Optimizing the Design of a Wind Farm Collection Network

Alain Hertz
GERAD and Département de mathématiques et génie industriel
École Polytechnique de Montréal
alain.hertz@gerad.ca

Odile Marcotte
GERAD and Département d'informatique
Université du Québec à Montréal
odile.marcotte@gerad.ca

Asma Mdimagh
GERAD and Département de mathématiques et génie industriel
École Polytechnique de Montréal
asma.mdimagh@gerad.ca

Michel Carreau
Hatch, Montréal, Québec, Canada
mcarreau@hatch.ca

François Welt
Hatch, Niagara Falls, Ontario, Canada
fwelt@hatch.ca

Abstract

In this article we study a network design problem that arises in the exploitation of wind energy. We formulate this problem as a mixed integer programming problem, relate this formulation to other problems in combinatorial optimization, strengthen the formulation and propose cutting planes, and finally present experimental results for real-world instances of the collection network design problem.

Key Words: Network design, wind farm, mixed integer programming, cutting planes.

1 Introduction

The design of networks plays an important role in the planning of large systems, especially in the fields of telecommunications, transportation, and energy. In particular it has become necessary to design networks for collecting the energy produced through eco-friendly means, such as wind turbines or solar panels. In this article we study a problem proposed by a Canadian company involved in the design of wind farm collection networks. In Section 2 we give a precise definition of this problem and in Section 3 a formulation of our problem as a mixed integer program. In Section 4 we survey the literature on problems that are similar to ours. In Section 5 we discuss ways of strengthening our model and introduce classes of cutting planes. Finally we present experimental results in Section 6 and our conclusions in Section 7.

2 The collection network design problem

A wind farm is a group of wind turbines that come into operation at approximately the same time and whose energy must be collected and distributed through an existing electrical network. In this article we assume that the locations of the turbines are already known and focus on the problem of collecting the energy produced by the turbines and sending it to a known sub-station. The network to be designed consists of:

- underground (UG) *cables*, each of which links two turbines or a turbine to the above-ground network,
- above-ground transmission *lines* that usually follow existing roads and link two geographical points (i.e., road intersections), and
- *disconnects* between the endpoints of UG cables and the above-ground network.

The transmission lines can be built only on roads for which municipal permits (called Right-of-Way) have been obtained. We assume that the road segments where the lines may be built, and the associated building costs, are known.

Each type of link (UG cable or transmission line) has a limited capacity. Also several parallel links may be installed between two endpoints; for instance there may be several cables between two turbines (say, u and v) and on each cable the energy may flow from u to v or v to u. In this article, for the sake of simplicity, we assume that there is only one type of UG cable and one type of transmission line. The capacity of each type of link (cable or line) is known. The model presented in the next section could be easily modified if this assumption were relaxed. We also assume that we know the cost of installing a given link (i.e., a UG cable or transmission line) between two given endpoints; this cost includes the cost of the disconnect when the UG cable links a turbine to the above-ground network. In general, if there are parallel links between two nodes or vertices, the cost of installing the first one is greater than that of installing the second one, the second is more costly than the third, and so on. Finally we assume that there is one unit of energy produced by each turbine; this assumption could be relaxed as well.

Figure 1 displays the graph underlying a wind farm collection design problem. The turbines are represented as black nodes and the sub-station as a black square; the other nodes represent the endpoints of transmission lines. The potential UG cables are precisely those edges that have at least one turbine (i.e., a black node) as an endpoint. The transmission lines are those edges that link two white nodes. Therefore in this example, a node in the underground network is always a turbine; in other examples the underground network may contain intermediate nodes as well as turbines.

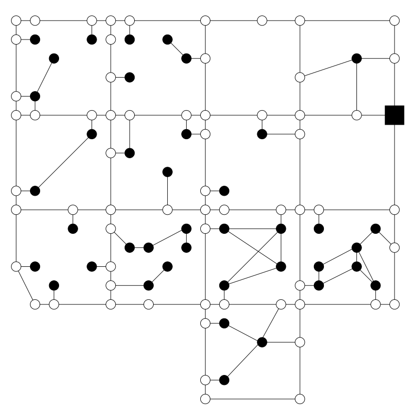

Figure 1: The graph underlying a wind farm collection design problem

We now discuss an important engineering constraint. The energy flowing through a link (e.g., a cable) is *unsplittable*, i.e., if a "chunk" Q of energy flows from point u to point v through a cable or transmission line, then there must be a point w such that this chunk flows from v to w through a given cable or transmission line. Of course the flow from v to w may be greater than Q since v may receive flow from more than one vertex (and in particular from a vertex u' different from u). Hence a solution with no parallel links may look like an anti-rooted tree (with the sub-station as anti-root). An anti-rooted tree is a directed tree in which every arc is directed towards a distinguished vertex called the anti-root. On the other hand there are solutions that do not look like anti-rooted trees, for instance those containing parallel links or a substructure consisting of arcs (u_1, v), (u_2, v), (v, w_1), and (v, w_2) (in which case the solution includes a cycle going through v and the sub-station).

The paths along which flows the energy produced by the wind farm constitute a *circuit* (in the terminology of electrical engineering), hence the use of the phrase "overhead circuit" to describe the transmission lines that are part of the network. The engineering practice of designing such circuits justifies the modelling of our problem as an unsplittable flow problem (see Section 3). Figure 2 displays a feasible solution of the design problem described in Figure 1. The number on any given edge is the number of links between its two endpoints. We have omitted the label when there is only one link between the endpoints. The *collection network design problem* consists of finding a network of minimum cost through which the energy produced by the turbines will be sent to the sub-station.

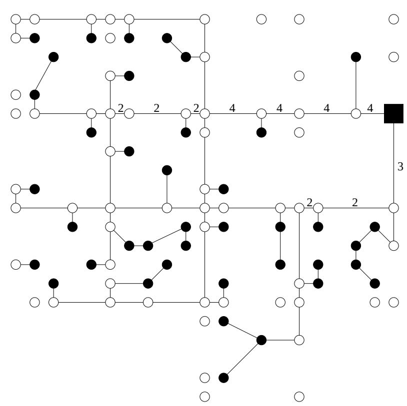

Figure 2: A feasible solution for the problem depicted in Figure 1

In order to model the network design problem, we must describe formally the graph $G = (V, A)$ underlying the problem. This graph is directed since the energy may flow in either direction on a given link. The node set V is the disjoint union of T (the set of turbines), R (the set of endpoints of transmission lines), and $\{0, s\}$ (where s denotes the sub-station and 0 a *source node*). Similarly the set A is the disjoint union of the following five categories of arcs:

1. all the arcs of the form $(0, u)$ for $u \in T$;
2. all the arcs of the form (u, v) and (v, u) for each UG cable with endpoints u and v in T;
3. all the arcs of the form (u, v), where u belongs to T and v to R;
4. all the arcs of the form (u, v) and (v, u) for each transmission line with endpoints u and v in R; and
5. all the arcs of the form (v, s), where v belongs to R and there is a transmission line between v and s.

In the sequel A_1 will denote the set of arcs (u, v) such that (v, u) does not belong to A, and E the set of pairs $\{u, v\}$ such that (u, v) and (v, u) both belong to A.

We let m denote the maximal number of parallel links between vertices u and v. For instance, there may be up to m UG cables between turbines u and v. Then the links are denoted by $(u, v, 1)$, $(v, u, 1)$, $(u, v, 2)$, $(v, u, 2)$, etc., but only one of (u, v, k) and (v, u, k) is selected for any k (this amounts to choosing the flow direction on the kth link between u and v). The value of m is 4 in the problem instances supplied by the industrial partner. Because of our assumptions the capacities of the arcs are easily described. The capacity of an arc in Category 1 equals 1 since exactly one unit of energy is produced by each turbine. The capacity of an arc in Category 2 or 3 equals C_{ug}, where C_{ug} is a constant. The capacity of an arc in Category 4 or 5 equals C_{ag}, where C_{ag} is another constant. The arc costs cannot be described so easily, and one must distinguish between links. More precisely, the cost of each additional link between two vertices u and v is at most the cost of the previous link between u and v. Thus if c_{uv}^k denotes the cost of the link (u, v, k), the relation $c_{uv}^1 \geq c_{uv}^2 \geq \cdots \geq c_{uv}^m$ holds for every arc (u, v) (note that $c_{uv}^k = c_{vu}^k$ holds for any k). The cost c_{0u}^k equals 0 for any k and any u in T since the source node is "fictitious".

The goal of the model is to minimize the total cost of the cables used to collect the energy produced by the wind turbines and send it to the sub-station. The company will derive an obvious economic benefit from minimizing the total cost of the cables. It is also reasonable to expect that an optimal solution will provide a simple but robust design for the collection network.

3 The model

As explained in the previous section, the problem consists of choosing the cables and lines in order to transport the energy produced from the wind turbines to the sub-station. Hence the most important decision variables are the t_{uv}^k, defined as follows: t_{uv}^k equals 1 if the kth copy of arc (u,v) belongs to the network and 0 otherwise. We also need variables to represent the energy flowing through a given arc: x_{uv}^k denotes the flow through the kth copy of arc (u,v). As explained above, the flow is unsplittable, and we must express this constraint in mathematical terms. We have chosen to introduce the binary variables $y_{uvw}^{kk'}$, defined as follows: $y_{uvw}^{kk'}$ equals 1 if and only if all the energy flowing through the kth copy of arc (u,v) also flows through the (k')th copy of arc (v,w). Finally $z_{uvw}^{kk'}$ denotes the amount of energy flowing through the kth copy of arc (u,v) and then through the (k')th copy of arc (v,w); recall that all the energy flowing through the kth copy of arc (u,v) must be channelled through a single arc originating at v (unless v equals s).

The goal of the model is to minimize the total cost of the cables and lines, i.e., $\sum_{(u,v)\in A}\sum_{k=1}^{m} c_{uv}^k t_{uv}^k$, where c_{uv}^k denotes the cost of the kth copy of arc (u,v). The constraints (2) express the fact that flow must be conserved, except at the source node 0 (from which $|T|$ units leave) and the sub-station s (which receives $|T|$ units). Note that $P(u)$ (resp. $S(u)$) is the set of predecessors (resp. successors) of u, i.e., the set of nodes v such that (v,u) (resp. (u,v)) is an arc of G. The constraints (3) ensure that one unit of flow coming from the source 0 enters each of the turbine nodes. The constraints (4) express the fact that the flow on the kth copy of arc (u,v) cannot be greater than the capacity of that arc (denoted by C_{uv}) and there is a strictly positive flow on the arc only if t_{uv}^k equals 1. Observe that K denotes the set $\{1,2,\ldots,m\}$.

The constraints (5) express the fact that a given link (say, the kth link between u and v) can be used in one direction only. The constraints (6) and (7) express the fact that the $(k+1)$th link can be used only if the kth link is. Two groups of constraints are needed: one for asymmetric arcs and one for symmetric arcs. The constraints (8) and (9) relate the variables $z_{uvw}^{kk'}$, on one hand, and x_{uv}^k, on the other. The constraints (10) express the fact that the flow on the kth copy of arc (u,v) that is channelled through arc (v,w) cannot be greater than the capacity of arc uv or arc vw, and there is a strictly positive flow of that kind only if $y_{uvw}^{kk'}$ equals 1. In writing this constraint we used P_2 to denote the set of simple paths of length 2, i.e., the set of triples (u,v,w) such that (u,v) and (v,w) are arcs of G and u and v

are distinct nodes. Finally, constraints (11) express the fact that if the kth link between u and v is used in the network and v is not the sub-station, then the flow on the kth link should be channelled out of v.

$$\min \sum_{(u,v)\in A} \sum_{k=1}^{m} c_{uv}^k t_{uv}^k \quad \text{s.t.} \tag{1}$$

$$\sum_{v\in P(u)} \sum_{k=1}^{m} x_{vu}^k - \sum_{v\in S(u)} \sum_{k=1}^{m} x_{uv}^k = \begin{cases} |T| & \text{if } u = s \\ -|T| & \text{if } u = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\sum_{k=1}^{m} x_{0v}^k = 1 \quad \forall v \in T \tag{3}$$

$$x_{uv}^k \le C_{uv} t_{uv}^k \quad \forall (u,v) \in A, \forall k \in K \tag{4}$$

$$t_{uv}^k + t_{vu}^k \le 1 \quad \forall \{u,v\} \in E, \forall k \in K \tag{5}$$

$$t_{uv}^{k+1} \le t_{uv}^k \quad \forall (u,v) \in A_1, \forall k \in K, k < m \tag{6}$$

$$t_{uv}^{k+1} + t_{vu}^{k+1} \le t_{uv}^k + t_{vu}^k \quad \forall \{u,v\} \in E, \forall k \in K, k < m \tag{7}$$

$$\sum_{u\in P(v)} \sum_{k=1}^{m} z_{uvw}^{kk'} = x_{vw}^{k'} \quad \forall (v,w) \in A, v \ne 0, \forall k' \in K \tag{8}$$

$$\sum_{w\in S(v)} \sum_{k'=1}^{m} z_{uvw}^{kk'} = x_{uv}^k \quad \forall (u,v) \in A, v \ne s, \forall k \in K \tag{9}$$

$$z_{uvw}^{kk'} \le \min\left(C_{uv}, C_{vw}\right) y_{uvw}^{kk'} \quad \forall (u,v,w) \in P_2, \forall k,k' \in K \tag{10}$$

$$\sum_{w\in S(v)} \sum_{k'=1}^{m} y_{uvw}^{kk'} = t_{uv}^k \quad \forall (u,v) \in A, v \ne s, \forall k \in K \tag{11}$$

$$x_{uv}^k \ge 0 \quad \forall (u,v) \in A, \forall k \in K \tag{12}$$

$$z_{uvw}^{kk'} \ge 0 \quad \forall (u,v,w) \in P_2, \forall k,k' \in K \tag{13}$$

$$t_{uv}^k \in \{0,1\} \quad \forall (u,v) \in A, \forall k \in K \tag{14}$$

$$y_{uvw}^{kk'} \in \{0,1\} \quad \forall (u,v,w) \in P_2, \forall k,k' \in K \tag{15}$$

The above model exhibits many symmetries and the gap between the optimal values of the mathematical program and its linear relaxation can be large. Hence solving it by using a commercial package such as CPLEX can consume a lot of time. In Section 5 we describe some inequalities that tighten the linear programming relaxation of this model. Note that if the model has a feasible solution, it has an integral optimal solution. This follows easily, because if the t_{uv}^k and the $y_{uvw}^{kk'}$ have fixed integer values, we can solve a capacitated network flow problem in order to obtain integral values for the x_{uv}^k.

4 Related problems and related work

The operations research literature does not contain many references to the problem we address in the present article. Berzan *et al.* (2011) consider a problem similar to ours (but simpler) and decompose it into three layers: the circuit, the substation, and the full farm. When there is a single cable type, the problems for the first two layers reduce to graph-theoretic problems (the uncapacited and capacitated minimum spanning tree problems, respectively). They also formulate the circuit problem as a mixed integer program and use this formulation to solve instances with up to 8 turbines. Fagerfjäll (2010) addresses two models: the production model and the infrastructure model. Only the latter is related to ours, but there are differences between Fagerfjäll's model and ours: for instance the locations of the wind turbines are not completely fixed in Fagerfjäll's infrastructure model. The author has tested his models on instances with at most 30 turbines and reports that CPLEX takes a long time for producing results for the infrastructure model.

As we pointed out in Section 2, many feasible solutions of the collection network design problem actually are anti-rooted trees. This observation led us to explore the relationship between the feasible solutions of our problem, on one hand, and directed Steiner trees, on the other. We will show that there is a bijection between the feasible solutions of the collection network problem and certain directed Steiner trees in a related network. This network, denoted by $H = (V', A')$, is almost identical to the line-graph of the multigraph obtained from G by replacing each arc (u, v) of G by m parallel arcs between u and v. The vertex set of H (i.e., V') consists of s' (a new vertex) and all vertices of the form p_{uvk} (where uv is an arc of G and $1 \leq k \leq m$ holds). The arc set of H (i.e., A'), is defined formally as the union of

- the set of couples of the form $(p_{uvk}, p_{vw\ell})$ such that u and w are distinct vertices, and
- the set of couples of the form (p_{usk}, s').

We assign a cost of c_{uv}^k to an arc of the form $(p_{uvk}, p_{vw\ell})$ and a cost of c_{us}^k to an arc of the form (p_{usk}, s'). Given a feasible solution of the collection network problem defined on the graph G, it is straightforward to define a corresponding anti-rooted tree (directed Steiner tree) in the graph H. To do so, observe that the entire flow on the kth copy of arc (u, v) of G is routed on a single link of tail v. Thus the kth copy of arc (u, v) has one "successor" only, say, the ℓth copy of arc (v, w). Thus if a feasible solution of the network collection problem contains the kth copy of arc (u, v) and the flow on this link is channelled through the ℓth copy of arc (v, w), we include the arc $(p_{uvk}, p_{vw\ell})$ into the directed Steiner tree (within H). Similarly, if the feasible solution contains the kth copy of arc (u, s), we include the arc (p_{usk}, s') into the directed Steiner tree.

This construction can be reversed, i.e., a directed Steiner tree in H can be transformed into a feasible solution of the network collection problem, provided the following conditions are satisfied:

- if it includes a vertex of the form p_{uvk}, then it includes all vertices of the form $p_{uv\ell}$ for ℓ smaller than k; and
- if it includes a vertex of the form p_{uvk}, then the anti-rooted subtree of anti-root p_{uvk} contains at most C_{uv} turbines, where C_{uv} is the capacity of arc uv in the network collection problem.

Therefore if one wishes to formulate our problem as a directed Steiner tree problem, one must take these additional constraints into account. In practice we will not transform the graph G into the graph H, because the latter is very cumbersome.

Figure 3 displays the directed graph underlying a tiny instance of the collection network problem, and the graph constructed by the above transformation. In this example m equals 1. Nodes 1, 2, and 3 represent turbines and node 8 is the sub-station. In the graph on the right every node except s' corresponds to an arc of the tiny graph and is labelled accordingly: for instance node 24 corresponds to arc $(2, 4)$ and node 75 to arc $(7, 5)$. The black rectangles are the nodes of the line graph corresponding to the subgraph induced by $\{1, 2, 3\}$. Similarly the grey rectangles correspond to the arcs of the subgraph induced by the nodes of the above-ground network (except the sub-station 8). White rectangles represent two groups of arcs: the arcs linking a turbine to a node in the above-ground network, and the arcs whose head is the sub-station (i.e., node 8). A feasible solution of the collection network problem is displayed on the left: it consists of the "real" arcs $12, 25, 35, 54, 57, 46, 78, 68$ and the "fictitious" arcs $01, 02, 03$. This solution is not a tree. Observe that in order to describe the solution completely, one must specify that the flow on arc $(3, 5)$ is channelled through arc $(5, 7)$ and the flow on arc $(2, 5)$ through arc $(5, 4)$. The directed Steiner tree corresponding to this solution is displayed on the right.

In spite of the differences outlined above, the literature on Steiner trees is relevant for our problem. Karp (1972) proved that the Steiner tree problem is NP-complete and Goemans and Myung (1993) have given a survey of formulations for the Steiner tree problem. Exact algorithms for solving this problem have been proposed by Wong (1984), Beasley (1984), Beasley (1989), Lucena and Beasley (1998), and Chopra *et al.* (1992). Since the Steiner tree problem is NP-complete, many authors have also proposed heuristic algorithms for finding good solutions (see for instance Khoury and Pardalos (1996), Gendreau, Larochelle and Sansò

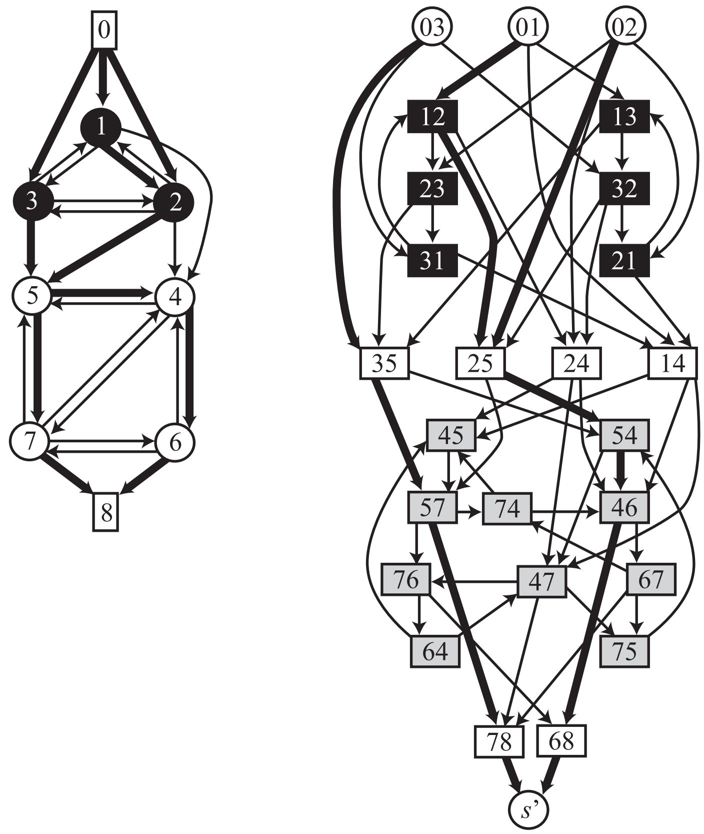

Figure 3: A solution of the collection design problem and the corresponding directed Steiner tree

(1999), Esbensen (1995), Voss and Gutenschwager (1998), and Ribeiro and Souza (2000)).

5 Model strengthening and cutting planes

As we noted at the end of Section 3, the model presented in that section must be strengthened. First, consider a node v that is neither the source nor the sub-station. Assume that the capacity of a link between u and v equals the capacity of any link between v and w, for any $w \neq u$. Then if the kth link from u to v is used in an optimal solution, we may conclude that the number of arcs coming out of v in that solution is at least k. Hence the following relation is satisfied.

$$kt_{uv}^{k} \leq \sum_{w \in S(v), w \neq u} \sum_{i=1}^{k} t_{vw}^{i} \quad \forall (u,v) \in A, \forall k \in K \tag{16}$$

The inclusion of Inequalities (16) into the model will remove some feasible solutions but no optimal solution. Note that in the special case where v is a turbine node, we obtain the inequality $1 = t^1_{0v} \leq \sum\limits_{w \in S(v)} t^1_{vw}$.

Now consider any optimal solution of the above model and assume that the capacity of a link between u and v equals the capacity of any link between v and w. If a quantity Q flows through the arc (v, w, k'), this same quantity (or a smaller one) can flow through **one** link between u and v. Therefore any optimal solution should include one such link only and the following constraint is satisfied.

$$\sum_{k=1}^{m} y^{kk'}_{uvw} \leq 1 \quad \forall (u, v, w) \in P_2, \forall k' \in K \tag{17}$$

Consider any arc of the form (u, s), where s denotes the sub-station. In any optimal solution, the $(k+1)$st link between u and s will be used only if the sum of the flows on the kth and $(k+1)$th links from u to s is greater than the capacity of a link (C_{ag}, in this case). Therefore the following inequality is satisfied by at least one optimal solution (recall that there is at least one integral optimal solution), but it is not satisfied by some feasible solutions.

$$(C_{\text{ag}} + 1) t^{k+1}_{us} \leq x^k_{us} + x^{k+1}_{us} \quad \forall (u, s) \in A, \forall k < m \tag{18}$$

We also observe that because of Constraints (11), the inequalities

$$y^{kk'}_{uvw} \leq t^k_{uv} \quad \forall (u, v, w) \in P_2, \forall k, k' \in K \tag{19}$$

are valid. The following constraints are satisfied by all optimal solutions.

$$y^{kk'}_{uvw} \leq t^{k'}_{vw} \quad \forall (u, v, w) \in P_2, \forall k, k' \in K \tag{20}$$

Finally, it is possible to remove some symmetry from the model by making sure that if there is a positive flow on path (u, v, w) using the kth copy of arc (u, v) and the k'th copy of arc (v, w), on one hand, and the ℓth copy of arc (u, v) and the ℓ' copy of arc (v, w), on the other, then $k < \ell$ implies that $k' \leq \ell'$. This condition can be expressed by the following constraints.

$$y^{kk'}_{uvw} + y^{\ell\ell'}_{uvw} \leq 1 \quad \forall (u, v, w) \in P_2, \forall k, k', \ell, \ell' \text{ such that } k < \ell, k' > \ell' \tag{21}$$

We now turn to a large family of valid inequalities that play a crucial role in the tightening of the model. There are too many of them for us to include them all in the model, but it is possible to find (in a reasonable time) some inequalities of this type that are violated by the current linear programming relaxation. Such inequalities are called *cutting planes*. We first give two instances of inequalities belonging to this family. Recall that $P(s)$ is the set of predecessors of the sub-station s and let D_s denote the set of arcs of the form (u, s, k) for u in $P(s)$. For the flow of all turbines to reach the sub-station, the following inequality must be satisfied (where C^k_{uv} denotes the capacity of the link (u, v, k) and $|T|$ the number of turbines).

$$\sum_{(u,v,k)\in D_s} C^k_{uv} t^k_{uv} \geq |T|$$

Since the t^k_{uv} are integers, many valid inequalities can be derived from this one. For example, if all the C^k_{uv} appearing in this inequality equal C_{ag} (as in the instances supplied by the industrial partner), we may conclude that the inequality

$$\sum_{(u,v,k)\in D_s} t^k_{uv} \geq \lceil |T|/C_{\text{ag}} \rceil \tag{22}$$

is satisfied by any feasible solution of the mixed integer program.

In a similar fashion, if D_b (where b stands for "border") denotes the set of all arcs going from the underground network to the above-ground network, the inequality

$$\sum_{(u,v,k)\in D_b} C^k_{uv} t^k_{uv} \geq |T|$$

holds. If all the arcs in D_b have a capacity of C_{ug} (as in our instances), we may conclude that the inequality

$$\sum_{(u,v,k)\in D_b} t^k_{uv} \geq \lceil |T|/C_{\text{ug}} \rceil \tag{23}$$

is satisfied by any feasible solution of the mixed integer program. The two inequalities

$$\sum_{(u,v,k)\in D_s} t^k_{uv} \geq \lceil |T|/C_{\text{ag}} \rceil, \quad \sum_{(u,v,k)\in D_b} t^k_{uv} \geq \lceil |T|/C_{\text{ug}} \rceil$$

are actually special cases of Inequality (25) below, which we now derive.

Consider a subset of vertices V_1 containing the vertex 0 but not s. The set of arcs (u, v) such that u belongs to V_1 and v to the complement of V_1 is called a *cut*. This cut separates 0 from s. Let D be a cut separating 0 from s and D_1 the subset of arcs in D whose origin is the source (i.e., 0). We also let T' denote the set of turbines u such that $(0, u)$ does not belong to D_1. We have

$$\sum_{(u,v,k)\in D} C^k_{uv} t^k_{uv} \geq |T|$$

and thus

$$\sum_{(u,v,k)\in D\setminus D_1} C^k_{uv} t^k_{uv} \geq |T| - \sum_{(u,v,k)\in D_1} t^k_{uv}.$$

Since all the arcs (u, v, k) in D_1 have the property that t^k_{uv} equals 1, we obtain

$$\sum_{(u,v,k)\in D\setminus D_1} C^k_{uv} t^k_{uv} \geq |T| - |D_1| = |T'|, \tag{24}$$

and one can derive many valid inequalities from this one. For instance, if M denotes the largest value of C^k_{uv} for $(u, v, k) \in D\setminus D_1$, the inequality

$$\sum_{(u,v,k)\in D\setminus D_1} t^k_{uv} \geq \lceil |T'|/M \rceil \tag{25}$$

is satisfied by any feasible solution of the mixed integer program.

It is possible to derive at least another valid inequality from (24). If we assume that all the capacities are equal to C_{ag} or C_{ug} and let q denote their greatest common divisor, then the following inequality is valid.

$$\sum_{(u,v,k)\in D\setminus D_1} \left(C^k_{uv}/q\right) t^k_{uv} \geq \lceil |T'|/q \rceil \tag{26}$$

When there are more than two values for the arc capacities, one can derive even more valid inequalities, for instance all the inequalities valid for a knapsack equation.

We now assume that we have solved the linear programming relaxation of the model described in Section 3. Some of the inequalities described at the beginning of this section may have been included into the model. We are now looking for instances of Inequality (25) that are violated by the current solution of the relaxation. In theory it is difficult to find violated inequalities because we don't know T'. We now outline an algorithm for finding a violated inequality. This algorithm constructs an auxiliary network having the same underlying graph as the original network, but different capacities.

- Choose a subset T' of turbines and define D_1 as the set of arcs $(0, u)$ for u in $T\setminus T'$.
- Define the capacity of arc $(0, u)$ as 1 for every turbine u in T' and 0 otherwise.
- Assign the capacity t^k_{uv} to any arc (u, v, k) such that $u \neq 0$ holds. In practice one can merge all the arcs of the form (u, v, k) for some couple (u, v) with $u \neq 0$, i.e., create a single arc from u to v of capacity $\sum_{k=1}^{m} t^k_{ij}$.
- Solve the maximum flow problem in the auxiliary network, where node 0 is the source and s the sink. Let D' denote a cut of minimum capacity separating 0 from s.
- Let D be the arc set obtained from D' by including into it all the arcs in D_1 and removing from it all the arcs of the form (u, v) for some u in $T\setminus T'$. Return D.

The above algorithm will be referred to as *separating* T' *from* s. Given the set T', a cut D is called a *standard cut for* T' if it contains D_1 (as defined at the beginning of the algorithm).

Proposition 5.1 *The arc set* D *returned by the above algorithm is a minimum-capacity cut separating* 0 *from* s. *If there exists a cut* D *that separates* 0 *from* s, *is a standard cut for* T', *and satisfies*

$$\sum_{(u,v,k)\in D\setminus D_1} t_{uv}^k < \lceil |T'|/M \rceil ,$$

then the above algorithm will return such a cut.

Proof. The maximum flow algorithm produces a partition of the vertex set V into sets V_1' and V_2' such that 0 is in V_1', s in V_2', and the capacity of the cut $D' = \{(u, v, k) \mid u \in V_1', v \in V_2'\}$ is minimal. If V_2' does not contain $T\setminus T'$, we define V_1 as $V_1'\setminus(T\setminus T')$ and V_2 as $V_2' \cup (T\setminus T')$. The cut defined by V_1 and V_2 is precisely the cut D returned by the algorithm, and it is a minimum-capacity cut because each arc in D_1 has a capacity of 0.

The second statement of the proposition follows easily from the first, since we may always assume that a minimum-capacity cut contains D_1 (by the argument in the preceding paragraph). □

For obvious reasons, one cannot run the above algorithm for every subset T' of turbines. In the next section we will discuss the strategy we employed. We conclude this section by noting that one can use a similar algorithm for finding violated inequalities of the form (26).

6 The experiments

We conducted experiments on nine instances provided by the industrial partner. Table 1 contains the description of these instances. We ran the model given in Section 3 on the nine instances using CPLEX 12.3; for Instance no. 7 one hour of computing time was not sufficient to find an (integral) feasible solution. We then included the Inequalities (16) to (21) into the model and after some experiments, concluded that only Inequalities (16) and (17) were useful. For Instance no. 3 CPLEX 12.3 could not find an integral solution within one hour of computing time.

We then introduced some cutting planes of the form (25). We proceeded as follows. We first introduced the constraints (22) and (23) into the model. Then we introduced three groups of cutting planes. The first group of cutting planes corresponds to inequalities of the form (25) where T' is a singleton (i.e., a set consisting of one turbine). Algorithm 1 describes the procedure we used to generate

Table 1: Description of the instances

	Number of turbines	Number of vertices	Number of arcs	Capacity C_{ug}	Capacity C_{ag}
Instance 1	40	143	384	5	10
Instance 2	88	220	517	10	25
Instance 3	53	306	868	10	15
Instance 4	73	256	679	10	15
Instance 5	33	64	160	11	11
Instance 6	42	91	232	11	11
Instance 7	60	189	478	3	10
Instance 8	112	281	793	5	12
Instance 9	79	231	677	5	12

this first group of cutting planes. Note that we also generated all the sets T' of cardinality 2 and added the corresponding cutting planes. In the final experiments, however, we only used the sets T' of cardinality one, since those of cardinality 2 did not improve by much the optimal value of the linear relaxation.

Algorithm 1 First group of cutting planes

for each turbine u **do**
  set T' equal to $\{u\}$
  repeat
    separate T' from s
    let D be a minimum-capacity standard cut for T'
    if Inequality (25) for D is violated **then**
      add Inequality (25) to the model
      solve the linear relaxation of the model
    end if
  until the optimal value of the linear relaxation has not increased
end for

For the second group of cutting planes, we set T' equal to T, i.e., we tried to separate the set of all turbines from the sub-station. We added Inequalities (25) and (26) to the model whenever they were violated. Algorithm 2 describes our procedure. Finally, we tried to find more cutting planes by considering each vertex v of the graph in turn. We let $M(v)$ denote the set of vertices u such that there is a path from u to v consisting of arcs (w, w', k) with $t^k_{ww'} > 0$. We also let $T(v)$ denote $M(v) \cap T$ and $D(v)$ the cut $\{(w, w') \in A \mid w \in M(v), w' \notin M(v)\}$. If Inequality (25) (with D replaced by $D(v)$) was violated, we memorized $T(v)$ and called it S_i (where i is the index of the current cutting plane). Once all the useful subsets of

Algorithm 2 Second group of cutting planes

```
repeat
    separate T from s
    let D be a minimum-capacity standard cut for T
    if Inequality (25) or (26) for D is violated then
        add the violated inequalities to the model
        solve the linear relaxation of the model
    end if
until the optimal value of the linear relaxation has not increased
```

turbines had been computed and memorized, we separated each subset from s and added a group of new cutting planes to the model. This procedure was repeated until the optimal value of the linear relaxation could not be improved. The whole subalgorithm is summarized in Algorithm 3.

The three groups of cutting planes that we have just described are computed only at the root node of the branch-and-bound tree. The experimental results are summarized in Table 2. Note that a table entry contains a question mark whenever CPLEX 12.3 could not find an integral feasible solution within one hour of computing time. With the formulation of Section 3, we could find optimal solutions for two instances only. After including constraints (16) and (17) into the model, four more instances could be solved to optimality, and after including (16), (17), and the cutting planes into the model, 7 instances out of 9 could be solved to optimality. In particular, the introduction of cutting planes enabled us to find an optimal solution of Instance no. 3, which has more vertices and more arcs than any other instance.

Note that Instances 7 and 8 are more "difficult" than the other ones. This may be due to the "incompatibility" between the values of C_{ug} and C_{ag}: in each case those values are relatively prime! We decided to try and solve the model for those two instances by including only the constraints of the original model, (16), (17), (22), and (23). After one hour of computing time CPLEX had not found an optimal solution for either instance, but it could find an optimal solution of Instance no. 7 in a little more than 8 hours. We could not solve Instance no. 8 because of a lack of memory. These results are summarized in Table 3.

7 Conclusion

In this article we have formulated the design of a wind farm collection system as a mixed integer programming problem. We have shown that tightening the mixed integer programming formulation and including some cutting planes at the root node of the branch-and-bound tree enabled one to solve all the instances provided by the industrial partner (except one). There is still much work to do, however. For

Algorithm 3 Third group of cutting planes

$i := 0$
for each vertex v **do**
  compute $M(v)$, $T(v)$, and $D(v)$
  if Inequality (25) with D replaced by $D(v)$ is violated **then**
    $i := i + 1$
    memorize $T(v)$ and call it S_i
  end if
end for
repeat
  for each index i **do**
    separate S_i from s
    let D_i be a minimum-capacity standard cut for S_i
    if Inequality (25) for D_i is violated **then**
      add Inequality (25) to the pool of new cutting planes
    end if
  end for
  add all the cutting planes found to the model
  solve the linear relaxation of the model
until the optimal value of the linear relaxation has not increased

Table 2: Summary of results

	Original Model			Model, (16), and (17)			Model, (16), (17), and cutting planes		
	Solution	Gap (%)	CPU (s)	Solution	Gap (%)	CPU (s)	Solution	Gap (%)	CPU (s)
Ins. 1	88424.8	3.72	3600	87592.4	0.0	206	87592.4	0.0	429
Ins. 2	126321.0	1.12	3600	126321.0	0.0	20	126321.0	0.0	48
Ins. 3	123941.0	16.28	3600	?	?	3600	121779.0	0.0	2588
Ins. 4	119887.0	3.31	3600	119887.0	0.0	975	119887.0	0.0	598
Ins. 5	45668.8	0.0	6	45668.8	0.0	1	45668.8	0.0	4
Ins. 6	63585.6	0.0	205	63585.6	0.0	45	63585.6	0.0	49
Ins. 7	?	?	3600	114185.0	2.86	3600	115241.0	5.29	3600
Ins. 8	190160.0	9.97	3600	185867.0	5.7	3600	188151.0	6.95	3600
Ins. 9	122189.0	3.07	3600	121909.0	0.0	1903	121909.0	0.0	1892

Table 3: Results with the original constraints plus (16), (17), (22), and (23)

	With a time limit			Without a time limit		
	Solution	Gap (%)	CPU (s)	Solution	Gap (%)	CPU (s)
Ins. 7	114323.6	2.94	3600	114185.0	0.0	30085
Ins. 8	187773.3	6.68	3600	185037.0	2.07	164015

instance, we have seen that it is in general difficult to recognize violated inequalities of the form (25) because there is a huge number of choices for the set T'. Hence one should try to develop heuristics for finding more violated inequalities of the form (25). Then one could try to find such inequalities at the internal nodes of the branch-and-bound tree, in the hope of decreasing the running time of the algorithm. Finally, given the close relationship between the problem described in this article and the Steiner tree problem, one should investigate ways in which to adapt to our problem the exact and heuristic algorithms developed for the Steiner tree problem.

Acknowledgments

The authors are very grateful to NSERC for its support through the discovery grants of Alain Hertz and Odile Marcotte, and to Hatch for providing the data sets on which our algorithms were tested.

References

Beasley, J. E. (1984). An Algorithm for the Steiner Problem in Graphs. *Networks* **14**:147–159.

Beasley, J. E. (1989). An SST-based Algorithm for the Steiner Problem in Graphs. *Networks* **19**:1–16.

Berzan, C., Veeramachaneni, K., McDermott, J., O'Reilly, U.-M. (2011). Algorithms for Cable Network Design on Large-scale Wind Farms. Technical Report, Massachusetts Institute of Technology.

Chopra, S., Gorres, E. R., Rao, M. R. (1992). Solving the Steiner Tree Problem on a Graph Using Branch-and-Cut. *ORSA Journal on Computing* **4**:320–335.

Esbensen, H. (1995). Computing Near-optimal Solutions to the Steiner Problem in a Graph Using a Genetic Algorithm. *Networks* **26**:173–185.

Fagerfjäll, P. (2010). Optimizing Wind Farm Layout - More Bang for the Buck Using Mixed Integer Linear Programming. Master's Thesis, Department of Mathematical Sciences, Chalmers University of Technology and Gothenburg University, Göteborg, Sweden.

Gendreau, M., Larochelle, J.-F., Sansò, B. (1999). A Tabu Search Heuristic for the Steiner Tree Problem. *Networks* **34**:162–172.

Goemans, M. X., Myung, Y.-S. (1993). A Catalog of Steiner Tree Formulations. *Networks* **23**:19–28.

Karp, R. M. (1972). Reducibility Among Combinatorial Problems. *Complexity of Computer Computations (R. E. Miller and J. W. Thatcher, eds.)*, 85–103.

Khoury, B. N., Pardalos, P. M. (1996). A Heuristic for the Steiner Problem in Graphs. *Computational Optimization and Applications* **6**:5–14.

Lucena, A., Beasley, J. E. (1998). A Branch-and-Cut Algorithm for the Steiner Problem in Graphs. *Networks* **31**:39–59.

Ribeiro, C. C., Souza, M. C. D. (2000). Tabu Search for the Steiner Problem in Graphs. *Networks* **36**:138–146.

Voss, S., Gutenschwager, K. (1998). A Chunking Based Genetic Algorithm for the Steiner Tree Problem in Graphs. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **40**:335–355.

Wong, R. T. (1984). A Dual Ascent Approach for Steiner Tree Problems on a Directed Graph. *Mathematical Programming* **28**:271-287.