9

Models and Algorithms for the MPSCP: An Overview

R. Montemanni,
L.M. Gambardella, and A.K. Das

9.1 Introduction

Ad hoc wireless networks have received significant attention in recent years due to their potential applications in battlefields, emergency disasters relief, and other application scenarios (see, for example, Blough et al.,[2] Chu and Nikolaidis,[4] Clementi et al.,[5] and [7], Kirousis et al.,[10] Lloyd et al.,[11] Ramanathan and Rosales-Hain,[17] Singh et al.,[19] Wan et al.,[20] and Wieselthier et al.[22]). Unlike wired networks of cellular networks, no wired backbone infrastructure is installed in ad hoc wireless networks. A communication session is achieved either through single-hop transmission if the recipient is within the transmission range of the source node, or by relaying through intermediate nodes.

We consider wireless networks where individual nodes are equipped with omnidirectional antennae. Typically, these nodes are also equipped with limited-capacity batteries and have a restricted communication radius. Topology control is one of the most fundamental and critical issues in multihop wireless networks that directly affect network performance. In wireless networks, topology control essentially involves choosing the right set of transmitter power to maintain adequate network connectivity. Incorrectly designed topologies can lead to higher end-to-end delays and reduced throughput in error-prone channels. In energy-constrained networks where replacement or periodic maintenance of node batteries is not feasible, the issue is all the more critical because it directly impacts the network lifetime.

In a seminal paper on topology control using transmission power control in wireless networks, Ramanathan and Rosales-Hain[17] approached the problem from an optimization viewpoint and showed that a network topology that minimizes the maximum transmitter power allocated to any node can be constructed in polynomial time. This is a critical criterion in battlefield applications because using higher transmitter power increases the probability of detection by enemy radar. This chapter attempts to solve the minimum power topology problem in wireless networks. Minimizing the total transmitter power has the effect of limiting the total interference in the network. It has been shown by Clementi et al.[6] that this problem is NP-complete. Related work in the area of minimum power topology construction include Wattenhofer et al.,[21] Huang et al.,[9] and Borbash and Jennings,[3] all of which propose distributed algorithms. Specifically, Wattenhofer et al.[21] propose a cone-based distributed algorithm that relies only on angle-of-arrival estimates to establish a power-efficient connected topology. Huang et al.[9] describe a distributed protocol that is designed for sectored antenna systems. The work in Borbash and Jennings[3] explores the use of relative neighborhood graphs (RNG) for topology control and suggests an algorithm for distributed computation of the RNG.

For a given set of nodes, the *min-power symmetric connectivity problem* (*MPSCP*), sometimes also referred to as the *minimum power topology problem*, is to assign transmission powers to the nodes of the network, which are equipped with omnidirectional antennae, in such a way that all the nodes are connected by bidirectional links and the total power consumption over the network is minimized. Having bidirectional links simplifies one-hop transmission protocols by allowing acknowledgment messages to be sent back for every packet (see Althaus et al.[1]). It is assumed that no power expenditure is involved in reception/processing activities, that a complete knowledge of pairwise distances between nodes is available, and that there is no mobility.

Unlike in wired networks, where a transmission from i to m generally reaches only node m, in wireless networks with omnidirectional antennae it is possible to reach several nodes with a single transmission (this is the so-called *wireless multi-cast advantage*; see Wieselthier et al.[22]). In the example of Figure 9.1, nodes j and k receive the signal originatating from node i and directed to node m because j and k are closer to i than m, that is, they are within the transmission range of a communication from i to m. This property is used to minimize the total transmission power required to connect all the nodes of the network.

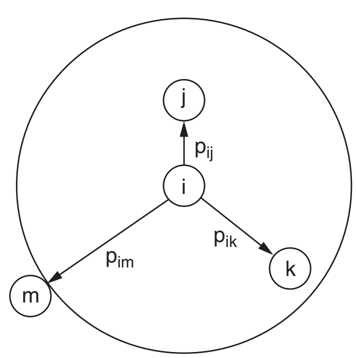

FIGURE 9.1 Communication model.

In Section 9.2 the *MPSCP* is formally described. Section 9.3 is devoted to an overview of the mathematical models and exact algorithms presented thus far in the literature. In Section 9.4 a preprocessing rule, useful to reduce problem dimensions, is described. In Section 9.5 a comparison of the performance of the available exact algorithms is proposed, while Section 9.6 is devoted to conclusions.

9.2 Problem description

To represent the problem in mathematical terms, a model for signal propagation must be selected. We adopt the model presented in Rappaport[18] and used in most of the articles appearing in the literature (see, for example, Wieselthier et al.,[22] Montemanni et al.,[14] and Althaus et al.[1]). According to this model, signal power falls as $\frac{1}{d^\kappa}$, where d is the distance from the transmitter to the receiver and κ is a environment-dependent coefficient, typically between 2 and 4. Under this model, and adopting the usual convention (see, for example, Althaus et al.[1]) that every node has the same transmission efficiency and the same detection sensitivity threshold, the power requirement for supporting a link from node i to node j, separated by a distance d_{ij}, is then given by

$$p_{ij} = (d_{ij})^\kappa \tag{9.1}$$

Using the model described above, power requirements are symmetric; that is, $p_{ij} = p_{ji}$.

Constraints on maximum transmission powers of nodes can be treated by artificially modifying the power requirements. If, for example, node i cannot reach node j even when it is transmitting at its maximum power (i.e., $d_{ij}^\kappa >$ maximum power of node i), then p_{ij} can be redefined as $+\infty$.

MPSCP can be formally described as follows. Given the set V of the nodes of the network, a *range assignment* is a function $r : V \to \mathcal{R}^+$. A *bidirectional link* between nodes i and j is said to be established under the range assignment r if $r(i) \geq p_{ij}$ and $r(j) \geq p_{ij}$. Let now $B(r)$ denote the set of all bidirectional links established under the range assignment r. *MPSCP* is the problem of finding a range assignment r minimizing $\Sigma_{i\in V} r(i)$, subject to the constraint that the graph $(V, B(r))$ is connected.

As suggested by Althaus et al.,[1] a graph theoretical description of *MPSCP* can be given as follows. Let $G = (V, E, p)$ be an edge-weighted complete graph, where V is the set of vertices corresponding to the set of nodes of the network and E is the set of edges containing all the possible pairs $\{i, j\}$, with $i, j \in V$, $i \neq j$. A cost p_{ij} is associated with each edge $\{i, j\}$. It corresponds to the power requirement defined by Equation 9.1.

For a node i and a spanning tree T of G, let $\{i, i_T\}$ be the maximum cost edge incident to i in T; that is, $\{i, i_T\} \in T$ and $p_{ii_T} \geq p_{ij}$ $\forall \{i, j\} \in T$. The *power cost* of a spanning tree T is then $c(T) = \Sigma_{i\in V} p_{ii_T}$. Because a spanning tree is contained in any connected graph, *MPSCP* can be described as the problem of finding the spanning tree T with minimum power cost $c(T)$.

9.3 Mathematical Models and Exact Algorithms

In this section we present four mathematical models for the *MPSCP* that have recently appeared in the literature, all based on mixed-integer programming. For each mathematical formulation discussed, some reinforcing inequalities and an exact algorithm, strongly based on the formulation, are also presented.

9.3.1 Althaus et al.[1]

Althaus et al.[1] have presented a mathematical formulation, with some reinforcing inequalities, and an exact algorithm. They are summarized in this section.

9.3.1.1 Mathematical Formulation

A weighted, directed, complete graph $G' = (V, A, p)$ is derived from G by defining $A = \{(i, j) | i, j \in V\}$, that is, for each edge in E, there are the respective two arcs in A and a dummy arc (i, i) with $p_{ii} = 0$ is inserted for each $i \in V$. p_{ij} is defined by Equation 9.1 when $i \neq j$.

In formulation AL, variables x define the spanning tree T on which the connectivity structure is based. $x_{ij} = 1$ if edge $\{i, j\}$ belongs to the spanning tree T, 0 otherwise. w variables represent the transmission range of nodes. $w_{ij} = 1$ if $i_T = j$ (see Section 9.2), $w_{ij} = 0$ otherwise.

$$(AL)\ \text{Min} \sum_{(i,j)\in A} p_{ij} w_{ij} \tag{9.2}$$

$$\text{s.t.} \sum_{j\in V|(i,j)\in A} w_{ij} = 1 \qquad \forall i \in V \tag{9.3}$$

$$x_{ij} \le \sum_{\substack{(i,k)\in A,\\ p_{ik}\ge p_{ij}}} w_{ik} \qquad \forall \{i, j\} \in E \tag{9.4}$$

$$x_{ij} \le \sum_{\substack{(j,k)\in A,\\ p_{jk}\ge p_{ij}}} w_{jk} \qquad \forall \{i, j\} \in E \tag{9.5}$$

$$\sum_{\{i,j\}\in E} x_{ij} = |V| - 1 \tag{9.6}$$

$$\sum_{\substack{i,j\in S,\\ \{i,j\}\in E}} x_{ij} \le |S| - 1 \qquad \forall S \subset V \tag{9.7}$$

$$x_{ij} \in \{0, 1\} \qquad \forall \{i, j\} \in E \tag{9.8}$$

$$w_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A \tag{9.9}$$

Constraints 9.3 enforce that exactly one range variable for every node $i \in V$ is selected; that is, the range of each node is properly defined. Constraints 9.4 and 9.5 enforce that an edge $\{i, j\}$ is included in the tree only if the range of each endpoint is at least the cost of the edge. Constraints 9.6 and 9.7 enforce that the tree variables indeed form a spanning tree. Constraints 9.8 and 9.9 define the domains of variables.

The bottleneck of formulation AL is represented by constraints 9.7, which are in exponential number and make difficult to handle the formulation in the case of real problems. An idea to overcome this problem is implemented within the exact algorithm described below.

9.3.1.2 Valid Inequalities

Althaus et al.[1] also discuss a set of valid inequalities for formulation *ALT*, which is used within the exact algorithm presented by the authors. These inequalities are summarized in this section.

Definition 9.1 Given $W \subset V$, $\forall i \in W$ we define $i^W \in V\backslash W$ such that $p_{ii^W} \le p_{ij}\ \forall j \in V\backslash W$.

Theorem 9.1 (Crossing inequalities) *The set of inequalities*

$$\sum_{i\in W} \sum_{\substack{(i,j)\in A,\\ p_{ij}\ge p_{ii^W}}} w_{ij} \ge 1 \quad \forall W \subset V \tag{9.10}$$

is valid for formulation AL.

Proof Because T must be a spanning tree, at least one of its edges must cross the cut W. Let $\{i, j\}$ be such an edge, with $i \in W$. Then $p_{ij} \ge p_{ii^W}$ and the range of i at least p_{ij}. Inequality 9.10 must then be satisfied. □

9.3.1.3 Exact Algorithm

Althaus et al.[1] proposed a branch and cut algorithm based on formulation AL. Formulation AL^R_{LR} is considered by the algorithm. It is obtained from AL by substituting constraints 9.8 and 9.9 with their linear relaxation, formally

$$0 \leq x_{ij} \leq 1 \quad \forall \{i, j\} \in E \tag{9.11}$$

$$0 \leq w_{ij} \leq 1 \quad \forall (i, j) \in A \tag{9.12}$$

and by adding constraints 9.10.

Formally, the branch and cut algorithm works by solving formulation AL^R_{LR}. If the solution is integral, the optimal solution has been found, otherwise a variable with a fractional value is picked up and the problem is split into two subproblems by setting the variable to 0 and 1 in the subproblems. The subproblems are solved recursively and disregard a subproblem if the lower bound provided by AL^R_{LR} is worse than the best known solution.

Because there are an exponential number of inequalities of type 9.7, AL^R_{LR} cannot be solved directly at each node of the branching tree. Instead, the algorithm starts with a small subset of these inequalities and algorithmically test whether the solution violates an inequality that is not in the current problem. If so, the inequality is added to it, otherwise the solution of AL^R_{LR} has been retrieved. The separation algorithm described by Padberg and Wolsey[15] has been used for these constraints.

A similar approach applies also for inequalities 9.10, which are again in exponential number. Because there was no known separation algorithm for them, the following heuristic was used. Capacity q_{kl} is defined for each edge $\{k, l\}$:

$$q_{kl} = \sum_{\substack{(k,r)\in A,\\ p_{kr}\geq p_{kl}}} w_{kr} \tag{9.13}$$

An arbitrary node i is chosen and for every node $j \in V\backslash\{i\}$, the minimal directed cut from i to j and from j to i (with capacities defined by Equation 9.13) is computed and it is tested whether or not the corresponding inequality is violated.

9.3.2 Das et al.[8]

Das et al.[8] have presented a mathematical formulation, with some reinforcing inequalities, and an exact algorithm. They are summarized in this section.

9.3.2.1 Mathematical Formulation

The mixed integer programming formulation DAS, described in this section is based on a network flow model (see Magnanti and Wolsey[12]). A node s is elected as the source of the flow, and one unit of flow is sent from s to every other node. The meaning of variables in the formulation is as follows. Variable w_i contains the transmission power of node i. Variable t_{ij} represents the flow on arc (i, j), while u_{ij} is an indicator variable and assumes value 1 if the flow on arc (i, j) is greater than 0 (i.e., $t_{ij} > 0$), 0 otherwise.

$$(DAS) \text{ Min } \sum_{i\in N} w_i \tag{9.14}$$

$$\text{s.t.} \sum_{(i,j)\in A} t_{ij} - \sum_{(k,i)\in A} t_{ki} = \begin{cases} |V| - 1 & \text{if } i = s \\ -1 & \text{otherwise} \end{cases} \quad \forall i \in V \tag{9.15}$$

$$(|V| - 1)u_{ij} \geq t_{ij} \quad \forall (i, j) \in A \tag{9.16}$$

$$w_i \geq p_{ij}u_{ij} \quad \forall (i, j) \in A \tag{9.17}$$

$$u_{ij} = u_{ji} \quad \forall \{i, j\} \in E \tag{9.18}$$

$$u_{ij} \in \{0, 1\} \quad \forall (i, j) \in A \tag{9.19}$$

$$t_{ij} \geq 0 \quad \forall (i, j) \in A \tag{9.20}$$

$$w_i \geq 0 \quad \forall i \in V \tag{9.21}$$

Equations 9.15 define the flow problem on t variables. Constraints 9.16 are the activators for u variables; that is, they connect u and t variables. Inequalities 9.17 connect u variables to w variables, while Equations 9.18 force u variables corresponding to the arcs of a same edge to assume the same value (u must be symmetric because they regulate transmission powers in equalities 9.17). Constraints 9.19, 9.20, and 9.21 define the variables' domains.

The main drawback of formulation *DAS* is represented by constraints 9.16 and 9.17, which tend to push indicator variables u to assume fractional values, making the mixed-integer program very difficult to solve.

9.3.2.2 Valid Inequalities

Das et al.[8] also some valid inequalities. They all rely on the symmetric nature of indicator variables u.

Theorem 9.2 (Connectivity inequalities 1) *The set of inequalities*

$$\sum_{\substack{(i,j)\in A,\\ i\neq j}} u_{ij} \geq 1 \quad \forall i \in V \tag{9.22}$$

is valid for formulation DAS.

Proof To be connected to the rest of the network, each node i must be able to communicate with at least one other node; that is inequality 9.22 must be satisfied. □

Theorem 9.3 (Connectivity inequalities 2) *The set of inequalities*

$$\sum_{\substack{(i,j)\in A,\\ i\neq j}} u_{ji} \geq 1 \quad \forall i \in V \tag{9.23}$$

is valid for formulation DAS.

Proof To be connected to the rest of the network, each node i must receive the signal of at least one other node; that is, inequality 9.23 must be satisfied. □

Theorem 9.4 (Connectivity inequality 3) *The inequality*

$$\sum_{\substack{(i,j)\in A,\\ i\neq j}} u_{ij} \geq 2(|V|-1) \tag{9.24}$$

is valid for formulation DAS.

Proof To have a topology connected by bidirectional links, there must be at least $2(|V|-1)$ active indicator variables (i.e., the number of edges of a spanning tree times 2), as stated by constraint 9.24. □

9.3.2.3 Exact algorithm

Formulation DAS^R is defined to be formulation *DAS* reinforced with the inequalities 9.22, 9.23, and 9.24. The exact algorithm described in Das et al.[8] works by directly solving formulation DAS^R. As observed by Das et al., experimental results suggest that solving DAS^R instead of *DAS* produces shorter computation times.

9.3.3 Montemanni and Gambardella[13] (a)

Montemanni and Gambardella[13] presented a mathematical formulation, with some reinforcing inequalities, and an exact algorithm. They are summarized in this section.

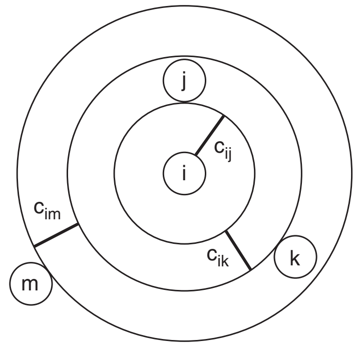

FIGURE 9.2 Incremental mechanism for costs.

9.3.3.1 Mathematical formulation

To describe this mathematical formulation, the following definition is required.

Definition 9.2 Given $(i, j) \in A$, we define the *ancestor* of (i, j) as

$$a_j^i = \begin{cases} i & if p_{ij} = \min_{k \in V}\{p_{ik}\} \\ arg\ \max_{k \in V}\{p_{ik} | p_{ik} < p_{ij}\} & otherwise \end{cases} \tag{9.25}$$

According to this definition, (i, a_j^i) is the arc that originated in node i with the highest cost such that $p_{ia_j^i} < p_{ij}$*. In case an *ancestor* does not exist for arc (i, j), vertex i is returned, that is, the dummy arc (i, i) is addressed.

In the example of Figure 9.1, arc (i, k) is the ancestor of arc (i, m); (i, j) is the ancestor of (i, k); and the dummy arc (i, i) is returned as the ancestor of (i, j).

The formulation is based on an incremental mechanism over the variables representing transmission powers. The costs associated with these variables in the objective function 9.27 will be given by the following formula:

$$c_{ij} = p_{ij} - p_{ia_j^i} \quad \forall (i, j) \in A \tag{9.26}$$

where c_{ij} is equal to the power required to establish a transmission from nodes i to node j (p_{ij}) minus the power required by nodes i to reach node a_j^i $(p_{ia_j^i})$. In Figure 9.2, the costs arising from the example of Figure 9.1 are depicted.

It is important to observe that the incremental mechanism is the most important element of the formulation. It will allow us to define very strong reinforcing inequalities, which are the basis for the good performance of the exact algorithm based on the formulation (see Section 9.5).

The mixed-integer programming formulation MGa described in this section is based on a network flow model (see Magnanti and Wolsey[12]). A node s is elected as the source of the flow, and one unit of flow is sent from s to every other node. Variable x_{ij} represents the flow on arc (i, j). Variable y_{ij} is 1 when

*For the sake of simplicity, we have considered the (usual) case where $\forall i \in V \nexists k, l \in V$ s.t. $p_{ik} = p_{il}$. In case this is not true, the following formula, which breaks ties, has to be used in place of 9.25:

$$a_j^i = \begin{cases} i & if p_{ij} = \min_{k \in V}\{p_{ik}\} \\ arg\ \max_{k \in V}\left\{p_{ik} \middle| \left(\begin{array}{c} (p_{ik} < p_{ij} \wedge (\nexists l \in V \text{ s.t. } p_{ik} = p_{il} \wedge l > k)) \\ \vee\, (p_{ij} = p_{ij} \wedge (\nexists l \in V \text{ s.t. } p_{ik} = p_{il} \wedge j > l > k)) \end{array} \right)\right\} & otherwise \end{cases}$$

node i has a transmission power that allows it to reach node j, $y_{ij} = 0$ otherwise.

$$(MGa)\ \text{Min} \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{9.27}$$

$$\text{s.t. } y_{ij} \le y_{ia^i_j} \qquad \forall (i,j) \in A, a^i_j \ne i \tag{9.28}$$

$$x_{ij} \le (|V|-1) y_{ij} \qquad \forall (i,j) \in A \tag{9.29}$$

$$x_{ij} \le (|V|-1) y_{ji} \qquad \forall (i,j) \in A \tag{9.30}$$

$$\sum_{(i,j)\in A} x_{ij} - \sum_{(k,i)\in A} x_{ki} = \begin{cases} |V|-1 & \text{if } i = s \\ -1 & \text{otherwise} \end{cases} \qquad \forall i \in V \tag{9.31}$$

$$x_{ij} \in \mathcal{R} \qquad \forall (i,j) \in A \tag{9.32}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{9.33}$$

Constraints 9.28 realize the incremental mechanism by forcing the variables associated with arc (i, a^i_j) to assume value 1 when the variable associated with arc (i, j) has value 1; that is, the arcs that originated in the same node are activated in increasing order of p. Inequalities 9.29 and 9.30 connect the flow variables x to y variables. Equations 9.31 define the flow problem, while 9.32 and 9.33 are domain definition constraints. We refer the interested reader to Magnanti and Wolsey[12] for a more detailed description of the spanning tree formulation behind the formulation presented above.

In formulation MGa, the bottleneck is represented by constraints 9.29 and 9.30, which tend to push y variables to be fractional. Fortunately, the incremental mechanism on which the mixed-integer program is based allows us to define strong reinforcing inequalities that help overcome this problem (see below).

9.3.3.2 Valid Inequalities

The valid inequalities presented in this section were proposed in Montemanni and Gambardella[13] to reinforce mathematical formulation MGa.

In the remainder of this section we refer to the subgraph of G' defined by the y variables with value 1 as G_y. Formally, $G_y = (V, A_y)$, where $A_y = \{(i,j) \in A | y_{ij} = 1 \text{ in the solution of } MPSC\}$.

Theorem 9.5 (Connectivity inequalities) *The set of inequalities*

$$y_{ij} = 1 \ \ \forall (i,j) \in A \text{ s.t. } a^i_j = i \tag{9.34}$$

is valid for formulation MGa.

Proof To have the graph G_y connected, each node must be able to communicate with at least one other node. Then its transmission power must be sufficient to reach at least the node which is closest to it; that is, $y_{ia^i_j} = 1$. □

Theorem 9.6 (Bidirectional inequalities 1) *The set of inequalities*

$$y_{a^i_j i} \ge y_{ia^i_j} - y_{ij} \ \ \forall (i,j) \in A \text{ s.t. } a^i_j \ne i \tag{9.35}$$

is valid for formulation MGa.

Proof If $y_{ij} = 1$, then $y_{ia^i_j} = 1$ because of inequalities 9.28 and consequently in this case the constraint does not give any new contribution.

If $y_{ij} = 0$ and $y_{ia^i_j} = 0$, then again the constraint does not give any new contribution.

If $y_{ij} = 0$ and $y_{ia^i_j} = 1$, then the transmission power of node i is set to reach node a^i_j and nothing more. The only reason for node i to reach node a^i_j and nothing more is the existence of a bidirectional link on edge $\{i, a^i_j\}$ in G_y. Consequently, $y_{a^i_j i}$ must be equal to 1, as stated by the constraint. □

Theorem 9.7 (Bidirectional inequalities 2) *The set of inequalities*

$$y_{ji} \geq y_{ij} \ \forall (i,j) \in A \ s.t. \ \nexists (i,k) \in A, a_k^i = j \tag{9.36}$$

is valid for formulation MGa.

Proof If $y_{ij} = 0$, the constraint does not give any new contribution.

If $y_{ij} = 1$, then the transmission power of node i is set in such a way to reach node j, which is the farthest node from i in G. The only reason for node i to reach node j is the existence of a bidirectional link on edge $\{i, j\}$ in G_y. Consequently, y_{ji} must be equal to 1, as stated by the constraint. □

Theorem 9.8 (Tree inequality) *The inequality*

$$\sum_{(i,j)\in A} y_{ij} \geq 2(|V| - 1) \tag{9.37}$$

is valid for formulation MGa.

Proof To be strongly connected, the directed graph G_y must have at least $2(|V| - 1)$ arcs, as stated by constraint 9.37. □

Definition 9.3 $G_a = (V, A_a)$ is the subgraph of the complete graph G' such that $A_a = \{(i, j) | \, a_j^i = i\}$.

Notice that $|A_a| = |V|$ by definition.

Definition 9.4 $\mathcal{R}_i = \{j \in V| \ j$ can be reached from i in $G_a\}$.

Theorem 9.9 (Reachability inequalities 1) *The set of inequalities*

$$\sum_{\substack{(k,l)\in A \\ s.t. k\in\mathcal{R}_i, l\in V\setminus\mathcal{R}_i}} y_{kl} \geq 1 \ \ \forall i \in V \tag{9.38}$$

is valid for formulation MGa.

Proof Because graph G_y must be strongly connected, it must be possible to reach every node j starting from each node i. This implies that at least one arc must exist between the nodes which is possible to reach from i in G_a (i.e., $\mathcal{R}_i$) and the other nodes of the graph (i.e., $V\setminus\mathcal{R}_i$). □

Definition 9.5 $\mathcal{Q}_i = \{j \in V| \ i$ can be reached from j in $G_a\}$.

Theorem 9.10 (Reachability inequalities 2) *The set of inequalities*

$$\sum_{\substack{(k,l)\in A \\ s.t. \ k\in\mathcal{Q}_i, l\in V\setminus\mathcal{Q}_i}} y_{kl} \geq 1 \ \ \forall i \in V \tag{9.39}$$

is valid for formulation MGa.

Proof Because graph G_y must be strongly connected, it must be possible to reach every node i from every other node j of the graph. This means that at least one arc must exist between the nodes that cannot reach i in G_a (i.e., $V\setminus\mathcal{Q}_i$) and the other nodes of the graph (i.e., $\mathcal{Q}_i$). □

9.3.3.3 Exact Algorithm

Formulation MGa^R is defined as the formulation MGa reinforced with the inequalities 9.34, through 9.40. The exact algorithm described by Montemanni and Gambardella[13] works by directly solving formulation MGa^R. The authors showed that solving MGa^R instead of MGa produces computation times that are shorter up to a factor of 1920 for some problems.

9.3.4 Montemanni and Gambardella[13] (b)

Montemanni and Gambardella[13] also presented a second mathematical formulation and a second exact algorithm. They are summarized in this section, together with a new valid inequality.

9.3.4.1 Mathematical Formulation

The mathematical model described in this section is based on the same incremental mechanism discussed in Section 9.3.3. In formulation MGb, a spanning tree is defined by z variables. Variable z_{ij} is 1 if edge $\{i, j\}$ is on the spanning tree, $z_{ij} = 0$ otherwise. Variable y_{ij} is 1 when node i has a transmission power, which allows it to reach node j, $y_{ij} = 0$ otherwise.

$$(MGb) \text{ Min } \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{9.40}$$

$$\text{s.t. } y_{ij} \leq y_{ia^i_j} \quad \forall (i, j) \in A, a^i_j \neq i \tag{9.41}$$

$$z_{ij} \leq y_{ij} \quad \forall \{i, j\} \in E \tag{9.42}$$

$$z_{ij} \leq y_{ji} \quad \forall \{i, j\} \in E \tag{9.43}$$

$$\sum_{\substack{\{i,j\}\in E,\\ i\in S, j\in V\setminus S}} z_{ij} \geq 1 \quad \forall S \subset V \tag{9.44}$$

$$z_{ij} \in \{0, 1\} \quad \forall \{i, j\} \in E \tag{9.45}$$

$$y_{ij} \in \{0, 1\} \quad \forall (i, j) \in A \tag{9.46}$$

Constraints 9.41 realize the incremental mechanism by forcing the variables associated with arc (i, a^i_j) to assume value 1 when the variable associated with arc (i, j) has value 1; that is, the arcs originated in the same node are activated in increasing order of p. Inequalities 9.42 and 9.43 connect the spanning tree variables z to y variables. Equations 9.44 state that all the vertices must be mutually connected in the subgraph induced by z variables, while 9.45 and 9.46 are domain definition constraints.

The main problem of formulation MGb is related to the exponential number of constraints (9.44). In the exact algorithm described below, a technique to overcome this bottleneck is implemented.

9.3.4.2 Valid Inequalities

Because all the inequalities presented in Section 9.3.3 use y variables only, and because the role of y variables is the same in both formulations MGa and MGb (i.e., y variables implement the incremental mechanism described in Section 9.3.3), the results presented in Section 9.3.3 for formulation MGa are valid also for formulation MGb.

In addition, the following inequality is also considered. It will be used within the exact algorithm described in the next section.

Theorem 9.11 (z tree inequalities)

$$\sum_{\{i,j\}\in E} z_{ij} \geq |V| - 1 \tag{9.47}$$

Proof Inequality 9.47 forces the number of active z variables to be at least $|V| - 1$. This condition is necessary in order to have a spanning tree. □

9.3.4.3 Exact Algorithm

The integer program MGb^R is defined as MGb without constraints 9.44 but with the inequalities 9.34, 9.35, 9.36, 9.37, 9.38, 9.39 and 9.47. Notice that constraint 9.47 forces the active z variables to be at least $|V| - 1$ already during the very first iterations of the method we are describing in this section. This will contribute to speed up the algorithm.

The idea at the basis of the method is that it is very difficult to deal directly with constraints 9.44 of formulation MGb_R in case of large problems. For this reason, some techniques that leave some of them out must be considered. In this section we present an iterative approach that in the beginning does not consider any constraint 9.44, and adds them step by step in case they are violated. Formulation MGb is solved and the values of the z variables in the solution are examined. If the edges corresponding to variables with value 1 form a spanning tree, then the problem has been solved to optimality; otherwise, constraints 9.48, described below, are added to the integer program and the process is repeated.

At the end of each iteration, if edges corresponding to z variables with value 1 in the last solution generate a set $\mathcal{CC}$ of connected components, with $|\mathcal{CC}| > 1$, then the following inequalities are added to the formulation:

$$\sum_{\substack{\{i,j\}\in E,\\ i\in C, j\in V\setminus C}} z_{ij} \geq 1 \quad \forall C \in \mathcal{CC} \tag{9.48}$$

Inequalities 9.48 force z variables to connect the (elsewhere disjoint) connected components of $\mathcal{CC}$.

9.4 Preprocessing Procedure

The results described in this section are used to delete some arcs of graph G' and consequently to speed up the exact algorithms previously presented. They were originally presented by Montemanni and Gambardella.[13]

We suppose heuristic solution for the problem, *heu*, is available, and its cost is *cost*(*heu*). All the variables that, if active, would induce a cost higher than *cost*(*heu*) can be deleted from the problem.

Theorem 9.12 *If the following inequality holds*

$$p_{ij} + p_{ji} + \sum_{\substack{k\in V\setminus\{i,j\},\\ a_l^k = k}} p_{kl} > cost(heu) \tag{9.49}$$

then arc (i, j) *can be deleted from* A.

Proof Using the same intuition at the basis of the proofs of Theorems 9.6 and 9.7, we have that if p_{ij} is the power of node i in a solution, this means that the power of node j must be greater than or equal to p_{ji} (i.e., arc (j, i) must be in the solution), because otherwise there would be no reason for node i to reach node j. The left-hand side of inequality 9.49 represents then a lower bound for the total power required to maintain the network connected in case node i transmits to a power that allows it to reach node j and nothing farther. For this reason, if inequality 9.49 holds, arc (i, j) can be deleted from A. □

It is important to notice that once arc (i, j) is deleted from A, the value of the ancestor of node k (see Section 9.3.3), with $a_k^i = j$, must be updated to a_j^i.

9.5 Computational Results

Computational tests have been carried out on two different families of problems, randomly generated as described by Althaus et al.[1] and Das et al.,[8] respectively. In Althaus et al.,[1] $\kappa = 4$ and a problem with

TABLE 9.1 Average Computation Times (sec) on the Problems Described by Althaus et al.[1]

	$\|V\|$						
Algorithms	10	15	20	25	30	35	40
AL	2.144	18.176	71.040	188.480	643.200	2278.400	15120.000
MGa	0.192	0.736	8.576	33.152	221.408	1246.304	9886.080
Preprocessing $+MGa$	0.078	0.289	0.715	4.924	28.908	87.357	583.541
Preprocessing $+MGb$	0.052	0.196	0.601	2.181	13.481	28.172	79.544

$|V|$ nodes is obtained by choosing $|V|$ points uniformly at random from a grid of size 10000×10000. For the problems described in Das et al.,[8] the procedure is the same but the grid has dimension 5×5. In addition, for these last problems, a maximum transmission power, depending on the number of nodes of the network, is fixed. The following pairs (*number of nodes, maximum transmission power*) have been adopted: $(15, 3.00)$, $(20, 3.00)$, $(30, 2.50)$, $(40, 1.50)$, $(50, 0.75)$. ILOG CPLEX* 6.0 has been used to solve integer programs.

In the remainder of this section we refer to the algorithm presented by Althaus et al.[1] (see Section 9.3.1) as *AL*, to the one described in Das et al.[8] (see Section 9.3.2) as *DAS*, and to those proposed by Montemanni and Gambardella[13] (see Sections 9.3.3 and 9.3.4) as *MGa* and *MGb*, respectively.

In Table 9.1 we present the average computation times required (on a SUNW Ultra-30 machine) by some of the exact algorithms on the problems described by Althaus et al.[1] for different values of V. Fifty instances are considered for each value of $|V|$.

Table 9.1 shows that the MGa and MGb outperform AL. MGb also performs clearly better than MGa. In Table 9.1, the benefit derived from the use of the preprocessing technique described in Section 9.4 is highlighted. To apply this preprocessing procedure, a heuristic solution to the problem must be available. For this purpose we use one of the simplest algorithms available, which works by calculating the *Minimum Spanning Tree* (see Prim[16]) on the weighted graph with costs defined by Equation 9.1, and by assigning the power of each transmitter i to p_{ii_T}, as described near the end of Section 9.2. The computational times of the algorithm MGa are improved up to 17 times (for $|V| = 40$) when this technique is used (on average, 79 percent of the arcs were deleted for $|V| = 40$; see Montemanni and Gambardella[13]).

In Table 9.2 we present the average computational times required (on a Pentium 4 1.5-GHz machine) by some of the exact algorithms on the problems described by Das et al.[8] for different values of V. In brackets we also report the average standard deviation on solving times. Twenty five instances are considered for each value of $|V|$. Some entries are marked with '—'; this means that the corresponding algorithms failed to solve some of the corresponding instances in less than 3600 seconds.

Table 9.2 suggests again that MGa and MGb obtain the best performance. For these problems, the algorithms highlight that all the algorithms are not extremely robust (see large standard deviation on solution times); that is, there are very different performances on instances of the same family. This could

TABLE 9.2 Average Computation Times (sec) on the Problems Described by Das et al.[8]

	$\|V\|$				
Algorithms	15	20	30	40	50
DAS	0.014 (0.018)	7.511 (36.697)	—	—	—
MGa	**0.008** (0.006)	**0.027** (0.013)	1.518 (4.401)	24.723 (111.378)	**12.233** (18.025)
MGb	0.019 (0.010)	0.058 (0.038)	**0.795** (1.093)	**9.906** (20.312)	47.756 (136.234)

*http://www.cplex.com.

depend on the small grid adopted, which tends to flatten power requirements, and this causes many almost equivalent solutions. On the other hand, average computational times are much shorter than those reported in Table 9.1, and this depends on the maximum transmission power constraints, that substantially contribute to reduce the number of variables of the problems.

9.6 Conclusion

We have presented an overview of the mathematical formulations presented so far for the min-power symmetric connectivity problem in wireless networks. Some exact algorithms, strongly based on these formulations and on some reinforcing inequalities developed for them, have been discussed, together giving a preprocessing rule.

Computational results have been presented, aiming to compare the performance of the different exact approaches discussed in this chapter.

Acknowledgments

Two of the authors (Montemanni and Gambardella) were partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923) and by the Swiss National Science Foundation through Project 200021-100539.

References

1. E. Althaus, G. Călinescu, I.I. Măndoiu, S. Prasad, N. Tchervenski, and A. Zelikovsky. Power efficient range assignment in ad-hoc wireless networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'03)*, pages 1889–1894, 2003.
2. D. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. In *Proceedings of the 2nd IFIP International Conference on Theoretical Computer Science (TCS 2002)*, pages 71–82, 2002.
3. S.A. Borbash and E.H. Jennings. Distributed topology control algorithm for multihop wireless networks. In *Proceedings of IJCNN*, 2002.
4. T. Chu and I. Nikolaidis. Energy efficient broadcast in modile ad hoc networks. In *Proceedings of AD-HOC NetwOrks and Wireless*, 2002.
5. A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy consumption broadcast subgraphs. In *Symposium on Theoretical Aspects of Computer Science*, pages 121–131, 2001.
6. A. Clementi, P. Penna, and R. Silvestri. Hardness results for the power range assignment problem in packet radio networks. *Lectures Notes on Computer Science*, 1671:195–208, 1999.
7. A. Clementi, P. Penna, and R. Silvestri. On the power assignment problem in radio networks. Technical Report TR00-054, Electronic Colloquium on Computational Complexity (ECCC), 2000.
8. A.K. Das, R.J. Marks, M. El-Sharkawi, P. Arabshani, and A. Gray. Optimization methods for minimum power bidirectional topology construction in wireless networks with sectored antennas. Submitted for publication.
9. Z. Huang, C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo. Topology control for ad hoc networks with directional antennas. In *Proceedings of the Eleventh International Conference on Computer Communications and Networks*, 2002.
10. L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243:289–305, 2000.
11. E. Lloyd, R. Liu, M. Marathe, R. Ramanathan, and S. Ravi. Algorithmic aspects of topology control problems for ad hoc networks. In *Proceedings of the ACS MobiHoc*, pages 123–134, 2002.
12. T.L. Magnanti and L. Wolsey. Optimal trees. In *Network Models, Handbook in Operations Research and Management Science* (M.O. Ball et al., Eds.), Vol. 7, pages 503–615. North-Holland, 1995.

13. R. Montemanni and L.M. Gambardella. Exact algorithms for the minimum power symmetric connectivity problem in wireless networks. *Computers and Operations Research*, to appear.
14. R. Montemanni, L.M. Gambardella, and A.K. Das. The minimum power broadcast tree problem in wireless networks: a simulated annealing approach. Submitted for publication.
15. M. Padberg and L. Wolsey. Trees and cuts. *Annals of Discrete Mathematics*, 17:511–517, 1983.
16. R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36:1389–1401, 1957.
17. R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of the IEEE Infocom*, pages 404–413, 2000.
18. T. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.
19. S. Singh, C. Raghavendra, and J. Stepanek. Power-aware broadcasting in mobile ad hoc networks. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'99)*, 1999.
20. P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder. Minimum energy broadcast routing in static ad hoc wireless networks. In *Proceedings of the IEEE Infocom*, pages 1162–1171, 2001.
21. R. Wattenhofer, L. Li, P. Bahl, and Y.M. Wang. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In *Proceedings of the Infocom*, 2001.
22. J. Wieselthier, G. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *Proceedings of the IEEE INFOCOM 2000 Conference*, pages 585–594, 2000.