Analysis and Computational Study of Several Integer Programming Formulations for Minimum-Energy Multicasting in Wireless Ad Hoc Networks

Joanna Bauer [a] Dag Haugland [a,*]

[a] *Department of Informatics, University of Bergen, PB. 7800, N-5020 Bergen, Norway*

Di Yuan [b]

[b] *Department of Science and Technology, Linköping University, SE-601 74 Norrköping, Sweden*

[a] Supported by The Research Council of Norway under contract 160233/V30.

[b] Supported by CENIIT (Center for Industrial Information Technology), Linköping Institute of Technology, Sweden, and the Swedish Research Council under grant 621-2004-3902.

A multicast session in a wireless ad hoc network concerns routing messages from a source to a set of destination devices. Transmitting messages consumes energy at the source and intermediate devices of the session. Since a battery is the only energy source in many applications of wireless ad hoc networks, energy efficiency is an important performance measure of multicasting. In this paper, we present and analyze integer programming models for the problem of minimizing the total energy required by multicasting. We start from a straightforward multi-commodity flow model, which is strengthened by a more efficient representation of transmission power. Further strengthening is accomplished by lifting the capacity constraints of the model. We then present cut-based models for the problem, and prove, from a bounding standpoint, the equivalence in strength between these models and their flow-based counterparts. By expanding the underlying graph, we show that the problem can be transformed into finding a minimum Steiner arborescence. The expanded graph arises also in the separation procedure for solving one of the cut-based models. In addition to a theoretical analysis of the relation between various models, we perform extensive computational experiments to study the numerical strengths of these models and their efficiency in solving the problem.

Keywords: ad hoc networks; broadcasting; multicasting; integer programming

* Corresponding author.
Email addresses: Joanna.Bauer@ii.uib.no (Joanna Bauer), Dag.Haugland@ii.uib.no (Dag Haugland), diyua@itn.liu.se (Di Yuan).

1 INTRODUCTION

Wireless ad hoc networks are distinguished from both wired networks and wireless cellular systems. A wireless ad hoc network does not use a permanently installed infrastructure. A communication link is set up between two devices if they lie within transmission range of each other. The transmission range is determined by the power of the transmitting device. Typically, a device uses an omni-directional antenna. Applications of ad hoc networks range from emergency disaster relief to military command and control systems.

Since in many cases the only energy source of a device is a battery, it is important to minimize the energy required to accomplish communication tasks. In wireless networks, broadcasting is an inherent characteristic in message transmission, because the signal transmitted from one device reaches all other devices within the transmission range. Therefore, the power required at a transmitting device is the maximum rather than the sum of the powers needed to reach all intended recipients. This property is commonly referred to as the wireless multicast advantage [23,24].

A multicast session requires routing of messages from a source device to a set of destination devices. In the special case of a broadcast session, all devices except the source are destinations. To set up a multicast session, devices are assigned transmission powers such that any destination can receive messages either directly from the source, or through some intermediate devices. The total power consumption of a multicast session is the sum of the powers assigned to all devices. The Minimum-Energy Multicast Problem (MEMP) refers to minimizing the total power subject to the constraint that messages from the source are received by all destinations. Note the resemblance of this problem to the minimum Steiner tree problem (and the minimum spanning tree problem in the broadcast case), but also the important distinction implied by the wireless multicast advantage. The distinction has the consequence that MEMP is $\mathcal{NP}$-hard [5] even in the broadcast case.

A wireless network can be modeled as a graph. The nodes represent devices, and the cost of a link represents its power requirement. Commonly, the power requirement is the same for both directions of a link. A multicast session can be represented by a tree rooted at the source and spanning all destinations. At each node, the link connecting this node to its most power-demanding child defines the cost at the node. The total cost of the tree equals the sum of the node costs.

For the broadcast version of MEMP, an obvious heuristic is to compute the minimum spanning tree (MST). The MST is feasible, but in general not optimal to MEMP. Wieselthier et al. [23,24] proposed the Broadcast Incremental Power (BIP) heuristic. Starting from the source, BIP builds up a tree by adding the node requiring a minimum amount of incremental power in every iteration. Numerical experiments show that this tree-construction heuristic results in a lower total energy consumption than that of MST. In addition to a heuristic for tree construction, Wieselthier et al. also proposed a power-saving heuristic, called sweep, to detect and eliminate unnecessary transmissions.

BIP (and MST) may fail to reach optimality for a network containing as few as four nodes. An example is shown in Figure 1. Four nodes are located at $(0, 0)$, $(6, 0)$ $(7, 6)$, and $(4, 8)$, respectively. Node 1 is the source. In the example, the transmission power required at a node to reach another equals the square of the Euclidean distance between them, which is a common assumption for wireless networks [23]. The power requirements are shown beside the links. BIP starts from the source and adds node

2. Then, node 3 is selected and added to the tree via link $(2, 3)$. To add node 4, the minimum incremental power is given by link $(3, 4)$. The sweep procedure in [23,24] does not give any improvement in power. Thus BIP outputs a tree having a total power of 86, and in this example, the BIP tree coincides with the MST. The optimal solution of MEMP is however to let the source transmit directly to all nodes, resulting in a total power of 85.

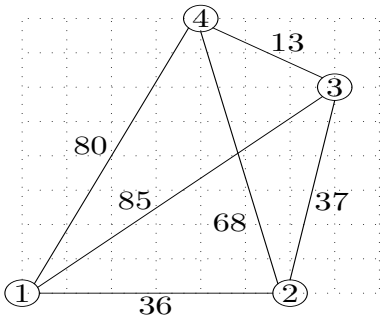

FIG. 1. A small network example.

Under the assumption that all nodes are points in the plane, and the power requirements are proportional to the square of the distances between the points, theoretical analysis of BIP has been conducted in a number of references. Wan et al. [21] showed that BIP has constant approximation ratio at least as large as 13/3. Recently, we have strengthened this lower bound from 13/3 to 4.6 [4]. Based on the work in [21], Klasing et al. [15] showed that the approximation ratio is no larger than 12.15. Ambühl [2] proved that this bound can be reduced to 6. Note that the bounds in [2,15] are derived for the MST heuristic, but these results also apply to BIP due to a lemma in [21].

For multicast, Wieselthier et al. [23,24] introduced the Multicast Incremental Power (MIP) heuristic, which is BIP followed by a procedure called *pruning*. This procedure detects nodes that are not involved in relaying messages to the destinations, removes them from the tree, and sets their power values to zero. Node removal also enables power reduction at nodes remaining in the tree. In [22], Wan et al. showed that MIP does not have a constant approximation ratio. They also showed that algorithms with this property can be designed by adapting any approximation algorithm for the minimum Steiner tree problem, provided that the latter algorithm has constant approximation ratio. Examples of such approximation algorithms are given e.g. by Hwang et al. [13].

In addition to the algorithms discussed so far, heuristics for MEMP have been proposed in [6,7,9–11,16–19]. Most of these heuristics apply to the broadcast version of MEMP. In these references, the BIP and sometimes the MIP heuristic have been used to provide benchmark results.

In contrast to the rich literature on solving MEMP by approximation algorithms and heuristics, approaches based on mathematical programming have been explored to a lesser extent. Das et al. [8] presented three linear integer programming models for MEMP. Among them, a model based on single-commodity flows has been cited in several other papers (e.g., [15]). Yuan [26] presented a model using multi-commodity flow and a Lagrangean relaxation scheme to numerically evaluate the performance of the BIP heuristic in large networks. For the broadcast version of MEMP, Altinkemer et al. [1] formulated a set-covering model and presented numerical results of a Lagrangean heuristic.

Integer programming formulations can be useful for obtaining exact or approximate solutions to MEMP. They also provide insight into the relation between MEMP and classical network design problems. In either case, it is relevant to study the *strength* of an integer programming model, i.e., the tightness of the bound obtained from solving its continuous relaxation. There are a couple of advantages to considering strong models. First, a stronger model often leads to shorter solution time in obtaining an integer optimum. Second, for large instances in which an integer optimum is out of reach, performance evaluation of heuristics can instead be carried out using the continuous relaxation of a strong model.

Solving an integer model typically requires centralized computation. For this reason, integer programming models are not intended for on-line use in ad hoc networks. For on-line computation, simple and often distributed heuristics are necessary. Integer programming models are of great value for evaluating those heuristics. Without access to the optimal solution, or a strong lower bound, it is difficult to judge the performance of a heuristic. A second, theoretical motivation originates from the apparent similarity between MEMP and the Steiner tree problem.

In this paper, we study a number of linear integer programming models that either use flows or cuts to characterize feasible solutions of MEMP. The contributions of this paper consist of the following. Starting from a straightforward multi-commodity flow model, we present two new, stronger flow-based models. The first model adopts a more efficient representation of transmission power, and the second model involves a further strengthening accomplished by lifting the capacity constraints. We also present two cut-based models of MEMP. We conduct a thorough analysis of the strengths of all models. In particular, we prove the equivalence in strength between the cut-based models and their flow-based counterparts. A result of our study is the strongest known model for MEMP. Moreover, we provide the insight that MEMP can be treated as the classical problem of determining a minimum Steiner arborescence in an expanded directed graph. We show that this graph also arises in the separation procedure for solving one of the cut-based models. In addition to a theoretical analysis, we perform extensive computational experiments to numerically examine the strengths of the models, and to identify models that are efficient in finding optimal or near-optimal solutions.

The remainder of the paper is organized as follows. In the next section we introduce some notation and formalize MEMP. Sections 3 and 4 treat the flow- and cut-based models, respectively. These sections contain our theoretical analyses of the strengths of these models. Section 5 is mainly devoted to the Steiner arborescence reformulation of MEMP. In Section 6, we report the results of our computational experiments. Conclusions are drawn in Section 7.

2 PRELIMINARIES

To discuss integer programming models of MEMP, some notation defined for an arbitrary directed graph G is useful. The sets of nodes and arcs of G are denoted by V_G and A_G, respectively. If $S \subseteq V_G$, then $\bar{S}$ denotes the complementary set of nodes, i.e., $\bar{S} = V_G \setminus S$. For a vector $u \in \Re^{|A_G|}$, we use either u_a or u_{ij}, whichever is more convenient, to denote the component of u corresponding to arc $a = (i, j)$. For a vector $b \in \Re^{|V_G|}$, we use b_i to denote the component of b corresponding to node $i \in V_G$. We use $A_G^+(i) \subseteq A_G$ and $A_G^-(i) \subseteq A_G$ to denote the sets of arcs of which i is the tail and head node, respectively.

Let n be the number of nodes not counting the source i_0, and denote the node set $V_G = \{i_0, i_1, \ldots, i_n\}$. The set of destinations is denoted by $D \subseteq V_G \setminus \{i_0\}$. We let $c \in \Re_+^{|A_G|}$ denote the power parameters, and consequently c_{ij} is the power required at node i to reach node j.

We use the term *multi-commodity flow* to refer to a set of distinguished flows. A commodity in MEMP corresponds to a destination node $d \in D$. We define a demand vector $b^d \in \Re^{|V_G|}$ for each $d \in D$, where $b^d_{i_0} = -1$, $b^d_d = 1$, and $b^d_i = 0$ for $i \in V_G \setminus \{i_0, d\}$. A flow vector of the commodity corresponding to destination d is denoted by $f^d \in \Re_+^{|A_G|}$. For $d \in D$, we let $\mathcal{F}(G, b^d)$ denote the set of flow vectors $f^d \in \Re_+^{|A_G|}$ satisfying the flow conservation equations $\sum_{a \in A_G^+(i)} f^d_a - \sum_{a \in A_G^-(i)} f^d_a = -b^d_i \ \forall i \in V_G$. Furthermore, we define the flow polytope $\mathcal{F}\left(G, b^d, u^d\right) = \left\{f^d \in \mathcal{F}(G, b^d) : f^d_a \leq u^d_a \ \forall a \in A_G\right\}$, where $u^d \in \Re_+^{|A_G|}$ is a vector of upper bounds on arc flows of the given commodity.

Assume that $G = (V_G, A_G)$ is complete, and let it represent a wireless ad hoc network. Problem MEMP is defined as follows:

[**MEMP**] Find a power assignment $p \in \Re_+^{|V_G|}$ minimizing $\sum_{i \in V_G} p_i$, such that for all destinations $d \in D$ there exists at least one path from i_0 to d with $p_i \geq c_{ij}$ for all arcs (i, j) in the path.

In most of our models, it is necessary to process the power levels at a node in non-decreasing order. Let $N = \{1, \ldots, n\}$. For all $i \in V_G$, let $\pi_i : N \mapsto V_G \setminus \{i\}$ be a bijection such that $\left(c_{i,\pi_i(1)}, \ldots, c_{i,\pi_i(n)}\right)$ is monotonously non-decreasing. Following the definition of π_i, $\pi_i(k)$ is the kth closest node to i, where distance is measured by power. For any $u \in \Re^{|A_G|}$ and $k \in N$, $u_{(ik)}$ is, whenever convenient, used as a short-hand notation for $u_{i,\pi_i(k)}$. We use π'_i to denote the inverse of π_i. In other words, when the nodes are sorted in ascending order with respect to their distances to i, $\pi'_i(j)$ denotes the position of $j \in V_G$ in the sorted sequence. We also define $\pi'_i(S) = \min\{\pi'_i(j) : j \in S\}$ for any $S \subseteq V_G \setminus \{i\}$.

If $\mathcal{M}$ is a (mixed) integer programming model for MEMP, and $\mathcal{I}$ is a MEMP instance, then $\mathcal{M}^{\mathbf{LP}}$ denotes the continuous relaxation of $\mathcal{M}$, while $\zeta(\mathcal{M}, \mathcal{I})$ and $\zeta^*(\mathcal{I})$ denote the optimal objective function values of $\mathcal{M}^{\mathbf{LP}}$ and $\mathcal{M}$, respectively, instantiated with $\mathcal{I}$.

3 FORMULATIONS USING MULTI-COMMODITY FLOW

Feasible solutions to MEMP can be characterized by network flows. A single-commodity flow model, like the one proposed by Das et al. in [8], is a straightforward way of using flow to formulate MEMP. For most network design problems (e.g., [3]), a single-commodity flow model can be strengthened by adopting multi-commodity flow. We therefore start our discussion with a multi-commodity flow model, denoted by **F0**, which is a natural strengthening of the model in [8]. Formulation **F0** uses three sets of variables, flow variables $\{f^d_{ij} : (i, j) \in A_G, d \in D\}$, design variables $\{z_{ij} : (i, j) \in A_G\}$, and power variables $\{p_i : i \in V_G\}$:

$$
\begin{aligned}
[\mathbf{F0}] \quad \min \quad & \sum_{i \in V_G} p_i \\
\text{s.t.} \quad & c_{ij} z_{ij} \leq p_i \quad \forall (i,j) \in A_G, && (1) \\
& f^d \in \mathcal{F}(G, b^d, z) \quad \forall d \in D, && (2) \\
& z \in \{0,1\}^{|A_G|}.
\end{aligned}
$$

Due to (2), all destinations $d \in D$ receive one unit of flow from the source, and the arc (i,j) can carry flow only if $z_{ij} = 1$. The design variables are linked to the node power by (1).

We can improve the strength of the formulation **F0** by representing the power assignment in a more efficient way. Note that in any optimal solution, the power of node i is either zero, or equals one of the values in the discrete set $\{c_{(i1)}, c_{(i2)}, \ldots, c_{(in)}\}$. We can therefore associate a set of binary variables to the power values, and represent the power of a node as a scalar product. Doing so results in a stronger formulation **F1**. Let

$$
y_{(ik)} = \begin{cases} 1, \text{ if node } i \text{ is assigned the power } c_{(ik)}, \\ 0, \text{ otherwise.} \end{cases}
$$

$$
\begin{aligned}
[\mathbf{F1}] \quad \min \quad & \sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} y_{(ik)} \\
\text{s.t.} \quad & f^d_{(ik)} \leq \sum_{\ell=k}^{n} y_{(i\ell)} \quad \forall i \in V_G, k \in N, d \in D, && (3) \\
& f^d \in \mathcal{F}(G, b^d) \quad \forall d \in D, && (4) \\
& y \in \{0,1\}^{|A_G|}.
\end{aligned}
$$

The constraints (3) restrict the assignment of flow to those arcs $(i, \pi_i(k))$ where i has a power assignment of at least $c_{(ik)}$. In **F1**, we no longer need the continuous variables p to express node power.

We can strengthen **F1** further. Since at most one unit of flow to destination d leaves node i, we can include all the flow variables for which the corresponding transmission powers are greater than or equal to $c_{(ik)}$ in the left-hand sides of (3). This yields a stronger flow formulation:

$$
\begin{aligned}
[\mathbf{F2}] \quad \min \quad & \sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} y_{(ik)} \\
\text{s.t.} \quad & \sum_{\ell=k}^{n} f^d_{(i\ell)} \leq \sum_{\ell=k}^{n} y_{(i\ell)} \quad \forall i \in V_G, k \in N, d \in D, && (5) \\
& f^d \in \mathcal{F}(G, b^d) \quad \forall d \in D, && (6)
\end{aligned}
$$

$$y \in \{0,1\}^{|A_G|}.$$

Clearly, the inequalities $\zeta(\mathbf{F0},\mathcal{I}) \leq \zeta(\mathbf{F1},\mathcal{I}) \leq \zeta(\mathbf{F2},\mathcal{I})$ hold for any instance $\mathcal{I}$. Our computational experiments show that both inequalities can be strict, and that the gaps can be significant.

The following inequalities are valid in **F1** and **F2**:

$$\sum_{k=1}^{n} y_{(ik)} \leq 1 \quad \forall i \in V_G. \tag{7}$$

Although adding (7) to **F1** and **F2** reduces the sets of feasible solutions, these inequalities do not alter the set of optimal integer solutions. Moreover, as will be proved below, they do not make the continuous relaxation of **F1** or **F2** any stronger. However, inequalities (7) may be useful for strengthening the subproblem when applying a Lagrangean relaxation technique [26].

Proposition 1 $\mathbf{F1^{LP}}$ *and* $\mathbf{F2^{LP}}$ *have an optimal solution* (f,y) *satisfying (7).*

Proof. It is obvious that $\mathbf{F1^{LP}}$ ($\mathbf{F2^{LP}}$) has at least one optimal solution. Let $(\bar{f},\bar{y})$ be one such solution, and let A_C be the set of arcs of the directed cycle C in G. Then, let f be a reduction of $\bar{f}$ such that for all $d \in D$ and all directed cycles C in G, f^d_a equals 0 for at least one $a \in A_C$. For all $i \in V_G$, let $y_{(i\ell)} = \bar{y}_{(i\ell)}$ $\forall \ell = 1,\ldots,n$ if $\sum_{\ell=1}^{n} \bar{y}_{(i\ell)} \leq 1$. Otherwise, let $k' = \max\left\{k \in N : \sum_{\ell=k}^{n} \bar{y}_{(i\ell)} > 1\right\}$, let $y_{(i\ell)} = \bar{y}_{(i\ell)}$ $\forall \ell = n, n-1,\ldots,k'+1$, $y_{(ik')} = 1 - \sum_{\ell=k'+1}^{n} \bar{y}_{(i\ell)}$, and $y_{(i,k'-1)} = \cdots = y_{(i1)} = 0$.

By construction, $f^d_a \leq \bar{f}^d_a$ $(a \in A_G, d \in D)$. Consider $\mathbf{F2^{LP}}$. Since the solution $(\bar{f},\bar{y})$ satisfies (5), we have $\sum_{\ell=k}^{n} f^d_{(i\ell)} \leq \sum_{\ell=k}^{n} \bar{y}_{(i\ell)}$ $(i \in V_G, k \in N, d \in D)$. Replacing $\bar{y}$ by y either leaves the right-hand side of (5) unchanged, or reduces it to one. Since f does not contain cyclic flow, the left-hand side of (5) does not exceed one, and therefore $\sum_{\ell=k}^{n} f^d_{(i\ell)} \leq \sum_{\ell=k}^{n} y_{(i\ell)}$, i.e., the solution (f,y) is feasible in $\mathbf{F2^{LP}}$. A similar argument applies to $\mathbf{F1^{LP}}$. In addition, it is clear that $\sum_{k=1}^{n} y_{(ik)} \leq 1$ and $\sum_{k=1}^{n} c_{(ik)} y_{(ik)} \leq \sum_{k=1}^{n} c_{(ik)} \bar{y}_{(ik)}$ for all $i \in V_G$. $\square$

4 CUT-BASED FORMULATIONS

Dualism between flow and cut is an important concept in dealing with network optimization problems. Instead of flow conservation constraints, conditions on cuts separating the source from the destinations can be imposed to enforce the connectivity between i_0 and D. We present two cut-based formulations for MEMP, and discuss the strengths of the two formulations in relation to their flow-based counterparts.

Any binary vector y satisfying (7) defines a feasible solution to a MEMP instance, if and only if G contains a path from i_0 to d for all $d \in D$, such that $\sum_{k=\pi'_i(j)}^{n} y_{(ik)} \geq 1$ for each arc (i,j) on the path. In **F1** and **F2**, this is ensured by (3)-(4) and (5)-(6),

respectively. In a cut-based formulation, the feasibility of y can be stated by the condition that any set $S \subset V_G$ for which $i_0 \in S$ and $\bar{S} \cap D \neq \emptyset$, contains a node with sufficient power to reach some node in $\bar{S}$. This condition is both sufficient and necessary to characterize all feasible solutions to MEMP.

Proposition 2 *A binary design vector y is feasible in MEMP if and only if for any $S \subset V_G$ such that $i_0 \in S$ and $\bar{S} \cap D \neq \emptyset$, the arc set $\{(i,j) : i \in S, j \in \bar{S}, \sum_{k=\pi'_i(j)}^{n} y_{(ik)} \geq 1\}$ is non-empty.*

The proof of the proposition is straightforward and is therefore omitted. Note that the condition in Proposition 2 is that at least one arc in the cut defined by S and $\bar{S}$ is assigned capacity one. Formulating the condition by stating that the sum of $\sum_{k=\pi'_i(j)}^{n} y_{(ik)}$ over the arcs (i,j) in the cut is at least one, yields our first cut-based formulation for MEMP:

$$
\begin{aligned}
[\mathbf{C1}] \quad \min \quad & \sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} y_{(ik)} \\
\text{s.t.} \quad & \sum_{i \in S} \sum_{j \in \bar{S}} \sum_{k=\pi'_i(j)}^{n} y_{(ik)} \geq 1 \quad \forall S \subset V_G : i_0 \in S, \bar{S} \cap D \neq \emptyset, \qquad (8) \\
& y \in \{0,1\}^{|A_G|}.
\end{aligned}
$$

In **C1**, a variable may appear more than once (i.e., it has a coefficient greater than one) in a constraint (8). Removing multiple occurrences of a variable in the same constraint leads to a stronger cut-based formulation, **C2**. Recall that in the sorted power vector of node i, $\pi'_i(\bar{S})$ is the position corresponding to the minimum power required at i to reach some node in $\bar{S}$.

$$
\begin{aligned}
[\mathbf{C2}] \quad \min \quad & \sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} y_{(ik)} \\
\text{s.t.} \quad & \sum_{i \in S} \sum_{k=\pi'_i(\bar{S})}^{n} y_{(ik)} \geq 1 \quad \forall S \subset V_G : i_0 \in S, \bar{S} \cap D \neq \emptyset, \qquad (9) \\
& y \in \{0,1\}^{|A_G|}.
\end{aligned}
$$

Clearly, $\zeta(\mathbf{C1}, \mathcal{I}) \leq \zeta(\mathbf{C2}, \mathcal{I})$ holds for any instance $\mathcal{I}$. Note that the inequalities (7) are valid in both **C1** and **C2**. Similar to their role in the flow-based formulations, inequalities (7) neither are necessary for defining the integer optimum, nor do they improve the continuous relaxations of **C1** or **C2**. We omit the proof of this result, because it is similar to that of Proposition 1.

The two cut-based formulations, **C1** and **C2**, are tightly connected to the two flow-based formulations **F1** and **F2**. We end this section by proving that **C1** and **F1** are equally strong.

Proposition 3 $\zeta(\mathbf{F1}, \mathcal{I}) = \zeta(\mathbf{C1}, \mathcal{I})$ *for any instance $\mathcal{I}$ of MEMP.*

Proof. Consider any $y \in [0,1]^{|A_G|}$, and assign capacity $\sum_{k=\pi'_i(j)}^{n} y_{(ik)}$ to the arc $(i,j) \in A_G$. Clearly, y is feasible in $\mathbf{C1^{LP}}$ (i.e., y satisfies (8)) if and only if all cuts separating i_0 from some destination have at least unit capacity, which in turn is true if and only if the maximum flow from i_0 to all $d \in D$ is at least one. If the maximum flow arriving at $d \in D$ is greater than or equal to one, then this flow (scaled down to unit flow, if necessary) together with y constitute a feasible solution to $\mathbf{F1^{LP}}$. Conversely, a feasible solution (f, y) to $\mathbf{F1^{LP}}$ corresponds to a flow of one unit for every $d \in D$, and therefore y is feasible in $\mathbf{C1^{LP}}$. The proposition then follows from the fact that **F1** and **C1** have identical objective functions. □

For **F2** and **C2**, it is less straightforward to show their relation in strength. In the next section we discuss a Steiner arborescence model for MEMP. This model serves as the link we need for proving that $\zeta(\mathbf{F2}, \mathcal{I}) = \zeta(\mathbf{C2}, \mathcal{I})$.

5 ALTERNATIVE MODELS FOR MEMP

5.1 A Steiner Arborescence Model

Given a source node and a set of terminal nodes in a directed graph with arc weights, the Steiner arborescence problem [25] amounts to finding an arborescence that is rooted at the source and spans the set of terminal nodes, such that the total weight of the arcs in the arborescence is minimized. We show that MEMP can be formulated as a Steiner arborescence problem in a directed graph H, where H is an expansion of G.

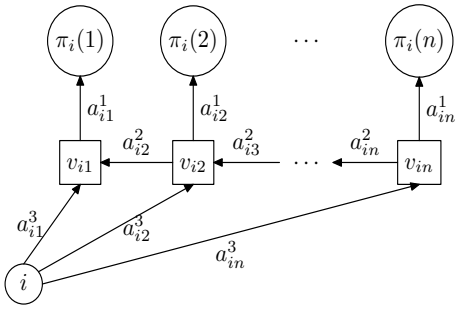

FIG. 2. An illustration of the graph expansion.

The node and arc sets of H are denoted $V_H = V'_H \cup V_G$ and $A_H = A^1_H \cup A^2_H \cup A^3_H$, respectively. For an arbitrary node $i \in V_G$, the graph expansion is illustrated in Figure 2. The additional node set V'_H is the union of the "square" nodes for all $i \in V_G$, that is, $V'_H = \{v_{ik} : i \in V_G, k \in N\}$. The three parts of the arc set A_H are defined as follows: $A^1_H = \{a^1_{ik} = (v_{ik}, \pi_i(k)) : i \in V_G, k \in N\}$, $A^2_H = \{a^2_{ik} = (v_{ik}, v_{i,k-1}) : i \in V_G, k = 2, \ldots, n\}$, and $A^3_H = \{a^3_{ik} = (i, v_{ik}) : i \in V_G, k \in N\}$. In Figure 2, these sets correspond to the vertical, horizontal, and diagonal arcs, respectively.

MEMP is equivalent to a minimum Steiner arborescence problem in H, where i_0 is the source and D is the set of terminal nodes. The arcs in A^1_H and A^2_H have zero weights, whereas the weight of arc $a^3_{ik} \in A^3_H$ equals $c_{(ik)}$, $i \in V_G, k \in N$. The problem can be formulated as follows:

$$
\begin{aligned}
[\mathbf{S}] \quad \min \quad & \sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} Y_{a^3_{ik}} \\
\text{s.t.} \quad & h^d \in \mathcal{F}(H, B^d, Y) \quad \forall d \in D, && (10) \\
& Y \in \{0,1\}^{|A_H|}. && (11)
\end{aligned}
$$

The variables of **S** are the flow vectors $h^d \in \Re_+^{|A_H|}$, $d \in D$, and the binary vector $Y \in \{0,1\}^{|A_H|}$. The demand vector $B^d \in \Re^{|V_H|}$ is an extension of b^d such that $B^d_v = 0$ for all $v \in V'_H$.

Because the size of **S** is much larger than that of **F2**, the former is likely to be less efficient in terms of solving MEMP by an integer programming solver. There are however several reasons for considering **S**. First, theory and methodolgy available in the literature on the Steiner arborescence problem, and the more general problem class of uncapacitated network design [3] can be applied to MEMP. Second, we use **S** to prove the equivalence in strength between **F2** and **C2**. Third, the proof shows that the expanded graph H is needed to implement a constraint-generation scheme for solving **C2**.

In the remainder of this section, we use **S** as the link between **F2** and **C2**. The following lemma is useful to prove that **S** is exactly as strong as **F2**.

Lemma 4 *Consider any $d \in D$, $f^d \in \mathcal{F}(G, b^d)$, and $Y \in [0,1]^{|A_H|}$. If $Y_a = 1$ $\forall a \in A^1_H \cup A^2_H$ and $\sum_{\ell=k}^{n} f^d_{(i\ell)} \leq \sum_{\ell=k}^{n} Y_{a^3_{i\ell}} \leq 1$ $\forall (i,k) \in V_G \times N$, then there exists some $h^d \in \mathcal{F}(H, B^d, Y)$ such that $h^d_{a^1_{ik}} = f^d_{(ik)}$ $\forall (i,k) \in V_G \times N$.*

Proof. For convenience, we omit the superscript d on f, h and B. Consider any $i \in V_G$. We use f and Y to define the flow h by maximizing the flow on arcs $a^3_{in}, \ldots, a^3_{i1}$, in the given order, while respecting the following conditions:

$$
\begin{aligned}
h_{a^1_{ik}} &= f_{(ik)} && \forall i \in V_G, k = 1, \ldots, n \\
h_{a^3_{in}} &= \min \left\{ Y_{a^3_{in}}, \sum_{\ell=1}^{n} h_{a^1_{i\ell}} \right\} && (12) \\
h_{a^3_{ik}} &= \min \left\{ Y_{a^3_{ik}}, \sum_{\ell=1}^{n} h_{a^1_{i\ell}} - \sum_{\ell=k+1}^{n} h_{a^3_{i\ell}} \right\} && \forall i \in V_G, k = n-1, \ldots, 1 \quad (13) \\
h_{a^2_{ik}} &= \sum_{\ell=k}^{n} h_{a^3_{i\ell}} - \sum_{\ell=k}^{n} h_{a^1_{i\ell}} && \forall i \in V_G, k = 2, \ldots, n \quad (14)
\end{aligned}
$$

We first show that h is non-negative, and does not exceed Y. Clearly, $h_a \in [0, Y_a]$ $\forall a \in A^1_H$. The constructions in (12) and (13) imply that $\sum_{\ell=1}^{n} h_{a^1_{i\ell}} \geq \sum_{\ell=k}^{n} h_{a^3_{i\ell}}$ $\forall k \in N$. Consequently, $h_{a^3_{ik}} \geq 0$ $\forall k \in N$, since it is the minimum of two non-negative numbers. Hence $h_a \in [0, Y_a]$ $\forall a \in A^3_H$. Furthermore, $\sum_{\ell=k}^{n} h_{a^3_{i\ell}} = \sum_{\ell=1}^{n} h_{a^1_{i\ell}}$ if the minimum in (13) equals $\sum_{\ell=1}^{n} h_{a^1_{i\ell}} - \sum_{\ell=k'+1}^{n} h_{a^3_{i\ell}}$ for some $k' \geq k$, otherwise

$\sum_{\ell=k}^{n} h_{a^3_{i\ell}} = \sum_{\ell=k}^{n} Y_{a^3_{i\ell}}$. Therefore, either $h_{a^2_{ik}} = \sum_{\ell=1}^{n} h_{a^1_{i\ell}} - \sum_{\ell=k}^{n} h_{a^1_{i\ell}} = \sum_{\ell=1}^{k-1} f_{(i\ell)}$, or $h_{a^2_{ik}} = \sum_{\ell=k}^{n} Y_{a^3_{i\ell}} - \sum_{\ell=k}^{n} h_{a^1_{i\ell}} = \sum_{\ell=k}^{n} Y_{a^3_{i\ell}} - \sum_{\ell=k}^{n} f_{(i\ell)}$. As a result, $h_a \in [0, Y_a]\ \forall a \in A^2_H$.

Next, we prove flow conservation. Consider first the total flow emanating from node i. Because $\sum_{\ell=1}^{n} h_{a^3_{i\ell}} \in \left\{\sum_{\ell=1}^{n} h_{a^1_{i\ell}}, \sum_{\ell=1}^{n} Y_{a^3_{i\ell}}\right\}$ and $\sum_{\ell=1}^{n} h_{a^1_{i\ell}} = \sum_{\ell=1}^{n} f_{(i\ell)} \leq \sum_{\ell=1}^{n} Y_{a^3_{i\ell}}$, it follows from the minimization in (13) that $\sum_{\ell=1}^{n} h_{a^3_{i\ell}} = \sum_{\ell=1}^{n} f_{(i\ell)}$. The total flow entering node i is $\sum_{a\in A^-_H(i)} h_a = \sum_{j\in V_G\setminus\{i\}} f_{\left(j,\pi'_j(i)\right)}$. Thus h satisfies flow conservation at all $i \in V_G$ because f does. Moreover, for any $v_{ik} \in V'_H$, flow conservation follows directly from Figure 2 and (14). Hence $h \in \mathcal{F}(H, B, Y)$. $\square$

The proposition below establishes the strong connection between feasible flow vectors in **F2** and those in **S**. Specifically, if $Y_{a^3_{ik}} = y_{(ik)}\ \forall (i,k) \in V_G \times N$, then projecting flow vectors satisfying (10) onto the arc set A^1_H gives the feasible set of flow vectors in **F2**.

Proposition 5 *Consider any $d \in D$, and assume that $y \in [0,1]^{|A_G|}$ and $Y \in [0,1]^{|A_H|}$ satisfy $\sum_{k=1}^{n} y_{(ik)} \leq 1\ \forall i \in V_G$, $Y_a = 1\ \forall a \in A^1_H \cup A^2_H$, and $Y_{a^3_{ik}} = y_{(ik)}\ \forall (i,k) \in V_G \times N$. Then the projection of $\mathcal{F}(H, B^d, Y)$ onto A^1_H is exactly the set of flow vectors in $\mathcal{F}(G, b^d)$ satisfying $\sum_{\ell=k}^{n} f^d_{(i\ell)} \leq \sum_{\ell=k}^{n} y_{(i\ell)}\ \forall i \in V_G, k \in N$.*

Proof. We omit the superscript d on f, h, b and B. Consider any $h \in \mathcal{F}(H, B, Y)$, and set $f_{(ik)} = h_{a^1_{ik}}\ \forall i \in V_G,\ k \in N$. We show that $f \in \mathcal{F}(G, b)$ and $\sum_{\ell=k}^{n} f_{(i\ell)} \leq \sum_{\ell=k}^{n} y_{(i\ell)}\ \forall i \in V_G,\ k \in N$.

First, note that the total flow entering the node set $\{v_{ik} : k \in N\}$ equals the total flow emanating from this set, i.e., $\sum_{k\in N} h_{a^3_{ik}} = \sum_{k\in N} h_{a^1_{ik}}$, because the set contains neither the source nor any destination. As h satisfies conservation of flow at i, we have $\sum_{a\in A^-_G(i)} f_a = \sum_{a\in A^-_H(i)} h_a = \sum_{k\in N} h_{a^3_{ik}} + b_i = \sum_{k\in N} h_{a^1_{ik}} + b_i = \sum_{k\in N} f_{(ik)} + b_i = \sum_{a\in A^+_G(i)} f_a + b_i$. Therefore, $f \in \mathcal{F}(G, b)$. Moreover, conservation of flow at $\{v_{ik} : k \in N\}$ implies $\sum_{\ell=k}^{n} f_{(i\ell)} = \sum_{\ell=k}^{n} h_{a^1_{i\ell}} = \sum_{\ell=k}^{n} h_{a^3_{i\ell}} - h_{a^2_{ik}} \leq \sum_{\ell=k}^{n} h_{a^3_{i\ell}} \leq \sum_{\ell=k}^{n} Y_{a^3_{i\ell}} = \sum_{\ell=k}^{n} y_{(i\ell)}$.

Next, assume that $f \in \mathcal{F}(G, b)$ and $\sum_{\ell=k}^{n} f_{(i\ell)} \leq \sum_{\ell=k}^{n} y_{(i\ell)}\ \forall k \in N$. By Lemma 4, f is the projection of some $h \in \mathcal{F}(H, B, Y)$ onto A^1_H. $\square$

The following proposition states that **S** and **C2** are equally strong. The key to establish the proof is to observe that the left-hand side of (9) in **C2** does not correspond to a cut in the graph G, but in the expanded graph H.

Proposition 6 *$\zeta(\mathbf{S}, \mathcal{I}) = \zeta(\mathbf{C2}, \mathcal{I})$ for any instance $\mathcal{I}$ of MEMP.*

Proof. Consider any $Y \in [0,1]^{|A_H|}$ for which there exists a flow h satisfying (10). We first show that setting $y_{(ik)} = Y_{a^3_{ik}}\ \forall (i,k) \in V_G \times N$ yields a feasible solution to $\mathbf{C2^{LP}}$. Consider any $S \subset V_G$ such that $i_0 \in S$ and $\bar{S} \cap D \neq \emptyset$, and the cut in H defined by $Z = S \cup \{v_{ik} : i \in S, k < \pi'_i(\bar{S})\}$. Clearly, $i_0 \in Z$ and $\bar{S} \cap D \subseteq \bar{Z}$. By

construction, both v_{ik} and $\pi_i(k)$ are in $\bar{Z}$ for all $i \in S, k \geq \pi'_i(\bar{S})$. Also, for all $k \in N$, $(i, \pi_i(k)) \notin A_H$. From these observations and Figure 2, it follows that the arcs in A_H going from Z to $\bar{Z}$ are $\{(i, v_{ik}) : i \in S, k \geq \pi'_i(\bar{S})\}$. The total capacity of this cut in H equals $\sum_{i \in S} \sum_{k=\pi'_i(\bar{S})}^{n} Y_{a^3_{ik}} = \sum_{i \in S} \sum_{k=\pi'_i(\bar{S})}^{n} y_{(ik)} \geq 1$, because Y admits a flow of one unit over the cut. Hence (9) is satisfied, and y is feasible in $\mathbf{C2^{LP}}$.

To complete the proof, we define the vector Y as $Y_a = 1 \; \forall a \in A^1_H \cup A^2_H$ and $Y_{a^3_{ik}} = y_{(ik)}$ $\forall (i,k) \in V_G \times N$, for any $y \in [0,1]^{|A_G|}$ satisfying (9), and show that Y admits a flow satisfying (10). Suppose this is not the case. Then, there exists a set $Z \subset V_H$, such that $i_0 \in Z$, $\bar{Z} \cap D \neq \emptyset$, and the total capacity of the cut defined by Z is strictly less than one. Obviously, the cut does not contain any arc in $A^1_H \cup A^2_H$, as otherwise the capacity is at least one. We construct a cut in G by setting $S = Z \cap V_G$, which implies $\bar{S} = \bar{Z} \cap V_G$. Observe that if $i \in S$, then $v_{ik} \in \bar{Z} \; \forall k \geq k' = \pi'_i(\bar{S})$, because otherwise the cut in H would contain some arc in A^1_H or A^2_H on the path $(v_{ik}, v_{i,k-1}, \ldots, v_{ik'}, \pi_i(k'))$. Therefore, the total capacity of the cut in H is no less than $\sum_{i \in S} \sum_{k=k'}^{n} Y_{a^3_{ik}} = \sum_{i \in S} \sum_{k=k'}^{n} y_{(ik)} \geq 1$, which gives a contradiction. $\square$

From Propositions 5 and 6, we conclude that **F2**, **S**, and **C2** are equallly strong.

Corollary 7 $\zeta(\mathbf{F2}, \mathcal{I}) = \zeta(\mathbf{S}, \mathcal{I}) = \zeta(\mathbf{C2}, \mathcal{I})$ *for any instance* $\mathcal{I}$ *of MEMP.*

The proof of Proposition 6 has a computational dimension. We can check whether a vector y satisfies (9), without explicitly generating all these constraints, by solving one maximum flow problem in H for every $d \in D$. The arc capacities are $Y_a = 1$ $\forall a \in A^1_H \cup A^2_H$, and $Y_{a^3_{ik}} = y_{(ik)} \; \forall (i,k) \in V_G \times N$. If the maximum flow is (at least) one for all d, then y is feasible in **C2**. Otherwise, it follows from the proof that at least one cut corresponding to a violated constraint of (9) is identified. This observation is useful for solving **C2** and $\mathbf{C2^{LP}}$ by constraint generation.

5.2 Formulations Based on Incremental Power

We can obtain alternative formulations for MEMP by considering the node powers in an incremental fashion. Consider the following set of binary variables:

$$x_{(ik)} = \begin{cases} 1, \text{ if the power of node } i \text{ is at least } c_{(ik)}, \\ 0, \text{ otherwise.} \end{cases}$$

A solution in these variables is feasible only if $x_{(i,k-1)} \geq x_{(ik)} \; \forall i \in V_G, k = 2, \ldots, n$. Assuming $c_{(i0)} = 0 \; \forall i \in V_G$, the total power is given by $\sum_{i \in V_G} \sum_{k=1}^{n} (c_{(ik)} - c_{(i,k-1)}) x_{(ik)}$.

Reformulating **F1**, **F2**, **C1**, and **C2** using incremental node power is straightforward. The relationship between x and y is given by $x_{(ik)} = \sum_{\ell=k}^{n} y_{(i\ell)}$ $i \in V_G$, $k \in N$. Note that the mapping between x and y is unique in both directions.

A transformation based on incremental power can be applied to **S** as well, leading to a second Steiner arborescence model. The graph of this model has the same set of nodes as H, but differs from H in the definition of the arc set.

Reformulations using incremental power neither affect the strength of **F1**, **F2**, **C1**, **C2**, or **S**, nor have a more favorable computational behavior in comparison to the models in the previous sections. We therefore do not discuss them in any detail.

6 NUMERICAL EXPERIMENTS

In order to examine their computational performance, we apply the flow- and cut-based formulations discussed in Sections 3 and 4 to randomly generated instances. Since **S** has more variables and constraints than **F2**, but the same strength, we exclude **S** from the experimental study.

We consider network instances of 10, 20, 50 and 100 nodes, and let D range from a small multicast group to $V_G \setminus \{i_0\}$ (broadcast). We apply the instance generation procedure by Wieselthier et al. [23,24] to generate 100 network instances for various combinations of $|V_G|$ and $|D|$. In the remainder of this section, we let $|V_G|/|D|$ denote the set of instances having $|V_G|$ nodes and $|D|$ destinations. Any reported average value referring to any of these sets is based on all 100 instances. Table entries that cannot be computed because the underlying values are not available for all 100 instances, appear as "–".

A commonly used formula (e.g., [23,24]) for calculating the power requirement c_{ij} is $c_{ij} = \kappa r_{ij}^{\alpha}$, where κ is a constant and r_{ij} is the Euclidean distance between nodes i and j. The parameter α is environment-dependent, and its value typically is between 2 and 4. We set $\alpha = 2$ in all our experiments. This leads to instances harder than those obtained with a larger value of α.

We apply CPLEX 10.1 [14] to the formulations and to their continuous relaxations. We use the dual simplex method as the root algorithm, and let all CPLEX-parameters concerning cut generation be set to their default values. This means that we let CPLEX determine to what extent valid inequalities should be generated in the course of the search. Cut generation may speed up the solution procedure, especially for the weaker formulations. All experiments have been conducted on a 2.4 GHz processor with 2 GB RAM.

The computational experiments are organized into four parts with different objectives as follows:

(1) As mentioned in Section 1, the strength of a model is relevant from a computational standpoint. In the first part of the experiments, we numerically compare the strengths of the flow-based models, that is the one by Das et al. in [8] (henceforth denoted by **DAS**), **F0**, **F1**, and **F2**. By Proposition 3 and Corollary 7, the latter two also indicate the strengths of the cut-based models **C1** and **C2**. Networks of 10 and 20 nodes are studied.
(2) We then compare the formulations in terms of solving MEMP to optimality. Based on the results from the first part, we have chosen to analyze **DAS**, **F1**, **F2**, **C1**, and **C2**. The networks studied consist of 20 and 50 nodes.
(3) The third part of experiments has been conducted for the largest network size (100 nodes) that we have generated. The models compared are **F2** and **C1**, which in part two of the experiments turn out to be more efficient than the other models.
(4) Assessing numerically the performance of heuristics for MEMP requires either optimal solutions or strong lower bounds. As our integer programming models

are useful for this purpose, we apply them in the last part of experiments for performance evaluation of the well-known MIP/BIP heuristic, and compare its numerical performance to its theoretical approximation ratio.

6.1 Strengths of the Models

In Section 3, we gave a sequence of flow-based models, **DAS**, **F0**, **F1**, **F2**, with increasing strength. We apply these formulations to instance sets 10/2, 10/5, 10/9, 20/5, 20/10, and 20/19. The results are presented in Tables 1 and 2, reporting the strength and the computational burden, respectively, of the formulations. Results for $|V_G| = 10$ are not reported in Table 2, since the instances are solved very quickly. In all instance sets, the average CPU time is less than 5 seconds for **F0**, and less than 1 second for the other formulations. The meaning of the columns in the tables is as follows:

- Column "$\zeta = \zeta^*$" in Table 1 shows the number of instances (among a total of 100 instances) for which the optimal solution to the continuous relaxation is integral.
- Column "gap" in Table 1 shows the integrality gap, defined as $\frac{\zeta^*(\mathcal{I})-\zeta(\mathcal{M},\mathcal{I})}{\zeta^*(\mathcal{I})}$ (recall that $\zeta^*(\mathcal{I})$ denotes the optimal total power of instance $\mathcal{I}$).
- Column "LP" in Table 2 is the time (in CPU seconds) spent on solving the LP-relaxation.
- Column "opt" in Table 2 is the time (in CPU seconds) needed to reach and prove optimality.
- Column "b&b" in Table 2 is the number of nodes in the branch and bound tree.

Throughout the section we use [s] to denote that the unit of the column entries is CPU seconds, and [#] to denote that the entries result from counting.

Note that the results in the columns of **F1** (**F2**) in Table 1 also apply to **C1** (**C2**).

TABLE 1
Comparing the strengths of the flow-based formulations.

	DAS		**F0**		**F1**		**F2**	
	$\zeta{=}\zeta^*$[#]	gap	$\zeta{=}\zeta^*$[#]	gap	$\zeta{=}\zeta^*$[#]	gap	$\zeta{=}\zeta^*$[#]	gap
10/2	0	0.70	2	0.46	11	0.24	98	0.00
10/5	0	0.84	0	0.46	8	0.18	95	0.00
10/9	0	0.89	0	0.47	9	0.15	89	0.00
20/5	0	0.89	0	0.63	0	0.33	87	0.00
20/10	0	0.93	0	0.61	0	0.27	75	0.01
20/19	0	0.95	0	0.59	2	0.22	51	0.02

In Table 1, the integrality gap decreases from left to right, and the number of instances solved to optimality increases accordingly. For **DAS**, the gap is above 70% and grows by instance size. The gap of **F0** remains large, although it is considerably lower than that of **DAS**. The results from **F1** and **F2** are better. For **F2**, the gap is close to zero, and there are a large number of instances where the optimum of the continuous relaxation coincides with the integer optimum. To conclude, each step of strengthening in Section 3 gives a significantly stronger relaxation.

Strong formulations are expected to perform well in terms of finding and verifying integer optima. In Table 2, this is indeed observed for the strongest formulation **F2**.

TABLE 2
Comparing the performances of the flow-based formulations.

	DAS			**F0**			**F1**			**F2**		
	LP[s]	opt[s]	b&b[#]	LP[s]	opt[s]	b&b[#]	LP[s]	opt[s]	b&b[#]	LP[s]	opt[s]	b&b[#]
20/5	0.01	2.85	1092.57	3.20	26.82	610.30	0.20	27.96	1930.54	0.11	0.30	0.13
20/10	0.05	13.03	5394.98	3.45	184.70	1572.33	0.51	86.67	2156.29	0.30	0.90	1.92
20/19	0.06	30.36	11899.70	13.33	–	2168.64	1.51	–	2228.02	0.60	3.06	6.01

However, both **F0** and **F1** require more computations than the weakest model **DAS**. Unless the bound is sufficiently tight, the increased computational effort of solving the continuous relaxation may slow down the overall solution process.

Table 2 shows that **F1** results in more branch and bound nodes than **F0**, and when $|D| = 5$, also more than **DAS**. This is explained by the valid inequalities for **DAS** and **F0** generated by CPLEX, which improve the lower bound. The inequalities that CPLEX adds to **F1** have less effect on bounding. Despite its smaller search tree, **F0** is more time consuming than **DAS** and **F1**. This indicates that the explicit representation of node power in combination with multicommodity flow variables makes $\mathbf{F0^{LP}}$ hard to solve. We also observe that **F2** produces a smaller search tree than **F1** does, which is in accordance with our theoretical results.

6.2 Performance in Approaching Optimality

The flow-based formulations **DAS**, **F1** and **F2**, and the cut-based formulations **C1** and **C2**, are further examined with respect to their ability to solve MEMP to optimality. We exclude **F0** due to its poor performance.

Both **C1** and **C2** contain exponentially many constraints. Generating all constraints and solving the resulting models is not a practical approach. Consequently, we have implemented a constraint generation scheme, to be explained below, which gradually extends a set $\mathbf{K}$ of subsets of nodes. Each $S \in \mathbf{K}$ satisfies the condition in (8) and (9). Constraint generation is first applied to the continuous relaxation of either of the cut-based models. When the relaxation is solved, we reintroduce the integrality requirement, and use constraint generation to solve the resulting cut-based model.

Initially, we let $\mathbf{K}$ contain the sets $\{i_0\}$ and $V_G \setminus \{d\}$ for all $d \in D$, and the resulting model is solved to optimality. The constraint generation scheme examines whether the current solution violates (8) or (9) for some $S \subset V_G$ where $i_0 \in S$ and $\bar{S} \cap D \neq \emptyset$. Such sets are found by solving one maximum flow problem for each $d \in D$. If the maximum flow from i_0 to d is smaller than one, then d is not yet connected to i_0, and the values of the dual variables of the flow balance equations are used to detect a violated constraint. The constraint generation scheme thus adds at most $|D|$ constraints per iteration.

The graph in which the maximum flow problems of **C2** are defined differs from that of **C1**. For **C1**, the maximum flow problems are defined on the MEMP graph $G = (V_G, A_G)$. From constraint (8) in **C1**, it can be derived that the capacity of arc $(i, j) \in A_G$ equals $\sum_{k=\pi'_i(j)}^{n} y_{(ik)}$. For **C2**, on the other hand, the capacity of a cut in G cannot be linked to the arc capacity using constraint (9). Instead, as a result of Proposition 6, the maximum flow problems are defined in the expanded graph H. The values of the y-variables define the capacities of the arcs in A_H^3, and all arcs in A_H^1 and A_H^2 have unit capacity. Figure 3 outlines the constraint generation scheme for solving **C1**. To solve **C2**, we replace the constraint on line (7) in Figure 3 with $\sum_{i \in S} \sum_{k=\pi'_i(\bar{S})}^{n} y_{(ik)} \geq 1 \ \forall S \in \mathbf{K}$, and substitute the model given by (11)–(14) in Figure 3 with the model shown in Figure 4.

We apply **DAS**, **F1**, **F2**, **C1**, and **C2** to instance sets 20/5, 20/10, 20/19, 50/5, 50/10, 50/25 and 50/49. For each of the models, we limit the time available for solving the continuous relaxation of an instance to one hour. If the relaxation is solved within

(1) $S \leftarrow \{i_0\}$; $\mathbf{K} \leftarrow \{S\}$
(2) **for all** $d \in D$
(3) $\quad S \leftarrow V_G \setminus \{d\}$; $\mathbf{K} \leftarrow \mathbf{K} \cup \{S\}$;
(4) **repeat**
(5) $\quad \bar{y} \leftarrow$ solution to the cut-based model:
(6) $\quad\quad$ **min** $\sum_{i \in V_G} \sum_{k=1}^{n} c_{(ik)} y_{(ik)}$
(7) $\quad\quad$ **s.t.** $\sum_{i \in S} \sum_{j \in \bar{S}} \sum_{k=\pi'_i(j)}^{n} y_{(ik)} \geq 1 \ \forall S \in \mathbf{K}$
(8) $\quad\quad\quad\quad y \in [0,1]^{|A_G|}$
(9) $\quad$ **totalFlow** $\leftarrow 0$
(10) $\quad$ **for all** $d \in D$
(11) $\quad\quad$ **flow** $\leftarrow \max \ b_d$
(12) $\quad\quad\quad\quad$ **s.t.** $b_i = 0 \ \forall i \in V_G \setminus \{i_0, d\}$
(13) $\quad\quad\quad\quad\quad\quad b \in \Re_+^{|V_G|}$, $f \in \mathcal{F}(G, b)$
(14) $\quad\quad\quad\quad\quad\quad f_{ij} \leq \sum_{k=\pi'_i(j)}^{n} \bar{y}_{(ik)} \ \forall (i,j) \in A_G$
(15) $\quad\quad$ **if** (**flow** < 1)
(16) $\quad\quad\quad S \leftarrow \{i_0\}$;
(17) $\quad\quad\quad$ **for all** $i \in V_G$
(18) $\quad\quad\quad\quad$ **if** (dual variable of flow balance at i = dual variable of flow balance at i_0)
(19) $\quad\quad\quad\quad\quad S \leftarrow S \cup \{i\}$;
(20) $\quad\quad\quad \mathbf{K} \leftarrow \mathbf{K} \cup \{S\}$;
(21) $\quad\quad$ **totalFlow** $\leftarrow$ **totalFlow** + **flow**
(22) **until** (**totalFlow** $= |D|$)
(23) **if** ($\bar{y}$ not integral)
(24) $\quad$ **redo** (4)–(22), where $y \in [0,1]^{|A_G|}$ in line (8) is replaced by $y \in \{0,1\}^{|A_G|}$

FIG. 3. Solving **C1** by constraint generation.

flow $\leftarrow \max \ B_d$
$\quad$ **s.t.** $B_i = 0 \ \forall i \in V_H \setminus \{i_0, d\}$
$\quad\quad B \in \Re_+^{|V_H|}$, $h \in \mathcal{F}(H, B)$
$\quad\quad h_a \leq 1 \ \forall a \in A_H^1 \cup A_H^2$
$\quad\quad h_{a_{ik}^3} \leq \bar{y}_{(ik)} \ \forall (i,k) \in V_G \times N$

FIG. 4. The maximum flow problem in constraint generation for **C2**.

the time limit, a maximum of one (additional) hour is allowed for solving the instance to integer optimality. We summarize the following results in Table 3:

- Column "ζ" shows the number of instances in which the continuous relaxation is solved to optimality within the time limit.
- Column "ζ^*" shows the number of instances in which integer optimality is obtained within the time limit.
- Column "rgap" shows the remaining gap after completion/interruption of the solution procedure, averaged over all instances $\mathcal{I}$. This is defined as $\frac{U(\mathcal{M},\mathcal{I}) - L(\mathcal{M},\mathcal{I})}{U(\mathcal{M},\mathcal{I})}$, where $L(\mathcal{M},\mathcal{I})$ and $U(\mathcal{M},\mathcal{I})$ are the best lower and upper bounds on $\zeta^*(\mathcal{I})$ obtained by use of formulation $\mathcal{M}$.

When the time limits do not apply, the gap is zero. Otherwise, for $\mathcal{M} \in \{\mathbf{DAS}, \mathbf{F1}, \mathbf{F2}\}$,

we get $L(\mathcal{M},\mathcal{I})$ and $U(\mathcal{M},\mathcal{I})$ directly from the branch and bound algorithm in CPLEX. If no feasible solution is found, we apply the default upper bound $U(\mathcal{M},\mathcal{I}) = c_{(i_0,n)}$, and if the LP-relaxation is unsolved, we let $L(\mathcal{M},\mathcal{I}) = 0$ (resulting in an entry in the "rgap" column equal to 1).

TABLE 3
Performance in solving networks of 20 and 50 nodes.

	DAS			**F1**			**F2**			**C1**			**C2**		
	ζ[#]	ζ^*[#]	rgap	ζ[#]	ζ^*[#]	rgap	ζ[#]	ζ^*[#]	rgap	ζ[#]	ζ^*[#]	rgap	ζ[#]	ζ^*[#]	rgap
20/5	100	100	0.00	100	100	0.00	100	100	0.00	97	93	0.07	100	100	0.00
20/10	100	100	0.00	100	100	0.00	100	100	0.00	93	79	0.17	100	100	0.00
20/19	100	100	0.00	100	99	0.01	100	100	0.00	82	60	0.32	100	100	0.00
50/5	100	41	0.26	100	4	0.42	100	100	0.00	97	93	0.07	25	25	0.75
50/10	100	3	0.53	100	0	0.49	100	99	0.00	99	90	0.07	3	3	0.97
50/25	100	0	0.61	100	0	0.51	83	72	0.18	88	70	0.23	0	0	1.00
50/49	100	0	0.58	100	0	0.59	0	0	1.00	66	28	0.57	1	1	0.99

For $\mathcal{M} \in \{\mathbf{C1}, \mathbf{C2}\}$, we set $L(\mathcal{M}, \mathcal{I})$ to the optimal objective function value of the last linear (integer) program (6)-(8) in Fig. 3 ($y \in [0,1]^{|A_G|}$ possibly replaced by $y \in \{0,1\}^{|A_G|}$) that was solved before interruption. However, no upper bound is available in the case of interruption, and therefore the default upper bound is applied in this case.

Consider the results from networks of 20 nodes. By use of **DAS**, **F2**, or **C2**, an integer optimum is obtained in all instances. This holds also for **F1** with one exception. The weaker performance of **C1** compared to **C2** is due to its need for a larger number of iterations and constraints in order to reach optimality.

Solving $\mathbf{DAS^{LP}}$ and $\mathbf{F1^{LP}}$ remains easy in cases where the network has 50 nodes, but from the "rgap" columns it follows that the bounds on $\zeta^*(\mathcal{I})$ are weak. Not many instances can be solved to integer optimality by using **DAS** or **F1**. However, **F2** performs better in this respect. The superiority in strength of **F2** over **F1** is illustrated by the instance set 50/10. When applied to any instance in this set, both $\mathbf{F1^{LP}}$ and $\mathbf{F2^{LP}}$ are solved to optimality. However, whereas **F2** admits integer optimum in 99 instances, the corresponding number of **F1** is zero. The limitation of **F2** is its size. For a large multicast group, in particular broadcast, size prohibits not only **F2** but also $\mathbf{F2^{LP}}$ to be solved.

The performance picture of the two cut-based models applied to networks of 50 nodes differs from that of smaller networks. Solving $\mathbf{C2^{LP}}$ becomes too time-consuming when $|V_G| = 50$. As a result, the exact solution can be computed in only a few instances. Since **C2** is a strong model, however, it admits integer optimum within the time limit in every instance where the relaxation is solved. Compared to **C2**, **C1** performs well when applied to networks of 50 nodes. In all sets of instances of this size, the formulation leads to integer optimum in more instances than does **C2**. It also outperforms its flow-based counterpart **F1** in this respect, although the theoretical analysis proved that these two formulations are equally strong.

6.3 Experiments with Large Networks

From the results presented in Section 6.2, we conclude that **F2** performs well for small $|D|$, and that **C1** is a potential formulation for large $|D|$. We apply these formulations to networks of 100 nodes. The results are presented in Table 4.

The results confirm that **F2** is efficient for the purpose of solving instances with small multicast groups, and that **C1** is better suited for providing a lower bound when $|D| \geq 50$. Although **F2** fails to lead to the integer optimum in one third of the instances in 100/10, the average remaining gap is small (i.e., near-optimal integer solutions have been found). When $|D| \geq 50$, $\mathbf{F2^{LP}}$ is not solved since the solver runs out of memory. A lower bound is available by application of **C1**, but the absence of an upper bound other than the default bound leaves us with a large remaining gap.

6.4 Performance Evaluation of MIP

An important performance metric of a heuristic is its approximation ratio. For many heuristics, this theoretical value is difficult or impossible to obtain. For others, the

TABLE 4
Results for networks of 100 nodes.

	F2			**C1**		
	$\zeta[\#]$	$\zeta^*[\#]$	rgap	$\zeta[\#]$	$\zeta^*[\#]$	rgap
100/5	100	100	0.00	26	14	0.84
100/10	100	66	0.03	33	8	0.86
100/50	0	0	1.00	53	0	0.82
100/99	0	0	1.00	52	0	0.80

approximation ratio can be derived, but it is usually attained by an instance specifically designed to lure the heuristic. Thus the ratio may give a contorted picture of the performance on realistic instances. A complementary approach for performance evaluation is the performance ratio. This approach requires the computation of the optimal solution or a lower bound on the optimum (if the problem involves minimization). For MEMP, our integer programming models can be used for studying the numerical performance of heuristics. As we have observed in the previous section, the models are not applicable in computing integer optimum for large networks. Nevertheless, for these networks the models lead to lower bounds on the optimum, and hence remain useful from a performance evaluation standpoint.

We have chosen to examine the numerical performance of the MIP heuristic, not only because this is the best known heuristic for MEMP, but also because it has been used to benchmark many other heuristics (e.g., [7,9–11,15,19]). Assessing the MIP performance thus gives an indirect evaluation of many heuristics for MEMP.

TABLE 5
Performance evaluation of MIP.

	$\frac{\text{MIP}}{\zeta_{\max}}$		$\frac{\text{MIP}}{\zeta_{\max}}$
10/2	1.0784	20/5	1.1682
10/5	1.1068	20/10	1.2279
10/9	1.1387	20/19	1.2393
50/5	1.2871	100/5	1.3583
50/10	1.2800	100/10	1.3585
50/25	1.2505	100/50	1.6474
50/49	1.3080	100/99	1.5503

For every instance $\mathcal{I}$, we compute the best bound $\zeta_{\max}(\mathcal{I})$ obtained from **F1**, **F2**, **C1** and **C2**. For instances where $\zeta^*(\mathcal{I})$ is computed, we get $\zeta_{\max}(\mathcal{I}) = \zeta^*(\mathcal{I})$, whereas for the other instances we set
$\zeta_{\max}(\mathcal{I}) = \max\{\zeta_{\max}(\mathbf{F1}, \mathcal{I}), \zeta_{\max}(\mathbf{F2}, \mathcal{I}), \zeta_{\max}(\mathbf{C1}, \mathcal{I}), \zeta_{\max}(\mathbf{C2}, \mathcal{I})\}$, where

- $\zeta_{\max}(\mathcal{M}, \mathcal{I})$ is the best lower bound produced by the branch and bound procedure if $\mathcal{M}^{\mathbf{LP}}$ is solved, $\mathcal{M} \in \{\mathbf{F1}, \mathbf{F2}, \mathbf{C1}, \mathbf{C2}\}$,
- $\zeta_{\max}(\mathcal{M}, \mathcal{I})$ is the lower bound produced by the dual simplex algorithm if $\mathcal{M}^{\mathbf{LP}}$ is not solved, $\mathcal{M} \in \{\mathbf{F1}, \mathbf{F2}\}$,
- $\zeta_{\max}(\mathcal{M}, \mathcal{I})$ is the optimal LP-value obtained in the last completed iteration of the constraint generation procedure if $\mathcal{M}^{\mathbf{LP}}$ is not solved, $\mathcal{M} \in \{\mathbf{C1}, \mathbf{C2}\}$.

The results of the performance evaluation are presented in Table 5, which gives the performance ratio of MIP with respect to $\zeta_{\max}(\mathcal{I})$ averaged over all instances. The results show that, for broadcast, MIP's average numerical performance on randomly generated instances is much better than 4.6, a recently proved lower bound on its approximation ratio [4]. For multicast, the performance ratio of MIP is significantly below the theoretical value, which is proportional to $n + 1$ [22].

7 CONCLUSIONS

In this paper, we have studied how mathematical programming can be applied to the minimum-energy multicast problem (MEMP). We have proposed and compared several flow- and cut-based formulations. Starting from a network flow formulation proposed in [8], we have shown how this model can be strengthened successively, resulting in the flow-based formulations **F1** and **F2**. We have also developed the cut-based formulations, **C1** and **C2**, which are exactly as strong as **F1** and **F2**, respectively. The cut-based formulations need fewer variables than the flow-based ones, but contain exponentially many constraints. On the other hand, applying constraint generation to a cut-based formulation turns out to be useful when the size of the strong flow-based model cannot be handled efficiently by a solver.

We have also shown that MEMP can be formulated as a Steiner arborescence problem in an expanded graph. This insight has significance for several reasons. First, it provides us with a link between the strong flow- and cut-based models **F2** and **C2**. Second, previous and future results on the Steiner arborescence problem may be useful in developing solution methods for MEMP. Third, the expanded graph is useful when solving **C2** by constraint generation.

Our numerical experiments illustrate the relation between the strengths of the models and their computational capabilities. Our models can be used to solve small and some medium-sized instances of MEMP to optimality. Heuristics are necessary in order to quickly obtain solutions to large instances. Integer programming formulations remain useful because they provide bounds that are necessary for studying the numerical performance of heuristics. Conclusively, mathematical programming formulations are valuable tools for analyzing and solving MEMP.

There are several interesting directions for further research. One topic is to develop more tailored solution methods that can deliver optimal or near-optimal solutions of MEMP in large networks. As a first step, it is worth studying whether relaxation followed by constraint reinforcement can speed up the solution process of **F2**. A second topic is to utilize mathematical programming models in the study of strong

heuristics and approximation algorithms for MEMP. Moreover, based on the current work, we intend to study energy-optimization problems related to MEMP, for example source-independent minimum-energy broadcast [20]. Other related, and even harder problems, are found in the domain of medium access control. Cross-layer optimization of medium access and broadcast routing lead to topics of forthcoming research.

Acknowledgments

The authors would like to thank the anonymous referees and the associate editor for their valuable comments. We especially appreciate the first referee's thorough review and constructive suggestions.

References

[1] K. Altinkemer, F. S. Salman, and P. Bellur, Solving the minimum energy broadcasting problem in ad hoc wireless networks by integer programming, Proceedings of the 2nd International Network Optimization Conference (INOC), Lisbon, Portugal, 2005, pp. 635–642.

[2] C. Ambühl, An optimal bound for the MST algorithm to compute energy efficient broadcast trees in wireless networks, Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP), Lisbon, Portugal, 2005, pp. 1139–1150.

[3] A. Balakrishnan, T. L. Magnanti, and R. T. Wong, A dual-ascent procedure for large-scale uncapacitated network design, Oper Res 37 (1989), 716–740.

[4] J. Bauer, D. Haugland, and D. Yuan, New results on the broadcast incremental power algorithm (BIP), Paper presented at the First Nordic Optimization Symposium, April 22, Copenhagen, Denmark, 2006.

[5] M. Čagalj, J.-P. Hubaux, and C. Enz, Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues, Proceedings of ACM MobiCom, Atlanta, GA, 2002, pp. 172–182.

[6] J. Cartigny, D. Simplot, and I. Stojmenovic, Localized minimum-energy broadcasting in ad-hoc networks, Proceedings of IEEE INFOCOM, San Francisco, CA, 2003, pp. 2210–2217.

[7] T. Chu and I. Nikoladis, Energy efficient broadcast in mobile ad hoc networks, Proceedings of the 1st International Conference on Ad-Hoc Networks and Wireless (ADHOC-NOW), Toronto, Canada, 2002, pp. 177–190.

[8] A. K. Das, R. J. Marks, M. El-Sharkawi, P. Arabshahi, and A. Gray, Minimum power broadcast trees for wireless networks: Integer programming formulations, Proceedings of IEEE INFOCOM, San Francisco, CA, 2003, pp. 1001–1110.

[9] A. K. Das, R. J. Marks, M. El-Sharkawi, P. Arabshahi, and A. Gray, A cluster-merge algorithm for solving the minimum power broadcast problem in large scale wireless networks, Proceedings of IEEE MILCOM, Boston, MA, 2003, pp. 416–421.

[10] A. K. Das, R. J. Marks, M. El-Sharkawi, P. Arabshahi, and A. Gray, r-Shrink: A heuristic for improving minimum power broadcast trees in wireless networks, Proceedings of IEEE GLOBECOM, San Francisco, CA, 2003, pp. 523–527.

[11] A. K. Das, R. J. Marks, M. El-Sharkawi, P. Arabshahi, and A. Gray, e-Merge: A heuristic for improving minimum power broadcast trees in wireless networks, Technical report, Department of Electrical Engineering, University of Washington, WA, 2003.

[12] Ö. Eğecioğlu and T. F. Gonzalez, Minimum-energy broadcast in simple graphs with limited node power, Proceedings of IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS), Anaheim, CA, 2001, pp. 334–338.

[13] F. K. Hwang, D. S. Richards, and P. Winter, The Steiner tree problem, North-Holland, Amsterdam, The Netherlands, 1992.

[14] Ilog Cplex 10.1, User's manual, ILOG, 2006.

[15] R. Klasing, A. Navarra, A. Papadopoulos, and S. Pérennes, "Adaptive broadcast consumption (ABC), a new heuristic and new bounds for the minimum energy broadcast routing problem", Networking 2004. Lecture notes in Computer Science, N. Mitrou, K. Kontovasilis, G. N. Rouskas, I. Iliadis, and L. Merakos (Editors), Springer-Verlag, Berlin, 2004, vol. 3042, pp. 866–877.

[16] F. Li and I. Nikolaidis, On minimum-energy broadcasting in all-wireless networks, Proceedings of the 26th Annual IEEE Conference on Local Computer Networks, Tampa, FL, 2001, pp. 193–202.

[17] S. Lindsey and C. S. Raghavendra, Energy efficient all-to-all broadcasting for situation awareness in wireless ad hoc networks, J. Parallel Distrib. Comput 63 (2003), 15–21.

[18] R. J. Marks, A. K. Das, M. El-Sharkawi, P. Arabshahi, and A. Gray, Minimum power broadcast trees for wireless networks: Optimizing using the viability lemma, Proceedings of NASA Earth Science Technology Conference, Pasadena, CA, 2002, pp. 273–276.

[19] R. Montemanni, L. M. Gambardella, and A. K. Das, The minimum power broadcast problem in wireless networks: A simulated annealing approach, Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2005), New Orleans, LA, 2005, pp. 2057–2062.

[20] I. Papadimitriou and L. Georgiadis, Minimum-energy broadcasting in multi-hop wireless networks using a single broadcast tree, Mobile Network Appl 11 (2006), 361–375.

[21] P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder, Minimum-energy broadcast routing in static ad hoc wireless networks, Wireless Networks 8 (2002), 607–617.

[22] P.-J. Wan, G. Călinescu, and C.-W. Yi, Minimum-power multicast routing in static ad hoc wireless networks, IEEE/ACM Trans Netw 12 (2004), 507-514.

[23] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, On the construction of energy-efficient broadcast and multicast trees in wireless networks, Proceedings of IEEE INFOCOM, Tel-Aviv, Israel, 2000, pp. 585–594.

[24] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, Algorithms for energy-efficient multicasting in static ad hoc wireless networks, Mobile Network Appl 6 (2001), 251–263.

[25] P. Winter, Steiner problem in networks: A survey, Networks 17 (1987), 129–167.

[26] D. Yuan, An integer programming approach for the minimum-energy broadcast problem in wireless networks, Proceedings of the 2nd International Network Optimization Conference (INOC), Lisbon, Portugal, 2005, pp. 643–650.