMP approach. The constraint set (9) is much harder to satisfy than (8), hence a surrogate constraint (9′) is formed by creating a linear combination of the constraint set (9) using a nonnegative vector of weights w. If the variables Z_{ijq} are kept integer, the problem of finding a feasible solution to the surrogate equality constraint itself corresponds to the well-known *subset sum problem*, which is NP-Complete, both for integer coefficients (Garey and Johnson 1979) as well as for real coefficients (Orús 2005). In the general case the subset sum problem may not have a solution. In the present instance, however, the surrogate constraint derives from a linear combination of the flow-conservation constraints system, which must have a binary solution for any possible CMST problem, and hence the associated subset sum problem is guaranteed to have a solution. Moreover, optimizing over constraints (8) and (9′) for an optimal binary solution can be a complex task; therefore the integrality constraints are also relaxed. These considerations lead to the following surrogate problem:

$$(S_w) \quad \text{Minimize} \quad \sum_{i=0}^{n} \sum_{j=1}^{n} \sum_{q=d_j}^{Q-d_i} c_{ij} Z_{ijq} \tag{7}$$

$$\text{subject to} \quad \sum_{i=0}^{n} \sum_{q=1}^{Q-d_i} Z_{ijq} = 1, \quad j = 1, \ldots, n \tag{8}$$

$$\sum_{j=1}^{n} w_j \left[\sum_{i=0}^{n} \sum_{q=d_j}^{Q-d_i} q Z_{ijq} - \sum_{i=1}^{n} \sum_{q=d_i}^{Q-d_j} q Z_{jiq} - d_j \right] = 0, \tag{9′}$$

$$0 \le Z_{ijq} \le 1, \quad i = 0, \ldots, n;\ j = 1, \ldots, n;$$

$$q = d_j, \ldots, (Q - d_i) \tag{11}$$

The structure of this surrogate problem has some interesting characteristics. It corresponds to a semi-assignment problem with an additional (equality) knapsack constraint. It is easy to show that the semi-assignment problem alone satisfies the *integrality property* and therefore admits an integer (all binary) optimal solution. Such a solution can be simply determined by selecting the minimum cost variable with a non-zero coefficient in the corresponding assignment constraint. If such a solution also satisfies the knapsack constraint, the optimal solution for the complete surrogate problem will be all binary. Otherwise, to satisfy the

knapsack constraint, some variables will need to take fractional values. Since every variable appears in exactly one of the assignment constraints, assigning one variable a fractional (non-integer) value necessarily requires another variable to take a fractional value in the same constraint. Clearly, relaxing the integrality requirement of variables in an assignment constraint can only increase (or possibly not change) the cost of the solution. Because it is always possible to satisfy the knapsack constraint by relaxing the integrality of variables that lie in a single assignment constraint, an optimal solution for this surrogate problem is either binary or contains fractional values for exactly two variables. A binary optimal solution is only possible, however, in a very remote situation where the relaxation problem has no integrality gap. (The integrality gap here is defined to be the worst-case difference between the cost of the IP solution and that of its LP relaxation solution.) Consequently, a solution to the present surrogate problem typically contains $n - 1$ variables set to 1, two variables set to fractional values that together sum to 1, and all remaining variables set to 0.

4.3 An enhanced surrogate constraint relaxation

In the context of the RAMP approach we are mostly interested in relaxations that can produce diverse dual solutions and that at the same time correspond to potentially good starting points for the projection/improvement method. In our algorithm this is achieved by extending the basic surrogate problem with appropriate valid inequalities for the CMST. By changing the problem structure we must give up the convenience of having solutions with only two variables taking fractional values and therefore a somewhat more complex projection method is needed to create primal feasible solutions. We will come back to this issue in Sect. 5 when describing the projection and improvement methods. We first consider the type of valid inequalities used to meet the goal of generating good and diverse dual solutions.

Valid inequalities (or cutting planes) have been extensively used to facilitate the solution of integer programming problems. Such inequalities, which seek to reduce the solution space while not "cutting off" (excluding) the optimal solution, are generally more effective when problem specific. Their usefulness depends on the problem structure, the type of formulation, the solution method, and associated strategies that determine which and when the available valid inequalities should be added. Commonly, valid inequalities are added directly to the LP relaxation of the integer linear programming (ILP) problem, which is used explicitly or implicitly in Lagrangian-based methods, or within branch-and-cut algorithms. Several classes of these inequalities involve very large (sometimes exponential) number of constraints and therefore their use requires implicit generation schemes or specialized separation methods, generally called relax-and-cut methods. In cases where the separation process is computationally infeasible, heuristic methods are often used to ensure polynomial time separations. See for example Gavish (1985), Malik and Yu (1993), Zhang (1993), Gouveia (1995), and Hall (1996) for an overview of valid cuts for the CMST and associated solution methods.

Our purpose in using valid cuts is to create a surrogate relaxation with a convenient structure for a dual search within the RAMP algorithm rather than striving to close the duality gap. Also, we want to keep the relaxation problem as simple as possible for a fast computation of the corresponding optimal solution. Bearing these objectives in mind, and conducting experimental analyses with different sets of inequalities derived from Gouveia (1995) and Hall (1996), we devised the following enhanced surrogate relaxation:

$$(S_w^+) \quad \text{Minimize} \quad \sum_{i=0}^{n}\sum_{j=1}^{n}\sum_{q=d_j}^{Q-d_i} c_{ij} Z_{ijq} \tag{7}$$

$$\text{subject to} \quad \sum_{i=0}^{n}\sum_{q=d_j}^{Q-d_i} Z_{ijq} = 1, \quad j = 1, \ldots, n \tag{8}$$

$$\sum_{j=1}^{n} w_j \left[\sum_{i=0}^{n}\sum_{q=d_j}^{Q-d_i} q Z_{ijq} - \sum_{i=1}^{n}\sum_{q=d_i}^{Q-d_j} q Z_{jiq} - d_j \right] = 0 \tag{9'}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} Z_{ijq} \le \left\lfloor \frac{\sum_{i=1}^{n} d_i}{q+1} \right\rfloor, \quad q = \lceil Q/2 \rceil, \ldots, Q-1 \tag{12}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{q=d_j}^{Q-d_i} Z_{ijq} \le n - \left\lceil \frac{\sum_{i=1}^{n} d_i}{Q} \right\rceil, \tag{13}$$

$$0 \le Z_{ijq} \le 1, \quad i = 0, \ldots, n; \; j = 1, \ldots, n; \; q = d_j, \ldots, (Q - d_i) \tag{11}$$

Inequalities (12) derive directly from the use of the three-index Z_{ijq} variables and have the particular advantage of adding only $\lceil Q/2 \rceil - 1$ constraints to the relaxation problem. In Gouveia (1995) the variant of these inequalities for the homogeneous-demand case were stated to be most effective at improving the lower bounds when added to a Lagrangian relaxation of the original formulation. We have found a similar impact when adding our generalization of these inequalities for the heterogeneous-demand case as cutting planes to our surrogate relaxations. Inequalities (13), which set a lower bound on the number of arcs incident to the root, denote a special case of the general bin-packing constraints that were also found useful in Hall (1996). We found empirically that these cutting planes when incorporated in the RAMP algorithm not only improve the diversification of the dual solutions being generated but also have a significant impact on the quality of the primal solutions. Because these constraints can be massive in number, implicit generation is used in order to keep the problem size under control.

The solution for the surrogate problem provides a lower bound for the optimal value of the original CMST problem and a starting solution for the projection method.

5 Adaptive memory projection method

The solution provided by the dual (or relaxation) method is usually infeasible for the original problem, and hence some type of projection method is necessary to project solutions from the dual solution space onto the primal feasible space. In the context of the RAMP method this is typically achieved by some sort of heuristic process aimed at finding enhanced feasible solutions in the vicinity of the infeasible one. In the present implementation a dual solution is first projected into primal feasible space with a greedy constructive approach and then improved using a simple tabu search procedure. Similar type of infeasibility is also likely to occur with the solutions generated by the adaptive solution combination method discussed later; hence the situation is handled the same way.

5.1 Projection

The greedy approach is a three step process to achieve feasibility. First violations of the integrality constraint are handled, then a tree structure for the solution is ensured and finally

arc capacities are adjusted to overcome possible demand violations. If more than n variables are set to non-zero values in the dual solution, due to the presence of the constraint set (8) in the surrogate problem we know that integrality is violated. With the objective of setting exactly n variables to 1 and the rest to 0, we order all non-zero variables so that $c_1/\mu_1 \leq c_2/\mu_2 \leq \cdots \leq c_k/\mu_k \leq \cdots \leq c_r/\mu_r$, where c_k denotes the objective function coefficient of the variable of order k and μ_k denotes the corresponding coefficient in the surrogate constraint. The variables whose coefficients represent the n smallest ratios are set to 1 while the others are set to 0.

Relaxation of the constraint set (9) may result in the presence of cycles in the solution instead of producing the desired tree structure. Such cycles are eliminated by closing the most expensive arc in a cycle and connecting the corresponding set of nodes to the root by an arc of minimum cost. Finally, the capacity constraints are satisfied by using a method analogous to that by Elias and Ferguson (1974). The sub-trees that violate capacity constraints are iteratively fixed. This is done by strategically closing an arc to create a disjoint sub-tree and then opening an arc between the root node and the this disjoint sub-tree with a minimum possible increase in cost. This process is repeated until all arcs in the sub-tree have flow that does not exceed Q. The resulting solution is a primal feasible solution. This solution is then subjected to a tabu search procedure for further improvement.

5.2 Improvement

The improvement component of the adaptive projection method is restricted to a simple tabu search procedure that operates on each solution projected onto the primal feasible space with the objective of finding an enhanced feasible solution. In this framework, adaptive memory is chiefly maintained by two elements: (1) tabu restrictions (deriving from moves performed in previous iterations of the tabu search), and (2) the structure of the dual solution (provided by the adaptive memory relaxation method). This memory additionally takes account of the values of the lower and upper bound maintained throughout the search. The appropriate integration of this information makes up the memory structures used to drive the search back to the dual space.

An important aspect of a tabu search implementation is the neighborhood structure used to explore the solution space during iterations of the method. In contrast to the neighborhood structures used by Amberg et al. (1996), Shariaha et al. (1997) and Ahuja et al. (2003), we employ an *oscillating neighborhood structure* that proves well suited for use with the tabu search method to explore the neighborhood space efficiently and effectively. Following a multi-neighborhood design used in scheduling by Glover and McMillan (1986), the oscillating strategy consists of alternating between node-based and tree-based neighborhoods whenever one neighborhood fails to improve the best available solution for a fixed number of iterations. Accordingly, short-term memory is used within each neighborhood exploration by employing classical tabu restrictions on the moves to inhibit the possibility of cycling and add vigor to the search, while critical event memory, which keeps track of the number of successive failures of each neighborhood, is used to control the oscillation process. The method stops when both neighborhoods fail to produce an improvement of the current best local optimum.

6 Adaptive weighting update method

A fundamental component of the RAMP method is the interaction between the primal and dual components. Rather than relying solely on searching the primal solution space as in

customary metaheuristic approaches or strictly searching the dual space as in traditional relaxation-based approaches, the RAMP method focuses on developing search paths in a global primal-dual space. While the adaptive projection method develops paths from the dual space to the primal space the adaptive weighting update method is concerned with projections in the opposite direction, from the primal to the dual space. The aim is that both types of projections are guided by the information gathered from both spaces that best reflects the state of the search.

In the present RAMP implementation adaptive weighting is conceived by means of the subgradient method. Subgradient optimization (Polyak 1969) is a standard approach used to solve the Lagrangian dual. It has also been used successfully for finding weights to surrogate constraints (e.g. Karwan and Rardin 1984; Lorena and Narciso 1999). Using the surrogate problem described in Sect 4.3, dual solutions for each iteration within the RAMP framework are generated as follows. For the set of constraints (9) that have been relaxed, we compute a gradient vector G as

$$G_j = d_j - \left(\sum_{i=0}^{n} \sum_{q=d_j}^{Q-d_i} q Z_{ijq} - \sum_{i=1}^{n} \sum_{q=d_i}^{Q-d_j} q Z_{jiq} \right), \quad j = 1, \ldots, n$$

The scalar step-size s is defined as:

$$s = \frac{\pi (Z_{\mathrm{UB}} - Z_{\mathrm{CLB}})}{\sum_{j=1}^{n} G_j^2}$$

where Z_{UB} is the upper bound (that corresponds to the best primal feasible solution found), Z_{CLB} is the current lower bound (associated with the solution of the previous relaxation problem), and π is a user-defined step-size parameter initialized at a certain value (e.g. 2) and reduced by half when a certain number of successive iterations of the subgradient search do not improve the lower-bound. Hence, if at iteration k, the vector of weights w^k is used, we determine the vector for the next iteration $(k+1)$ as

$$w^{(k+1)} = w^k + sG.$$

Notice that in the present setting the memory structures guiding the method correspond to the surrogate weighting vector comprising the gap between primal and dual solution values, the number of iterations since the last improvement of the best lower bound, and the position of the previous surrogate dual solution to the primal feasible space (expressed by the degree of violation of each of the relaxed constraints).

7 Adaptive solution combination method

The improvement component of the adaptive projection method constitutes one thread for reaching enhanced primal solutions through linkages between primal and dual information. Advanced RAMP designs include a number of additional possibilities to generate multiple search threads between primal and dual spaces as well as within each individual space, both encompassing the integration of primal-dual adaptive memory structures.

Although the surrogate relaxation approach in the dual side has the ability to integrate information extracted from individual constraints, each item of information generated in the dual component is represented in a single solution that results from solving the associated

surrogate problem (and the application of the complementary projection method). These solutions may be viewed as *single memory structures* that can be integrated to create more complex *compound memory structures*. In our implementation, this is accomplished by extending the primal component with scatter search through generating weighted combinations of these solutions (and so their attributes) to create new composite solutions. Since the RAMP method creates composite memory structures from both the primal and dual spaces we employ very simple implementations of the tabu search and scatter search methods. This favors the alternation between the primal and dual searches in place of performing extensive searches by the improvement and solution combination methods. In order to distinguish versions of the method using single memory structures from those using composite memory structures, we call the latter a Primal-Dual RAMP (PD-RAMP).

The PD-RAMP algorithm specifically extends the basic RAMP algorithm by maintaining a reference set of several solutions instead of only the current best solution. To ensure proper diversification, the reference set RS is organized to contain two distinct subsets B and D, representing respectively the subsets of high-quality and diverse solutions, hence $RS = B \cup D$.

The consideration of multiple primal solutions entails two extra steps involved in combining the reference solutions and improving the corresponding offspring. Let $C(X_t)$ be the cost of a solution X_t in a selected subset $E \subseteq RS$, $r = |E|$, and let $H(E)$ denote the convex-hull of E. We generate offspring solutions $X \in H(E)$ represented as $X = \sum_{t=1}^{r} \lambda_t X_t$ with $\sum_{t=1}^{r} \lambda_t = 1$ and $\lambda_t \geq 0$ $(t = 1, \ldots, r)$, where the multiplier λ_t represents the weight assigned to the solution $X_t \in E$ and is computed by

$$\lambda_t = \frac{1/C(X_t)}{\sum_{t=1}^{r}(1/C(X_t))}$$

By this formula the better (lower cost) solutions receive higher weight than less attractive (higher cost) solutions. The score of each variable x_{ij} relative to the solutions in E is obtained by computing $\varphi(x_{ij}) = \sum_{t=1}^{r}(\lambda_t x_{ij}^t)$, where $x_{ij}^t = 1$ if x_{ij} is an arc in the solution X_t and $x_{ij}^t = 0$ otherwise. Finally, as variables are required to be binary, the value is obtained by rounding its score to give $x_{ij} = \lfloor \varphi(x_{ij}) + 1/2 \rfloor$.

We consider groups of subsets E of cardinality $r = 2, 3$, and 4, so that the first group contains all combinations of two different solutions in the reference set, and the other two groups are created recursively by augmenting each subset E of r solutions with the best solution $X \in RS \backslash E$ to form a subset E of size $r + 1$. Solutions in each subset are then combined using the foregoing solution combination method. As with the relaxation method, the offspring resulting from solutions combination are not expected to be feasible for the CMST problem, hence these solutions are then subjected to projection and improvement method for possible enhancement.

8 The RAMP algorithm

Our RAMP algorithm arises by integrating the aforementioned procedures with a relax-and-cut dual solution approach. The general structure of the RAMP and PD-RAMP algorithms is presented in Sect 8.1. Section 8.2 provides the details of the procedures implementing each of the component methods. Section 8.3 discusses problem reduction techniques used in the implementation.

8.1 The algorithms

Algorithm RAMP
Repeat the following steps until the maximum number of failures to update the reference set (i.e. the best upper bound) is reached:

1. Perform Adaptive Weighting Update Method. If the optimal solution was found exit.
2. Perform Adaptive Memory Relaxation Method.
3. Perform Adaptive Memory Projection Method.

End Repeat

Algorithm PD-RAMP
Repeat the following steps until both Dual and Primal fail to update the reference set:

Dual:
  Repeat the following steps until the maximum number of failures to update the reference set (i.e. the best upper bound) is reached:
    D1. Perform Adaptive Weighting Update Method.
        If the optimal solution was found stop the algorithm.
    D2. Perform Adaptive Memory Relaxation Method.
    D3. Perform Adaptive Memory Projection Method.
  End Repeat
End Dual

Primal:
  Repeat the following steps until the maximum number of failures to update the reference set is reached:
    P1. Perform Adaptive Solution Combination Method
    P2. Perform Adaptive Memory Projection Method
  End Repeat
End Primal

End Repeat

8.2 The algorithm procedures

8.2.1 Procedure adaptive weighting update method

If no surrogate problem has been solved yet, set all surrogate weights equal to 1 and end the procedure. Otherwise, using the surrogate solution of the last surrogate problem solved, check for violations of the flow-conservation constraints (9) and implicitly generate the new set of special bin-packing cuts associated with possible violations of inequalities (13). If no violation is found and the integrality constraints (10) are also satisfied, end the procedure—the solution is optimal for the CMST. Otherwise compute new surrogate weights using subgradient optimization.

8.2.2 Procedure adaptive memory relaxation method

Given the vector of nonnegative weights associated with the primal constraints to be relaxed, form a linear combination of these constraints to create a surrogate constraint which, along

with the selected set of cutting planes and the relaxation of the integrality constraints, creates the corresponding surrogate problem. Solve the surrogate problem to optimality, thus obtaining a lower bound on the objective function value of the original CMST problem. Update the best lower bound if the solution of the surrogate problem improves the current best lower bound. Make the surrogate solution as the only solution in the working solution list.

8.2.3 Procedure adaptive memory projection method

Repeat the following procedure for each solution in the working solution list: Subject the solution to projection and possible improvement, creating an enhanced primal feasible solution that is locally optimal. If the objective function value for the original problem produced by this solution is the best found so far by the algorithm, it is maintained as the best upper bound. Similarly, this new solution is added to the reference set if the set does not contain the designated number of elements or if the evaluation of the new solution is higher than that of the lowest evaluation solution in the set based upon the high-quality or diversity criterion. Update the number of failures to update the reference set.

8.2.4 Procedure adaptive solution combination method

Subject the solutions in the reference set to subset generation and solution combination and create a working solution list of these offspring solutions.

8.3 Problem reduction

Problem reduction techniques have been proposed for the CMST to eliminate variables (arcs) that do not belong to an optimal solution (see, e.g., Malik and Yu 1993). The arc elimination techniques used in the dual, the neighborhood search of the improvement method and the solution combination component of the scatter search make use of the following logical implications: If $c_{0j*} = \min\{c_{0j} : j \in V\}$, arc $(0, j^*)$ is essential and hence arcs from all the other nodes to j^* can be eliminated; for a non-root node j if $c_{0j} = \min\{c_{ij} : i \in V_0\}$, the arc $(0, j)$ can be included in the optimal solution, again allows us to eliminate arcs from all the other nodes to j; finally, for any two non-root nodes i and v_j, if $c_{ij} \geq \max\{c_{0i}, c_{0j}\}$, then arc (i, j) can be eliminated.

9 Computational experiments

The classical set of benchmark CMST problems from the OR-Library archive was used to gauge the performance of RAMP and PD-RAMP algorithms. Both algorithms were coded in C++ and tests were carried out on a desktop computer with an Intel Pentium P4 processor, 2 GB of RAM and running Windows XP professional. The CPLEX 8.1 LP solver was invoked as a subroutine, using the API Callable Library, to provide optimal solutions for the surrogate problems.

Extensive computational analysis was performed on problems of different type, size, and arc capacity. The TC and TE test sets contain homogeneous demand instances of 40 and 80 nodes while the CM test set contains heterogeneous demand instances of sizes 50, 100, and 200 nodes. All the problems are defined on complete graphs with costs represented by Euclidean distances between nodes. Known optimum solutions and best known solutions were obtained from the approaches due to Uchoa et al. (2008), Ahuja et al. (2001, 2003), and from Amberg et al. (1996) as reported in Patterson et al. (1999).

Table 1 Summary of results for 40-node instances

Problem	n	Q	BKS	PD-RAMP			RAMP		
				BSF	RPD	CPU	BSF	RPD	CPU
tc40-2	40	3	717	717	0.00	23	719	0.28	9
te40-1	40	3	1190	1190	0.00	96	1191	0.08	50
te40-4	40	3	1132	1134	0.18	24	1134	0.18	30
te40-5	40	3	1104	1104	0.00	35	1106	0.18	31
te40-8	40	3	1181	1181	0.00	398	1183	0.17	7
te40-9	40	3	1090	1090	0.00	114	1092	0.18	10
te40-10	40	3	1079	1079	0.00	33	1082	0.28	17
te40-1	40	5	830	830	0.00	240	835	0.60	39
te40-1	40	5	797	797	0.00	50	801	0.50	43
te40-8	40	5	827	827	0.00	53	832	0.60	55
te40-9	40	5	779	780	0.13	13	780	0.13	12
te40-1	40	10	596	596	0.00	87	598	0.34	31
te40-5	40	10	572	572	0.00	93	574	0.35	101
te40-7	40	10	591	591	0.00	102	593	0.34	75
te40-8	40	10	610	610	0.00	70	612	0.33	23
Average					0.02	95.4		0.30	35.5
Overall Average (60 Instances)					0.01	33.6		0.08	17.9

All the results were obtained in specified time limits found empirically to be the best compromise between computation time and solution quality for different problem types and sizes. A similar stopping criterion has also been used in the current state-of-the-art algorithm by Ahuja et al. (2003), denoted here by AOS. For the sake of comparative analysis we have run AOS on the same computer as our RAMP and PD-RAMP algorithms. AOS found all optimal solutions within 1000 seconds for 40-node instances and within 2000 seconds for 50- and 80-node instances. In general, RAMP and PD-RAMP algorithms require longer running times to find the best solutions or stabilize convergence; hence we extended the cut-off times for these algorithms to 2000 seconds for 40-node instances, and 3000 seconds for 50- and 80-node instances. Because the 100-node instances are significantly more difficult and the algorithms have different convergence rates for each instance, we report results at selected time intervals that were deemed relevant for the analysis.

For the 40-node instances both RAMP and PD-RAMP algorithms ran very fast. The (dual) RAMP algorithm found the optimal solution in 45 of the 60 instances with an average running time of 18 seconds. The PD-RAMP algorithm proved more effective, finding the optimal solution for all but 2 of the 60 instances with an average running time of 34 seconds. The relative average percent deviation is very small, being 0.00 for AOS, 0.01 for PD-RAMP, and 0.08 for RAMP. A summary of these results is presented on Table 1. Specifically, for each instance where an algorithm failed to find the optimal solution, we report the best known solution (BKS) from the literature, the best solution found (BSF) by the algorithms, the associated relative percent deviation (RPD) above BKS, and the computation time (CPU) in seconds for the algorithm to find its minimum cost solution. We also provide averages of solution quality and running times over the whole set of problems tested. Similar information is provided in the remaining tables used in the analysis.

Table 2 Results for 80-node homogeneous demand problems (30 instances)

Problem	n	Q	BKS	PD-RAMP			RAMP		
				BSF	RPD	CPU	BSF	RPD	CPU
tc80-1	80	5	1099	1100	0.09	139	1114	1.36	99
tc80-2	80	5	1100	1102	0.18	1272	1106	0.55	166
tc80-3	80	5	1073	1073	0.00	155	1079	0.56	178
tc80-4	80	5	1080	1080	0.00	1621	1091	1.02	215
tc80-5	80	5	1287	1287	0.00	426	1288	0.08	194
tc80-1	80	10	888	888	0.00	146	890	0.23	102
tc80-2	80	10	877	877	0.00	21	877	0.00	20
tc80-3	80	10	878	878	0.00	46	878	0.00	45
tc80-4	80	10	868	868	0.00	12	868	0.00	7
tc80-5	80	10	1002	1002	0.00	111	1002	0.00	117
tc80-1	80	20	834	834	0.00	48	834	0.00	48
tc80-2	80	20	820	820	0.00	2	820	0.00	3
tc80-3	80	20	828	828	0.00	2	828	0.00	3
tc80-4	80	20	820	820	0.00	4	820	0.00	3
tc80-5	80	20	916	916	0.00	5	916	0.00	3
Average TC					0.02	267.3		0.25	80.2
te80-1	80	5	2544	2548	0.16	908	2554	0.39	641
te80-2	80	5	2551	2557	0.24	2965	2568	0.67	653
te80-3	80	5	2612	2620	0.31	2340	2631	0.73	857
te80-4	80	5	2558	2558	0.00	2281	2581	0.90	785
te80-5	80	5	2469	2469	0.00	2301	2483	0.57	660
te80-1	80	10	1657	1671	0.84	1298	2674	1.03	1263
te80-2	80	10	1639	1646	0.43	1642	1646	0.43	1240
te80-3	80	10	1687	1689	0.12	1370	1712	1.48	1406
te80-4	80	10	1629	1629	0.00	1276	1656	1.66	1530
te80-5	80	10	1603	1614	0.69	1199	1625	1.37	1294
te80-1	80	20	1275	1277	0.16	1916	1279	0.31	2198
te80-2	80	20	1224	1228	0.33	2604	1228	0.33	2311
te80-3	80	20	1267	1267	0.00	2035	1267	0.00	2300
te80-4	80	20	1265	1265	0.00	1766	1265	0.00	2140
te80-5	80	20	1240	1240	0.00	1739	1240	0.00	2273
Average TE					0.22	1842.7		0.66	1436.7
Overall Average					0.11	1095.2		0.46	755.3

Table 2 reports detailed results for larger 80 node TC and TE instances. The table shows that the algorithms' performance is relatively better on TC instances than on TE instances. We conjecture that the location of the root node at one extreme of the node distribution space in TE instances, as opposed to being located around the center as in TC instances, may affect the effectiveness of a candidate list that relies on nearest neighbor strategies as in our improvement method. It happens that the candidate lists tend to be larger for TE instances, to account for possible deficiencies of the nearest neighbor strategy in determining

Table 3 Results for 50-node heterogeneous demand problems (15 instances)

Problem	n	Q	BKS	PD-RAMP			RAMP		
				BSF	RPD	CPU	BSF	RPD	CPU
cm50r1	49	200	1098	1104	0.55	1850	1115	1.55	924
cm50r2	49	200	974	980	0.62	874	984	1.03	1438
cm50r3	49	200	1186	1186	0.00	1668	1201	1.26	197
cm50r4	49	200	800	800	0.00	314	804	0.50	259
cm50r5	49	200	928	936	0.86	1400	940	1.29	788
cm50r1	49	400	679	679	0.00	2974	681	0.29	762
cm50r2	49	400	631	631	0.00	746	635	0.63	383
cm50r3	49	400	732	732	0.00	1079	739	0.96	697
cm50r4	49	400	564	564	0.00	387	564	0.00	51
cm50r5	49	400	611	612	0.16	254	612	0.16	245
cm50r1	49	800	495	495	0.00	253	495	0.00	247
cm50r2	49	800	513	513	0.00	719	513	0.00	710
cm50r3	49	800	532	532	0.00	98	532	0.00	93
cm50r4	49	800	471	471	0.00	13	471	0.00	17
cm50r5	49	800	492	492	0.00	129	492	0.00	327
Average					0.15	850.5		0.51	475.9

the most appropriate candidates. We conjecture that this may likewise affect the efficiency of the algorithm in those instances. Although the simple RAMP algorithm finds solutions of very high quality, PD-RAMP is faster in finding solutions of similar quality and is globally more effective.

Tables 3 and 4 report results for CM instances containing 50 and 100 nodes, respectively. Solutions for the largest 200-node CM instances of the test bank could not be obtained due to insufficient memory. As noted earlier, our base CMST formulation has the advantage of containing only $2n$ constraints, which likely contributes to the effectiveness of our surrogate dual model. On the other hand, the fact that the number of variables has a quadratic term in n may limit the size of the problems that can be addressed. Similarly, the constant term Q in the number of variables may also have an important effect on the space complexity in problems with very large arc capacity.

For comparative purposes, Table 4 also includes results for the AOS algorithm. In these experiments we allow the algorithms to run for a maximum of 86,000 seconds and report solutions whenever an algorithm found an improved solution in the past 1000 seconds relative to the corresponding times. Dashes in the table mean that the solution has not changed in the corresponding intervals.

The simple RAMP algorithm, although still effective and competitive with AOS with respect to the average of solution quality, cannot keep up with the PD-RAMP algorithm for the larger 100-node instances. AOS does slightly better than RAMP and PD-RAMP on the 50-node instances, but PD-RAMP performs better than these algorithms on the 100-node instances.

Specifically, for the 15 small 50-node instances AOS did better than PD-RAMP on 4 instances and did equally well on the remaining 11 instances. For the more challenging 100-node instances, however, PD-RAMP outperforms AOS on 3 out of the 5 instances, and equaled AOS on the only instance where this algorithm was capable of finding an optimal

Table 4 Results for 100-node heterogeneous demand problems (5 instances)

Problem	n	Q	BKS	CPU	AOS		PD-RAMP		RAMP	
					BSF	RPD	BSF	RPD	BSF	RPD
cm100r1	99	200	509	1000	516	1.38	509	0.00	527	3.54
				8000	–	–	–	–	520	2.16
				18,000	–	–	–	–	519	1.96
				26,000	–	–	–	–	518	1.77
				29,000	–	–	–	–	509	0.00
cm100r2	99	200	584	1000	596	2.05	621	6.34	606	3.77
				2000	593	1.54	615	5.31	–	–
				4000	–	–	612	4.79	604	3.42
				6000	–	–	609	4.28	–	–
				7000	–	–	607	3.94	–	–
				8000	–	–	602	3.08	–	–
				9000	–	–	599	2.57	602	3.08
				13,000	–	–	594	1.71	600	2.74
				23,000	–	–	–	–	598	2.40
				32,000	–	–	–	–	597	2.23
				40,000	–	–	–	–	595	1.88
				72,000	–	–	591	1.20	–	–
cm100r3	99	200	540	1000	541	0.19	574	6.30	569	5.37
				2000	–	–	564	4.44	568	5.19
				3000	–	–	–	–	562	4.07
				7000	–	–	–	–	560	3.70
				8000	–	–	562	4.07	555	2.78
				9000	–	–	551	2.04	–	–
				12,000	–	–	548	1.48	–	–
				20,000	–	–	–	–	547	1.30
				26,000	–	–	546	1.11	–	–
				27,000	–	–	542	0.37	–	–
				31,000	–	–	–	–	546	1.11
cm100r4	99	200	435	1000	437	0.46	455	4.60	443	1.84
				2000	435	0.00	437	0.46	–	–
				3000	–	–	–	–	442	1.61
				7000	–	–	435	0.00	–	–
				11,000	–	–	–	–	439	0.92
				28,000	–	–	–	–	435	0.00
cm100r5	99	200	418	1000	425	1.67	436	4.31	432	3.35
				3000	421	0.72	436	4.31	–	–
				4000	–	–	431	3.11	–	–
				5000	–	–	–	–	427	2.15
				6000	–	–	428	2.39	–	–
				9000	–	–	–	–	426	1.91
				10,000	–	–	423	1.20	426	1.91
				19,000	–	–	–	–	421	0.72

Table 4 (*Continued*)

Problem	*n*	*Q*	BKS	CPU	AOS		PD-RAMP		RAMP	
					BSF	RPD	BSF	RPD	BSF	RPD
				31,000	–	–	422	0.96	–	–
				32,000	–	–	421	0.72	–	–
				33,000	–	–	420	0.48	–	–
				40,000	–	–	419	0.24	–	–
				53,000	420	0.48	419	0.24	–	–
				78,000	–	–	418	0.00	–	–
Average RPD for BSF						0.72		0.31		0.74
Total Time to BSF						59000		185000		147000

solution. PD-RAMP found 3 optimal solutions. This resulted in an average percent deviation of 0.72 for AOS and 0.31 for PD-RAMP. Also PD-RAMP is usually faster than AOS in finding similar or improved solutions for several of these instances. Notably, PD-RAMP finds the optimal solution for cm100r1 in less than 1000 seconds while in this time AOS could only find a solution that is 1.38% away from optimality and was unable to improve it in another 28,000 seconds, which was sufficient for the simple RAMP algorithm to also find the optimal solution. Likewise, for cm100r5 AOS took about 20,000 seconds more than PD-RAMP to find a solution of similar quality, and then could not improve it within another 33,000 seconds. In addition, PD-RAMP reached the optimal solution in 8,000 seconds less than the time limit given to AOS. On the other hand, AOS performed exceedingly well on the cm100r3 instance by reaching a solution that is 0.19% above the optimum in less than 1,000 seconds, while it required about 27,000 seconds to PD-RAMP to find a solution 0.37% above the optimum. On average, for these five larger instances PD-RAMP required about 3 times longer than AOS to find its best solutions, but this extra time seems well justified by the superior solution quality produced by PD-RAMP.

Overall we observe that while PD-RAMP often (though not always) consumed more time than AOS to reach the point where it found a superior solution, the time necessary for PD-RAMP to find its best solutions in each group of problems is on average significantly less than the time limit imposed on the two methods. Though PD-RAMP is slightly better on average, the solution quality achieved by this method is fairly comparable to that of AOS. While AOS converges very quickly to a good solution, often optimal for the smaller problems, AOS has trouble finding further improvements for the larger instances even under extensive running times. PD-RAMP, on the other hand, is very robust, providing sustained improvement and demonstrating a high likelihood of finding progressively better solutions for larger and more complex problems as the allotted solution time is increased.

10 Summary and conclusions

The RAMP approaches proposed in this study for the CMST are based on two components. The first consists of a relaxation approach using surrogate constraints and associated cutting planes. The second component blends heuristic projection with improvement methods that take advantage of adaptive memory. The preliminary form of our RAMP procedure, the dual RAMP algorithm, couples surrogate dual information with a simple tabu approach

based on classical neighborhood structures. This method proved superior to all previous metaheuristic approaches except the very large-scale neighborhood (VLSN) approach of Ahuja et al. (2003). Our more advanced primal-dual RAMP algorithm, which enhances the primal approach with a scatter search procedure, compares very well against VLSN on small to medium size problems and is more effective in solving larger and more complex instances. These results encourage the development of additional problem reduction techniques for the surrogate dual to allow for the solution of large-scale CMST instances.

Acknowledgements We thank Ravindra Ahuja, James Orlin, and Dushyant Sharma for making their AOS code available to us. We are also indebted to two anonymous referees whose comments significantly helped improving the presentation of our paper. Part of this research was developed when Cesar Rego was visiting MIT Sloan School of Management.

References

Ahuja, R. K., Orlin, J. B., & Sharma, D. (2001). Multi-exchange neighborhood search structures for the capacitated minimum spanning tree problem. *Mathematical Programming*, *91*, 71–97.

Ahuja, R. K., Orlin, J. B., & Sharma, D. (2003). A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem. *Operations Research Letters*, *31*, 185–194.

Amberg, A., Domschke, W., & Voß, S. (1996). Capacitated minimum spanning trees: algorithms using intelligent search. *Combinatorial Optimization: Theory and Practice*, *1*, 9–39.

Caserta, M. (2007). Tabu search-based metaheuristic algorithm for large-scale set covering problems. In K. F. Doerner, M. Gendreau, P. Greistorfer, W. J. Gutjar, R. F. Hartl, & M. Reimann (Eds.), *Metaheuristics: progress in complex systems optimization* (pp. 43–63). Berlin: Springer.

Elias, D., & Ferguson, M. J. (1974). Topological design of multipoint teleprocessing networks. *IEEE Transactions on Communications*, *22*, 1753–1762.

Esau, L. R., & Williams, K. C. (1966). On teleprocessing system design. Part II. A method for approximating the optimal network. *IBM Systems Journal*, *5*, 142–147.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. New York: Freeman.

Gavish, B. (1982). Topological design of centralized computer networks: formulations and algorithms. *Networks*, *12*, 355–377.

Gavish, B. (1983). Formulations and algorithms for the capacitated minimal directed spanning tree problem. *Journal of ACM*, *30*(1), 118–132.

Gavish, B. (1985). Augmented Lagrangian based algorithms for centralized network design. *IEEE Transactions on Communications, COM-33*, *12*, 1247–1257.

Gavish, B. (1991). Topological design telecommunications networks: local access design methods. *Annals of Operations Research*, *33*, 17–71.

Glover, F. (1989a). Tabu search. Part I. *ORSA Journal on Computing*, *1*, 190–206.

Glover, F. (1989b). Tabu search. Part II. *ORSA Journal on Computing*, *2*, 4–32.

Glover, F. (1996). Tabu search and adaptive memory programming: advances, applications and challenges. In R. Barr, R. Helgason, & J. Kennington (Eds.), *Interfaces in computer science and operations research* (pp. 1–75). Norwell: Kluwer Academic.

Glover, F. (2005). Adaptive memory projection methods for integer programming. In C. Rego, & B. Alidaee (Eds.), *Metaheuristic optimization via memory and evolution* (pp. 425–440). Norwell: Kluwer Academic.

Glover, F., & McMillan, C. (1986). The general employee scheduling problem: an integration of management science and artificial intelligence. *Computers and Operations Research*, *13*(4), 563–593.

Gouveia, L. (1995). A 2*n* constraint formulation for the capacitated minimal spanning tree problem. *Operations Research*, *43*, 130–141.

Grünert, T. (2002). Lagrangian tabu search. In C. C. Ribeiro & P. Hansen (Eds.), *Essays and surveys in metaheuristics* (pp. 379–397). Norwell: Kluwer Academic.

Hall, L. A. (1996). Experience with a cutting plane approach for the capacitated spanning tree problem. *ORSA Journal on Computing*, *8*, 219–234.

Karwan, M. H., & Rardin, R. L. (1984). Surrogate dual multiplier search procedures in integer programming. *Operations Research*, *32*(1), 52–69.

Klincewicz, J. G., Luss, H., & Yan, D. C. K. (1998). Designing tributary networks with multiple ring families. *Computers*. *Operations Research*, *25*(12), 1145–1157.

Lorena, L. A. N., & Narciso, M. G. (1999). Lagrangian/surrogate relaxation for generalized assignment problems. *European Journal of Operational Research*, *123*, 325–332.

Malik, K., & Yu, G. (1993). A branch and bound algorithm for the capacitated minimum spanning tree problem. *Networks*, *23*, 525–532.

Mathew, F., & Rego, C., (2006). Recent advances in heuristics for the capacitated minimum spanning tree problem. In *Proceedings of the Decision Sciences Institute (DSI)*, San Antonio, TX (pp. 31021–31026).

Orús, R. (2005). Two slightly-entangled NP-complete problems. *Quantum Information and Computation*, *5*(6), 449–455.

Papadimitriou, C. (1978). The complexity of the capacitated tree problem. *Networks*, *8*, 217–230.

Patterson, R., Pirkul, H., & Rolland, E. (1999). Memory adaptive reasoning for solving the capacitated minimum spanning tree problem. *Journal of Heuristics*, *5*, 159–180.

Polyak, B. T. (1969). Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, *9*, 14–29.

Rego, C. (2005). RAMP: A new metaheuristic framework for combinatorial optimization. In C. Rego & B. Alidaee (Eds.), *Metaheuristic optimization via memory and evolution: tabu search and scatter search* (pp. 441–460). Norwell: Kluwer Academic.

Sharaiha, Y. M., Gendreau, M., Laporte, G., & Osman, I. H. (1997). A tabu search algorithm for the capacitated shortest spanning tree problem. *Networks*, *29*, 161–171.

Uchoa, E., Fukasawa, R., Lysgaard, J., Pessoa, A., de Aragão, M. P., & Andrade, D. (2008). Robust branch-but-and-price for the capacitated minimum spanning tree problem over a large extended formulation. *Mathematical Programming, Series A*, *112*(2), 443–472.

Yagiura, M., Kishida, M., & Ibaraki, T. (2006). A 3-flip neighborhood local search for the set covering problem. *European Journal of Operational Research*, *172*, 472–499.

Zhang, N. (1993). Facet-defining inequalities for minimum spanning trees. Master thesis, Princeton University, Princeton, New Jersey