Solving Connected Subgraph Problems in Wildlife Conservation

Bistra Dilkina and Carla P. Gomes

Department of Computer Science, Cornell University, Ithaca, NY 14853, U.S.A.
{bistra,gomes}@cs.cornell.edu

Abstract. We investigate mathematical formulations and solution techniques for a variant of the Connected Subgraph Problem. Given a connected graph with costs and profits associated with the nodes, the goal is to find a connected subgraph that contains a subset of distinguished vertices. In this work we focus on the budget-constrained version, where we maximize the total profit of the nodes in the subgraph subject to a budget constraint on the total cost. We propose several mixed-integer formulations for enforcing the subgraph connectivity requirement, which plays a key role in the combinatorial structure of the problem. We show that a new formulation based on subtour elimination constraints is more effective at capturing the combinatorial structure of the problem, providing significant advantages over the previously considered encoding which was based on a single commodity flow. We test our formulations on synthetic instances as well as on real-world instances of an important problem in environmental conservation concerning the design of wildlife corridors. Our encoding results in a much tighter LP relaxation, and more importantly, it results in finding better integer feasible solutions as well as much better upper bounds on the objective (often proving optimality or within less than 1% of optimality), both when considering the synthetic instances as well as the real-world wildlife corridor instances.

1 Introduction

A large class of decision and optimization problems can be captured as finding a *connected subgraph* of a larger graph satisfying certain cost and revenue requirements. In different realizations of the Connection Subgraph Problem costs and profits are associated with either edges, nodes or both. Examples of this family of problems are the Minimum Steiner Tree, Maximum-Weighted Connected Subgraph and Point-to-Point Connection Problem. Such problems arise in a large number of applications – e.g. network design, system biology, social networks and facility location planning.

Here, we are concerned with a variant of the Connected Subgraph Problem where we are given a graph with costs and profits associated with nodes and one or more designated nodes called terminals and we seek to find a connected subgraph that includes the terminals with maximal profit and total cost within a specified budget which we refer to as the *Budget-Constrained Steiner Connected*

Subgraph Problem with Node Profits and Node Costs. This problem is known to be NP-hard even for the case of no terminals [2]. Removing the connectivity constraint, we have a 0-1 knapsack problem. On the other hand, the connectivity constraint relates it to other important classes of well-studied problems such as the Traveling Salesman Problem and the Steiner Tree problem. The connectivity constraint plays a key role in the combinatorics of this problem and we propose new mathematical formulations to better capture the structure of the problem w.r.t. the connectivity constraint.

Our work is motivated by an important instance of this problem that arises in Conservation Planning. The general problem consists of selecting a set of land parcels for conservation to ensure species viability. This problem is also known in the literature in its different variants as site selection, reserve network design, and corridor design. Biologists have highlighted the importance of addressing the negative ecological impacts of habitat fragmentation when selecting parcels for conservation. To this effect, ways to increase the spatial coherence among the set of parcels selected for conservation have been investigated (see [14] for a review). We look at the problem of designing so-called wildlife corridors to connect areas of biological significance (e.g. established reserves). Wildlife corridors are an important conservation method in that they increase the genetic diversity and allow for greater mobility (and hence better response to predation and stochastic events such as fire, as well as long term climate change). Specifically, in the wildlife corridor design problem, we are given a set of land parcels, a set of reserves (land parcels that correspond to biologically significant areas), and the cost (e.g. land value) and utility (e.g. habitat suitability) of each parcel. The goal is to select a subset of the parcels that forms a connected network including all reserves. This problem is clearly an instance of the Connected Subgraph Problem with node profits and node costs, where the nodes correspond to parcels, the terminal nodes correspond to the reserves and the edges correspond to adjacency of parcels. Conservation and land use planners generally operate with a limited budget while striving to secure the land that results in the corridor with best habitat suitability. This results in the budget-constrained version of the connected subgraph problem.

The connected subgraph problem in the context of designing wildlife corridors was recently studied in [2, 7]. Conrad et al. [2] designate one of the terminals as a root node and encode the connectivity constraints as a single commodity flow from the root to the selected nodes in the subgraph. This encoding is small and easy to enforce. They present computational results which show an easy-hard-easy runtime pattern with respect to the allowed budget on a benchmark of synthetic instances [7]. Further, when solving large scale real world instances of this optimization problem, the authors report extremely large running time. Here, we try to improve the state-of-the-art for this problem by proposing alternative formulations. We show that the easy-hard-easy pattern in runtime solution for finding optimal solutions observed for synthetic instances aligns with a similar pattern in the relative integrality gap of the LP relaxation of the model. This observation suggests that formulations that have tighter LP relaxations might also lead to faster solution times for finding optimal solutions. To this effect, we propose two additional formulations.

One possible alternative which we explore in this paper is to establish the connectivity of each selected node to the root node by a separate commodity flow. This results in a multi-commodity flow encoding of the connectivity constraints. Although the multi-commodity flow encoding is larger than the single commodity encoding (yet still polynomial size), it can result in a stronger LP relaxation of the problem.

A completely different avenue is to adapt ideas from the vast literature on the Steiner Tree Problem. Encodings of the connectivity requirement with respect to edge decisions successfully applied to the Steiner Tree problem involve exponential number of constraints. The Steiner Tree variants involve costs and/or profits on edges and hence such models explicitly model binary decisions of including or excluding edges from the selected subgraph. In particular, for the Steiner Tree Problem with Node Revenues and Budgets, Costa et al. [4] suggest using the directed Dantzig-Fulkerson-Johnson formulation [5] with subtour elimination constraints enforcing the tree structure of the selected subgraph. For variants of the Connection Subgraph Problem that involve edge costs or edge profits one needs to model explicitly decisions about inclusion of edges in the selected subgraph. Given a graph $G = (V, E)$, in the problem variant we study we only need to make explicit decisions of which nodes to include (i.e., $V' \subseteq V$) and connectivity needs to be satisfied on the induced subgraph $G(V')$ that only contains edges of G whose endpoints belong to V'. Nevertheless, we adapt the directed Dantzig-Fulkerson-Johnson formulation to our problem, therefore considering the graph edges as decision variables instead of the nodes, which in general results in dramatically increasing the search space size from $2^{|V|}$ to $2^{|E|}$. Although at first glance this change seems counterproductive, the added strength that results from explicitly enforcing the connectivity of each selected node to a predefined terminal, in fact, results in a tighter formulation. This formulation involves an exponential number of connectivity constraints that cannot be represented explicitly for real life sized instances. To address this, we present a Bender's decomposition approach that iteratively adds connectivity constraints to a relaxed master problem [1, 12].

We provide computational results on the three different encodings of the connectivity constraints: 1) the single-commodity flow (SCF) encoding [2]; 2) a multi-commodity flow (MCF) encoding; 3) a modified directed Dantzig-Fulkerson-Johnson (DFJ) formulation using node costs. On a benchmark of synthetic instances consisting of grid graphs with random costs and revenues, we show that indeed the multi-commodity encoding provides better LP relaxation bounds than the single commodity flow, and that the directed Dantzig-Fulkerson-Johnson formulation provides the best bounds. Most importantly, the advantage of the bounds provided by the directed Dantzig-Fulkerson-Johnson formulation over the single-commodity flow encoding are greatest exactly in the hard region. The tighter bounds turn out to have a critical effect on the solution times for finding optimal integer feasible solutions. Despite the large size of the DFJ encoding, it works remarkably well for finding integer feasible solutions. The easy-hard-easy pattern with respect to the budget exhibited strongly by the SCF encoding is much less pronounced when using the DFJ encoding – this encoding is considerably more robust to the budget level. We show that the DFJ encoding finds optimal solutions two orders of magnitude faster than the SCF encoding in the interval of budget values that are hardest. This result is particularly relevant when solving real-world instances because the hard region usually falls over a budget interval close to the minimum cost solution to find a connected subgraph – i.e. it helps find solutions for tight budgets.

We test our formulations on real problem instances concerning the design of a Grizzly Bear Wildlife Corridor connecting three existing reserves [2]. We show that, for critically constrained budgets, the DFJ encoding proposed here can find optimal or close to optimal solutions, dramatically speeding up runtime. For the same problem instances and budget levels, the single flow encoding can only find considerably worse feasible solutions and has much worse objective upper bounds. For example, for a budget level which is 10% above the minimum cost required to connect all reserves, the DFJ encoding finds an optimal soltuion and proves optimality in 25 mins, while the SCF encoding after 10 hours has found an inferior solution and has proven an optimality gap of 31%. Similar behavior is observed for a budget of 20% above the minimum cost. Working budgets close to the minimum cost solution is a very likely scenario in a resource-constrained setting such as conservation planning. Hence, with the little money available, it is important to find the best possible solutions. The new DFJ encoding proposed here allows us to find optimal solutions to large scale wildlife corridor problems in exactly the budget levels that are most relevant in practice and that are out of reach in terms of computational time for the previously proposed formulations.

The DFJ encoding is better at capturing the combinatorial structure of the connectivity constraints which is reflected in the tightness of the LP relaxation as well as in the fact that it finds integer feasible solutions much faster and with very strong guarantees in terms of optimality (often proving optimality or within less than 1% of optimality), both when considering the synthetic instances as well as the real-world wildlife corridor instances.

2 Related Work

One of the most studied variant of the Connected Subgraph Problem is perhaps the Steiner Tree which involves a graph $G = (V, E)$, a set of terminal vertices $T \subset V$, and costs associated with edges. In the Minimum Steiner Tree Problem the goal is to select a subgraph $G' = (V' \subseteq V, E' \subseteq E)$ of the smallest cost possible that is a tree and contains all terminals ($T \subseteq V'$). Although including a budget constraint has important practical motivation, budget-constrained variants of the Steiner tree problem are not as nearly widely studied as the minimum Steiner tree or the prize-collecting variant. The variant that is more relevant here is the Budget Prize Collecting Tree Problem where in addition to costs associated with edges, there are also revenues associated with nodes. The goal is to select a Steiner tree with total edge cost satisfying a budget constraint while maximizing the total node revenue of the selected tree. Levin [10] gives a $(4+\epsilon)$-approximation scheme for this problem. Costa et al. [3, 4] study mathematical formulations and solution techniques for this problem in the presence of additional so-called hop constraints. They use a directed rooted tree encoding with an exponential number of connectivity constraints and a Branch-and-Cut solution technique. One can easily see that the Budget Prize Collecting Tree Problem is a special case of the Budget-Constrained Steiner Connected Subgraph with Node Profits and Node Costs by replacing each edge with an artificial node with the corresponding cost and adding edges to the endpoints of the original edge. We adapt some of the vast amount of work on tight formulations for the variants of the Steiner Tree problem with edge costs to the more general node-weighted problem.

Restricted variants of Budget-Constrained Steiner Connected Subgraph Problem with Node Profits and Node Costs have been addressed previously in the literature. Lee and Dooly [9] study the Maximum-weight Connected Subgraph Problem where profits and unit costs are associated with nodes and the goal is to find a connected subgraph of maximal weight and at most a specified R number of nodes. In the constrained variant they consider a designated root node that needs to be included in the selected subgraph.

Moss and Rabani [13] also study the connected subgraph problem with node costs and node profits and refer to this problem as the *Constrained Node Weighted Steiner Tree Problem.* They also only consider the special case where there is either no terminals or only one terminal - a specified root node. For all three optimization variants - the budget, quota and prize-collecting, Moss and Rabani [13] provide an approximation guarantee of $O(\log n)$, where n is the number of nodes in the graph. However, for the budget variant, the result is a bi-criteria approximation, i.e. the cost of the selected nodes can exceed the budget by some fraction. Finding an approximation algorithm for the budget-constrained variant is still an open question, as well as dealing with multiple terminals. Demaine et al. [6] have recently shown that one can improve the $O(\log n)$ approximation guarantee to a constant factor guarantee when restricting the class of graphs to planar but only in the case of the minimum cost Steiner Tree Problem with costs on nodes (but no profits). It is an open research question whether for planar graphs one can design a better approximation scheme for the budget-constrained variant. This is of particular interest because the Wildlife Corridor Design problem corresponds to finding a connected subgraph in a planar graph.

3 Mathematical Formulations

The Connected Subgraph Problem with Node Profits and Node Costs is specified by a connected graph $G = (V, E)$ along with a set of terminal nodes $T \in V$, a cost function on nodes $c : V \rightarrow \mathcal{R}$, and a profit function on nodes $u : V \rightarrow \mathcal{R}$. The goal is to select a subset of the nodes $V' \subseteq V$ such that all terminal nodes T are included ($T \subseteq V'$) and the induced subgraph $G(V')$ is connected. In the Budget-Constrained variant, given a budget C we seek to find a connected subgraph such that the total cost of the nodes in V' do not exceed the budget C, while maximizing the total profit of the selected nodes.

In the following formulations, for each vertex $i \in V$, we introduce a binary variable x_i, representing whether or not i is included in the connected subgraph. Then, the objective function, the budget constraint and the terminal inclusion constraint are stated as:

$$\text{maximize} \sum_{i \in V} u_i x_i, \tag{1}$$

$$\text{s.t.} \sum_{i \in V} c_i x_i \leq C \tag{2}$$

$$x_t = 1, \qquad \forall t \in T \tag{3}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in V \tag{4}$$

In the following subsections we outline three different ways of enforcing the connectivity constraints — the selected vertices should induce a connected subgraph of the original graph G.

3.1 Connectivity as Single Commodity Flow

Conrad et al. [2], Gomes et al. [7] use a single-commodity network flow encoding where each undirected edge $\{i, j\} \in E$ is replaced by two directed edges (i, j) and (j, i). Let us call the set of directed edges A. They introduce a source vertex 0, with maximum total outgoing flow $n = |V|$. One arbitrary terminal vertex is chosen as root $r \in T$, and a directed edge $(0, r)$ is defined to insert the flow into the network. Each selected node acts as a "sink" by consuming one unit of flow, and a node can be selected only if it has positive incoming flow. Connectivity of the selected nodes is ensured by enforcing flow conservation constraints at all nodes.

More formally, for each (directed) edge $(i, j) \in A$, there is a non-negative variable y_{ij} to indicate the amount of flow from i to j and the following constraints are enforced:

$$x_0 + y_{0r} = n \tag{5}$$

$$y_{ij} \leq n x_j, \qquad \forall (i, j) \in A \tag{6}$$

$$\sum_{i:(i,j) \in A} y_{ij} = x_j + \sum_{i:(j,i) \in A} y_{ji}, \qquad \forall j \in V \tag{7}$$

$$\sum_{j \in V} x_j = y_{0r} \tag{8}$$

$$y_{ij} \geq 0, \qquad \forall (i, j) \in A \cup (0, r) \tag{9}$$

$$x_0 \geq 0, \qquad \forall (i, j) \in A \tag{10}$$

For the source of the flow, they introduce a variable $x_0 \in [0, n]$, representing the eventual residual flow. Constraint (5) states that the residual flow plus the flow injected into the network corresponds to the total system flow. Each of the vertices with a positive incoming flow retains one unit of flow, i.e., $(y_{ij} > 0) \Rightarrow (x_j = 1), \forall (i,j) \in A$ enforced by Constraint (6). The flow conservation is modeled in Constraint (7). Finally, Constraint (8) enforces that the flow absorbed by the network corresponds to the flow injected into the system. This encoding requires $2|E| + 1$ additional continuous variables and ensures that all selected nodes form a connected component.

3.2 Connectivity as Multi-commodity Flow

In the first encoding we enforce the connectivity of all selected nodes though a single commodity flow. In this model, the key difference is that we enforce the connectivity of the selected set of nodes by associating a separate commodity with each node. There will be one unit of flow from the root to each selected node of its own "commodity" type. We arbitrarily select one of the terminals as a *root* node denoted $r \in T$. Each other node i is a potential sink of one unit of commodity flow of type i that will have to be routed from the root node to i. Let use denote the set of neighbors of a node i as $\delta(i) = \{j | (i,j) \in A\}$.

Similarly to the original model, we still have binary decision variable for each vertex and the objective, the budget constraint and the terminal inclusion constraint are defined as before.

For each (directed) edge $(i,j) \in A$ and each node k different from the *root node* r, we introduce a variable $y_{kij} \geq 0$ which when it is nonzero indicates that the edge carries flow of type k. If a node k is selected then in becomes an active sink for flow of type k.

$$\sum_{j: r \in \delta(j)} y_{kjr} = 0 \qquad \forall k \in V - r \tag{11}$$

$$\sum_{j \in \delta(k)} y_{kjk} = x_k \qquad \forall k \in V - r \tag{12}$$

$$\sum_{j \in \delta(k)} y_{kkj} = 0 \qquad \forall k \in V - r \tag{13}$$

$$\sum_{j \in \delta(i)} y_{kij} = \sum_{i \in \delta(j)} y_{kji} \qquad \forall k, \forall i \in V - r, i \neq k \tag{14}$$

$$y_{kij} \leq x_i \qquad \forall k, \forall i \in V - r, \forall j \in \delta(i) \tag{15}$$

$$y_{kij} \leq x_j \qquad \forall k, \forall i \in V - r, \forall j \in \delta(i) \tag{16}$$

$$y_{kij} \geq 0 \qquad \forall k \in V - r, \forall (i,j) \in A \tag{17}$$

For all nodes k, if node k is selected, then k is a sink for flow of commodity k (Constraint (12, 13)). Constraint (14) imposes conservation of flow for each commodity type k at each node different from the sink k and the source r, and

Constraint (11) imposes that the root does not have any incoming flow. Finally, the capacity of each edge is zero if either end node is not selected, and 1 otherwise (Constraint (15, 16)).

This encoding requires $(|V|-1)2|E|$ additional continuous variables – considerably more than the SCF encoding. However, we will see that enforcing the connectivity of each node to the root separately results in tighter LP relaxation.

3.3 Connectivity as Directed Steiner Tree

As suggested by the multi-commodity flow encoding, to enforce connectivity one may enforce that there exists a path from each selected node to the root node. In this third encoding, we in fact explicitly model the selection of edges as binary variables and insist that we select a set of nodes and edges such that there is a single path from each selected node to the root (using the selected nodes). In other words, we impose stronger constraints than necessary while preserving all feasible solutions in terms of subset of nodes that induce a connected subgraph. In effect, we enforce the connectivity constraints by adding constraints that ensure that we select edges that form a (Steiner) tree. Several studies on Steiner Tree problem variants have shown that often directed edge models are better than undirected ones in solving Steiner Tree problems (e.g. [11, 4]). Following these results, we adapt the directed Dantzig-Fulkerson-Johnson formulation of connectivity. We have a binary variable for each directed edge in A (Constraint (24)). We can avoid explicitly including binary variables x for each node, as these decisions can be inferred from the values of the edge binary variables. The set of selected nodes consists of the nodes that have exactly one incoming edge. Although the vertex variables are not explicitly represented, it will still be useful to refer to them. To this effect, given a solution vector over the edge variables $\mathbf{y}$, let us define an *associated vertex solution vector* $\mathbf{x}$ as $x_k = \sum_{k\in\delta(i)} y_{ik}$. Constraints (18) and (19) express the objective and the budget constraint in terms of edge variables. Constraint (20) enforces that each terminal node should have one incoming edge (i.e. it should be selected). To enforce the directed tree property, each non-root node is allowed to have at most one incoming edge (Constraint (21)). Connectivity is enforced through generalized subtour elimination constraints defined over edge variables (Constraints (23)). We also include Constraint (22) which strengthens the formulation by enforcing that each edge is used in at most one direction.

$$\max \sum_{i\in V}\left(u_i \sum_{j\in\delta(i)} y_{ji}\right) \tag{18}$$

$$\text{s.t.} \sum_{i\in V}\left(c_i \sum_{j\in\delta(i)} y_{ji}\right) \le C \tag{19}$$

$$\sum_{j\in\delta(i)} y_{ji} = 1 \qquad \forall i \in T \tag{20}$$

$$\sum_{j\in\delta(i)} y_{ji} \leq 1 \qquad \forall i \in V - T \tag{21}$$

$$y_{ij} + y_{ji} \leq 1 \qquad \forall i \in V - T, \forall j \in \delta(i) - r \tag{22}$$

$$\sum_{(i,j)\in A | j\in S, i\in V\setminus S} y_{ij} \geq \sum_{j\in\delta(k)} y_{jk}, \qquad \forall S \subset V - r, \forall k \in S \; [cuts] \tag{23}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{24}$$

Given the exponential number of connectivity Constraints (23), in the following section we describe a solution approach in which we relax these constraints in the context of cutting plane procedure and only add them as cuts when they become violated.

4 Solution Approaches

Conrad et al. [2], Gomes et al. [7] outline a preprocessing technique for the Budget-constrained Connected Subgraph problem which effectively reduces the problem size for tight budgets. The procedure computes all-pairs shortest paths in the graph and uses these distances to compute for each node the minimal Steiner Tree cost that covers all three terminals as well as the node under consideration. If this minimum cost exceeds the allowed budget, the node does not belong to any feasible solution and hence its variable is assigned to 0.

Gomes et al. [7] also outline a greedy method for finding feasible solutions to the Budget-constrained Connected Subgraph problem by first computing the minimum cost Steiner tree covering all the terminal nodes and then greedily adding additional nodes until the allowed budget is exhausted. They show that providing this greedy solution to their encoding of the Connected Subgraph Problem (the single commodity flow encoding) significantly improves performance.

We use both of these techniques. We apply the preprocessing step to all problem instances. In addition, we provide the greedy solution as a starting point to the SCF encoding.

Our approach to solving the DFJ encoding is based on a cutting plane or Bender's decomposition approach. We solve a relaxed "master" problem which omits the exponential number of connectivity constraints. In a first pass of this procedure all edge variables are relaxed from binary variables to continuous variables $\in [0,1]$. In this first phase, we solve a sequence of progressively tighter LP master problems and in effect this corresponds to a cutting plane approach. Once we find a (fractional) optimal solution to the LP master problem that does not violate any connectivity constraints, we have obtained the optimal solution to the LP relaxation of the DFJ formulation. If that solution is integral, then we have an optimal solution to the original problem. If the LP solution is not integral, we enforce the integrality constraints for all edge variables. We continue the same iteration steps where now the master problem includes the cuts learned during solving the LP relaxation as well as the integrality constraints. In the second phase, we need to solve a sequence of MIP master problems which is in effect a Bender's decomposition approach. At each iteration, the optimal solution to the MIP master might not be connected and more connectivity cuts would need to be added. Once we find an optimal MIP master solution, we have found an optimal integer solution to the original problem. The detailed algorithm is outlined below:

Master Algorithm:
0. (Initialize) Define the initial relaxation P_0 of the problem by Constraints (18, 19,20, 21, 22) as well as the integrality Constraint (24) relaxed to only enforce the bounds. Set iteration count $t = 0$.
1. (Master optimization) Solve P_t and obtain an optimal (edge) solution $\mathbf{y}_t$. Let the associated vertex solution be $\mathbf{x}_t$. If the associated vertex solution $\mathbf{x}_t$ is integral, go to Step 3, otherwise go to Step 4.
2. (Additional Check) Check the connectivity of the induced graph $G(\mathbf{x}_t)$. If it is connected, then $\mathbf{x}_t$ is optimal, and the algorithm returns solution $\mathbf{x}_t$. Otherwise, continue to Step 4.
3. (Master separation) Check if $\mathbf{y}_t$ satisfies all the connectivity constraints (23). If it does, go to Step 4. If a violated constraint is found, then add the corresponding cut to the master problem and let P_{t+1} be the problem obtained. Set $t = t + 1$ and return to Step 1.
4. (Optimality check) If the associated vertex solution $\mathbf{x}_t$ is integral, then $\mathbf{x}_t$ is optimal, and the algorithm returns solution $\mathbf{x}_t$. Otherwise, add the integrality constraints (24) back in to the problem, and let P_{t+1} be the problem obtained. Set $t = t + 1$ and go to Step 1.

Checking the exponential number of connectivity constraints (23) given an edge solution $\mathbf{y}_t$ in Step 3 is done through a polynomial time separation procedure. The separation procedure checks the connectivity of each selected vertex to the root and terminates as soon as it finds a disconnected node and infers a cut to be added. It first checks the connectivity of the terminals to the root and then other selected vertices. We solve a max-flow problem in the directed graph G'=(V,A) between the root and each node $k \in V - r$ selected in the proposed solution, i.e. in the associated vertex solution $x_t(k) > 1 - \epsilon$. The capacities of the edges are the current values of the edge variables $\mathbf{y}_t$ in the master solution. If the maximum flow is less than the sum of the incoming arcs from k, we have found a violated constraint. The dual variables of the max-flow subproblem indicate the partition of nodes $\{S, V \backslash S\}$ that define the minimum cut (let $r \in V \backslash S$).

Now, we can add the cut enforcing that at least one edge across the partition needs to be selected if parcel k is selected:

$$\sum_{(i,j) \in A | i \in V \backslash S, j \in S} y_{ij} \geq x_k \tag{25}$$

Step 2 of the algorithm is a special step that applies to the Connected Subgraph Problems with node costs and node profits. Given a solution $\mathbf{y}_{ij}$ of the DFJ formulation and the associated vertex vector $\mathbf{x}_t$ we can infer a set of selected nodes $V' = \{k \in V | x_k(t) = 1\}$. The original problem only requires that for the selected subset of vertices V' the induced graph $G(V')$ is connected, while the DFJ formulation poses a much stronger requirement to select a subset of edges forming a tree. Hence, it can be the case that that V' induces a connected subgraph in G, but the selected edges $E' = \{(i,j) \in A | y_{ij} = 1\}$ do not form a single connected component. To illustrate this, imagine that the selected edges E' form two vertex-disjoint cycles C_1 and C_2 and such that $u \in C_1$ and $v \in C_2$ and $u, v \in E$. The edge set E' clearly does form a connected subgraph, however the subgraph induced by the selected vertices is connected because of the edge u, v. Without Step 2, our separation procedure in Step 3 will infer a new cut and will wrongly conclude that the selected master solution is not a feasible solution. To avoid such cases, we introduce Step 2 to check the weaker connectivity in terms of the induced subgraph. If this connectivity check fails, then we use the max-flow separation procedure in Step 3 to infer a new connectivity cut to add to the master.

The solution procedure described above solves a series of tighter relaxation of the original problem and therefore the first solution that is feasible w.r.t. all the constraints in the original problem is in fact the optimal solution. One problem with this approach is that we need to wait until the very end to get one integer feasible solution which is also the optimal one. Ideally, one would like to have integer feasible solutions as soon as possible. We achieve this in the context of this solution technique by noticing that while solving the MIP master to optimality we discover a sequence of integer solutions. Some of these integer solutions might satisfy all connectivity constraints (i.e. they are feasible solutions to the original problem), but are discarded by the master as sub-optimal – there might be disconnected solutions to the master of better quality. To detect the discovery of feasible solutions to the original problem while solving the master problem, we introduce a connectivity check at each MIP master incumbent solution (not described in our algorithm outline above). If a MIP master incumbent is connected and is better than any other connected integer solution discovered so far, we record this solution as an incumbent to our original problem.

5 Experimental Results

We evaluate the strength of the LP relaxation of the three alternative encodings on a synthetically generated benchmark of instances [2]. We generate 100 instances of a 10 by 10 grid parcels (100 nodes) and 3 reserves (terminals) with uniformly sampled costs and utilities. We report median running times across the 100 instances where the budget is varied as percentage slack over the minimum cost solution for the particular instance. For example, given a minimum cost solution for connecting the terminal nodes of value c_{min}, 10% slack corresponds to budget $B = 110\% * c_{min}$. All computational experiments were performed using IBM ILOG CPLEX 11[8].

Figure 1 compares the relative gap between the optimal objective of the LP relaxation z^*_{LP} and the optimal objective of the problem z^*_{IP} at different budget levels given by $(z^*_{LP} - z^*_{IP})/z^*_{IP}$. One can see that the DFJ encoding indeed provides a relaxation which is much tighter than the relaxation of the single flow formulation of the problem. In particular, the smaller the budget is (up to some point), the bigger advantage the exponential formulation has. This added strength however is paid in computational time. The LP relaxation of the SCF model is solved really fast compared to the DFJ encoding. On the other had, the multi-commodity encoding does not dominate on either measure – it provides tighter bound on the optimal but not as tight as the DFJ formulation but at the same time takes a considerable computational time. In the rest of the experimental analysis, we concentrate on the single commodity flow and the DFJ encoding. The DFJ-style encodings in the context of Steiner tree problems are known to produce tight LP relaxations. Our results confirm this trend in

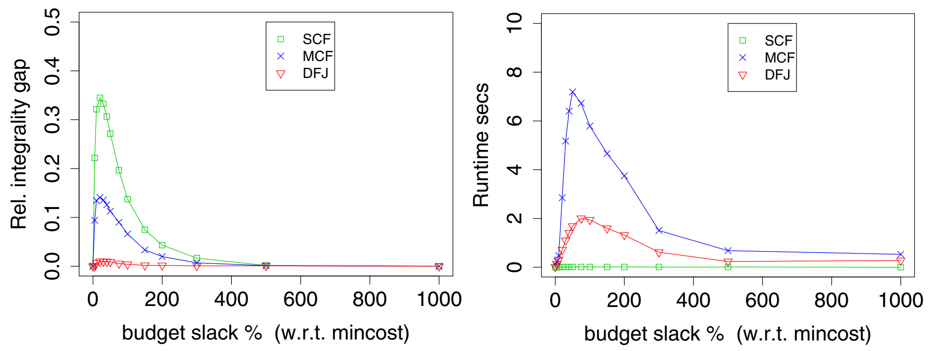

Fig. 1. Optimality gap and run times of LP relaxations of the three encodings on 10x10 lattices with 3 reserves, median over 100 runs

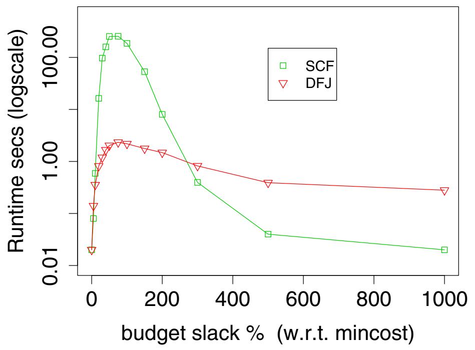

Fig. 2. Compare run times of DFJ and SCF for finding optimal integer solutions on 10x10 lattices with 3 reserves, median over 100 runs

the variant we are studying here. More importantly, one would like to use the strength of this encoding to find the optimal integer feasible solution.

Figure 2 compares the running time of the SCF encoding and the DFJ encoding. An easy-hard-easy pattern of the running time with respect to the budget was already observed by Gomes et al. [7]. Here, we clearly see that the DFJ encoding is in fact most beneficial in exactly the hard budget region. For large budgets, the DFJ encoding in fact has worst running time than the single commodity flow. However, more importantly it improves the running time in the hard region by 2 orders of magnitude.

We are interested in the running time performance of the SCF and the DFJ encoding when looking for integer feasible solutions. We evaluate the performance on a real-world Wildlife Corridor design problem attempting to connect three existing reserves. We tackle this problem at two different spatial scales. The coarser scale considers parcels grid cells of size 40 by 40 km and has 242 parcels (nodes). The finer spatial scale consider parcels of size 10 by 10 km and has 3299 parcels (nodes).

Figure 3 clearly demonstrates the advantage of the DFJ encoding on the 40km problem instance both in terms of the LP relaxation bound (left) and in terms of finding integer optimal solutions (right). The single flow encoding is fast for very tight and very large budgets, but for a critically constrained region the running time is much higher. The DFJ encoding on the other hand shows robust running times which do not vary much with the budget level.

We compare the running time to find integer solutions for the much larger instance at spatial resolution of 10 km. We set the budget at different (tight) levels as percent slack above the minimum cost required to connect the reserves. Table 1 presents solution quality, running times and optimality gap results for three different levels. For comparison, we also include the quality of the solution obtained by the greedy algorithm from [7] (which is usually much worse than the optimal). The results in Table 1 show that the DFJ encoding is much faster at finding optimal or near optimal solutions to the problem than the SCF encoding. Given a 8 hour cutoff time, for all three budget levels DFJ finds equal or better feasible solutions than SCF and also provides very tight optimality guarantee ($< 1\%$ in all cases). On the other hand, SCF in all three cases can only guarantee that the best solution it has founf is within at best 28% of optimality.

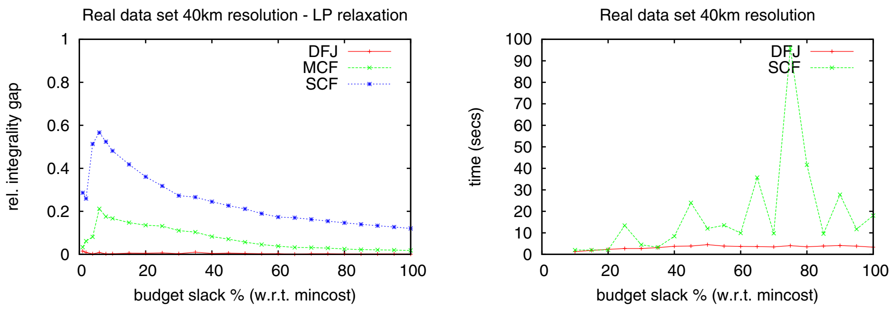

Fig. 3. Results on a Wildlife Corridor Problem at 40 km resolution

Table 1. The performance of the SCF and DFJ encoding on a large real world instance with an 8 hour cutoff time

budget slack	encoding	time	objective	opt. gap
10%	greedy	< 2 mins	10691163	NA
109475	SCF	8 hrs	10877799	31.15%
	DFJ	25 mins	12107793	0.01%
20%	greedy	< 2 mins	12497251	NA
119427	SCF	8 hrs	12911652	30.35%
	DFJ	2 hrs 25 mins	13640629	0.01%
30%	greedy	< 2 mins	13581815	NA
129379	SCF	8 hrs	13776496	28.64%
	DFJ	7 hrs 35 mins	14703920	0.62%

6 Conclusion

The budget-constrained Connection Subgraph Problem is computationally challenging problem with a lot of real world applications. Capturing well the combinatorial structure of the connectivity constraint is critical to effectively solving large scale instances. In this work, we proposed a novel solution approach to this problem that uses an adapted directed Dantzig-Fulkerson-Johnson formulation with subtour elimination constraints in the context of a cut-generation approach. This results in significant speed up in run times when the budget level falls in the interval that results in most computationally challenging instances. We evaluate performance on a relatively large instance of the Wildlife Corridor Design Problem and find optimal solutions for different budget levels. This work is a good example of identifying and extending relevant Computer Science results for problems arising in the area of Computation Sustainability.

Acknowledgments

This research was supported by IISI, Cornell University (AFOSR grant FA9550-04-1-0151), NSF Expeditions in Computing award for Computational Sustainability (Grant 0832782) and NSF IIS award (Grant 0514429).

References

[1] Benders, J.: Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 4, 238–252 (1962)

[2] Conrad, J., Gomes, C.P., van Hoeve, W.-J., Sabharwal, A., Suter, J.: Connections in networks: Hardness of feasibility versus optimality. In: Van Hentenryck, P., Wolsey, L.A. (eds.) CPAIOR 2007. LNCS, vol. 4510, pp. 16–28. Springer, Heidelberg (2007)

[3] Costa, A.M., Cordeau, J.-F., Laporte, G.: Steiner tree problems with profits. INFOR: Information Systems and Operational Research 4(2), 99–115 (2006)

[4] Costa, A.M., Cordeau, J.-F., Laporte, G.: Models and branch-and-cut algorithms for the steiner tree problem with revenues, budget and hop constraints. Networks 53(2), 141–159 (2009)

[5] Dantzig, G., Fulkerson, R., Johnson, S.: Solution of a Large-Scale Traveling-Salesman Problem. Operations Research 2(4), 393–410 (1954)

[6] Demaine, E.D., Hajiaghayi, M.T., Klein, P.: Node-weighted steiner tree and group steiner tree in planar graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555. Springer, Heidelberg (2009)

[7] Gomes, C.P., van Hoeve, W.-J., Sabharwal, A.: Connections in networks: A hybrid approach. In: Perron, L., Trick, M.A. (eds.) CPAIOR 2008. LNCS, vol. 5015, pp. 303–307. Springer, Heidelberg (2008)

[8] ILOG, SA, CPLEX 11.0 Reference Manual (2007)

[9] Lee, H.F., Dooly, D.R.: Decomposition algorithms for the maximum-weight connected graph problem. Naval Research Logistics 45(8), 817–837 (1998)

[10] Levin, A.: A better approximation algorithm for the budget prize collecting tree problem. Operations Research Letters 32(4), 316–319 (2004)

[11] Ljubic, I., Weiskircher, R., Pferschy, U., Klau, G.W., Mutzel, P., Fischetti, M.: Solving the prize-collecting steiner tree problem to optimality. In: ALENEX/ANALCO, pp. 68–76 (2005)

[12] McDaniel, D., Devine, M.: A Modified Benders' Partitioning Algorithm for Mixed Integer Programming. Management Science 24(3), 312–319 (1977)

[13] Moss, A., Rabani, Y.: Approximation algorithms for constrained node weighted steiner tree problems. SIAM J. Comput. 37(2), 460–481 (2007)

[14] Williams, J.C., Snyder, S.A.: Restoring habitat corridors in fragmented landscapes using optimization and percolation models. Environmental Modeling and Assessment 10(3), 239–250 (2005)