MIP modeling of Incremental Connected Facility Location

Ashwin Arulselvan, Andreas Bley, Stefan Gollowitzer, Ivana Ljubić, and Olaf Maurer

Abstract We consider the incremental connected facility location problem, in which we are given a set of potential facilities, a set of interconnection nodes, a set of customers with demands, and a planning horizon. For each time period, we have to select a set of facilities to open, a set of customers to be served, the assignment of these customers to the open facilities, and a network that connects the open facilities. Once a customer is served, it must also be served in subsequent periods. Furthermore, in each time period the total demand of all customers served must be at least equal to a given minimum coverage requirement for that period. The objective is to maximize the net present value of the network, which is given by the discounted revenues of serving the customers and by the discounted investments and maintenance costs for the facilities and the network.
We study different MIP models for this problem, discuss some valid inequalities to strengthen these formulations, and present a branch and cut algorithm for finding its solution. Finally, we report (preliminary) computational results of our implementation of this algorithm.

Ashwin Arulselvan, Andreas Bley and Olaf Maurer
Technische Universität Berlin, e-mail: {arulsel,maurer,bley}@math.tu-berlin.de, Straße des 17. Juni 136, 10623 Berlin, Germany

Stefan Gollowitzer
Department of Statistics and Operations Research, e-mail: stefan.gollowitzer@univie.ac.at, University of Vienna, Austria

Ivana Ljubić
Institute of Computer Graphics and Algorithms, e-mail: ivana@ads.tuwien.ac.at, Vienna University of Technology, Austria

1 Introduction

Practical Problem

The problem under consideration is an optimal design of a network topology in the context of a multi-period planning of local access networks. In this setting, a telecommunication company wants to increase the speed of broadband connections by combining fiber optic technology with existing copper connections, i.e., by means of the *Fiber-to-the-Curb* (FTTC) technology. Street segments along which fiber optic cables can be installed, determine the *core network*. Potential optical and *existing copper cables* intersect at locations where potential multiplexor devices need to be installed. Between a multiplexor and an end-customer, the existing copper connection is used. The existing copper paths are pre-processed building an *assignment network* whose edges are assignment links between potential multiplexor locations and end-customers. To build an FTTC network, one has to decide on which locations to install multiplexor devices so that each end-customer is assigned to a multiplexor, and each multiplexor is connected to the *central office* by a fiber optic path.

Due to the huge investment needed to build an FTTC network, the deployment is done in several stages. The company takes the strategic decision of fixing a minimal percentage of *customer demands* that should be served at each of the stages. Thereby, demand of a customer is defined as the number of end-subscribers (e.g., offices and/or households) behind the customer's address. The coverage of customer demands need to be increased over time.

We define the *incremental connected facility location problem*, denoted as *incremental ConFL*, as follows: We are given three disjoint sets of nodes: a set of facilities F, a set of customers R, and a set of Steiner nodes M. We denote $S = F \cup M$ and $V = S \cup R$. The potential connections among the nodes in S build the *core network* and are given as the undirected edge set E_S. The corresponding directed arc set is $A_S = \{(i,j),(j,i) \mid ij \in E_S\}$. The possible connections between the facilities F and the customers R are given by the edges $E_R \subseteq F \times R$, which define the directed arc set $A_R = \{(i,j) \in F \times R \mid ij \in E_R\}$. Note that it is sufficient to consider only arcs directed from facilities to customers here. We let $A = A_S \cup A_R$ and $E = E_S \cup E_R$. The considered planning horizon is given as a discrete set of (not necessarily equally long) time periods $T = \{1,\dots,\mathcal{T}\}$, $\mathcal{T} > 1$. In addition, we are given fixed costs for edges $c : E \to \mathbb{R}_+$ and facilities $g : F \to \mathbb{R}_+$ for opening the edge or facility for the first time, and maintenance costs for edges $m : E \to \mathbb{R}_+$ and facilities $m_f : F \to \mathbb{R}_+$ that arise for each period an edge or a facility is actually used. The pre-period revenue for serving the customers is given by $p : R \to \mathbb{R}_+$. Finally, we are given customer demands $d : R \to \mathbb{Z}_+$ and a minimum coverage requirement D^t for each time period $t \in T$.

We seek for a schedule that, for each time period, describes which subset of facilities to use, which set of customers to serve by these facilities, how to assign the served customers to the open facilities, and how to build the core network in order to connect the open facilities. In each time step, the total demand of the served customers must satisfy the minimum coverage requirement and the chosen edges in E_S must form a network connecting the open facilities. Furthermore, a customer must be served in all periods after it has been served for the first time. The goal is to maximize the net present value of the network.

Related Multi-Period Optimization Problems: Facility location problem over time is a well-studied problem. A recent survey is given in Owen and Daskin [13]. In a recent work, Albareda-Sambola et al. [2] consider a multi-period *incremental facility location problem*, where the coverage of customer demand needs to be increased over time. The authors combine subgradient optimization and a Lagrangian approach and generate feasible solutions with a Lagrangian based heuristic.

There has been intense research on multi-period network design problems since publication of the seminal articles by Christofides and Brooker [6], Doulliez and Rao [7] and Zadeh [17]. Optimization methods have been used for designing networks for telecommunication, transportation [16], distribution of gas or water [14] and many others.

Most of the literature on applications in the telecommunications sector consider capacitated problems. Recent contributions are, e.g., [5, 11]. Much less literature is available on the Connected Facility Location problem.

Single-Period Connected Facility Location: Early work on ConFL mainly includes approximation algorithms. The problem can be approximated within a constant ratio and the currently best-known approximation ratio is provided by Eisenbrand et al. [8]. Ljubić [12] describes a hybrid heuristic combining Variable Neighborhood Search with a reactive tabu search method. The author compares it with an exact branch and cut approach. In [15], a Greedy Randomized Adaptive Search Procedure (GRASP) for the unrooted ConFL problem is presented. The authors also provide a transformation that enables solving ConFL as the Steiner arborescence problem. Bardossy and Raghavan [3] develop a dual-based local search (DLS) heuristic for a generalization of the ConFL problem. The presented DLS heuristic computes lower and upper bound using a dual-ascent and then improves the solution with a local search procedure. Gollowitzer and Ljubić [9] study MIP formulations for ConFL, both theoretically and computationally. The authors provide a complete hierarchy of ten MIP formulations with respect to the quality of their LP bounds.

The remainder of this paper is organized as follows. In Section 2 we present integer programming formulations for the incremental ConFL problem and discuss a class of valid inequalities that may be used to strengthen these formulations. Section 3 provides a description of the separation subroutines that we implemented in order to solve these models. In Section 4 we describe the benchmark data sets, details of our implementation of the branch and cut algorithm, and (preliminary) results of our computational experiments.

2 MIP Modeling

In this section we present two alternative integer programming formulations for the incremental connected facility location problem.

We assume that one of the facilities, denoted as root r is open and used in all time periods. This node corresponds to the central office with an uplink to the backbone network of the area corresponding to the respective instance.

In order to model the connectivity constraints among the open facilities, it is sufficient to ensure that all other open facilities in F are connected to the root r [9]. For notational simplicity, we let F denote the set of all facilities except r throughout the remainder of this paper. Furthermore, we denote $\delta^-(W) := \{(i,j) \in A \mid j \in W,\, i \notin W\}$ for all $W \subset V$ and $F(j) := \{i \in F \mid (i,j) \in A_R\}$ for all $j \in R$.

In order to describe which customers and facilities are served and used at each time period, we introduce binary variables $y_j^t \in \{0,1\}$ for all $j \in R$ and $t \in T$ and $z_i^t \in \{0,1\}$ for all $i \in F$ and for all $t \in T$. These variables are interpreted as

$$y_j^t = \begin{cases} 1 \text{ if customer } j \text{ is served in time period } t, \\ 0 \text{ otherwise, and} \end{cases}$$

$$z_i^t = \begin{cases} 1 \text{ if facility } i \text{ is used in time period } t, \\ 0 \text{ otherwise.} \end{cases}$$

The assignment of the served customers to the open facilities and the network connecting the open facilities to the root node are modeled together by the arc variables $x_{ij}^t \in \{0,1\}$ for all directed arcs $(i,j) \in A$ and for all time periods $t \in T$, which are interpreted as

$$x_{ij}^t = \begin{cases} 1 \text{ if arc } (i,j) \text{ is used in time period } t, \\ 0 \text{ otherwise.} \end{cases}$$

To describe the initial opening of facilities and edges, we also introduce the facility variables $\tilde{z}_i^t \in \{0,1\}$ for all $i \in F$ and all $t \in T$ and the aggregated edge variables $\tilde{x}_e^t \in \{0,1\}$ for all $e \in E$ and all $t \in T$, which are interpreted as

$$\tilde{z}_i^t = \begin{cases} 1 \text{ if facility } i \text{ is opened for the first time in time period } t, \\ 0 \text{ otherwise and} \end{cases}$$

$$\tilde{x}_e^t = \begin{cases} 1 \text{ if edge } e \text{ is opened for the first time in time period } t, \\ 0 \text{ otherwise.} \end{cases}$$

Observe that variables $\tilde{x}_e^t$ are associated to edges instead to arcs of the core network for the following reason: In the general case, a facility $i \in F$ may be opened in period $t \in T$, and closed in period $t+k \in T$ $(k > 0)$. Consequently, an arc that was oriented like (i, j) in period t, may be used in the opposite direction in period $t + k$. Since the edge opening costs need to be payed only once, we have to leave the direction of set-up variables $\tilde{x}$ unspecified.

With these variables and notations, the objective function of the incremental ConFL problem can be formulated as follows:

$$\mathbf{f}(x, y, z) = \sum_{t=1}^{T} (1+\alpha)^{-t} \left[\sum_{j \in R} p_j y_j^t - \sum_{e \in E} c_e \tilde{x}_e^t - \sum_{(i,j) \in A} m_{ij} x_{ij}^t \quad - \sum_{i \in F} g_i \tilde{z}_i^t - \sum_{i \in F} m_i z_i^t \right]$$

For each period $t \in T$, the objective function comprises the collected profit for customers served in period t decreased by the investment (maintenance) costs that need to be paid for each edge and facility that are opened (used) in this period. The following mixed integer programming formulation models the incremental ConFL:

$$(CUT_F): \quad \max \ \mathbf{f}(x, y, z)$$

$$\sum_{i \in F(j)} x_{ij}^t = y_j^t \qquad \forall j \in R,\ t \in T \qquad (1)$$

$$x_{ij}^t \leq z_i^t, \qquad \forall (i, j) \in A_R,\ t \in T \qquad (2)$$

$$x_{ij}^t + x_{ji}^t \leq \sum_{k=1}^{t} \tilde{x}_e^k \qquad \forall (i, j) = e \in E,\ t \in T \qquad (3)$$

$$z_i^t \leq \sum_{k=1}^{t} \tilde{z}_i^k \qquad \forall i \in F,\ t \in T \qquad (4)$$

$$\sum_{j \in R} d_j y_j^t \geq D^t \qquad \forall t \in T \qquad (5)$$

$$y_j^t \geq y_j^{t-1} \qquad \forall j \in R,\ t \in T \qquad (6)$$

$$\sum_{(u,v) \in \delta^-(W)} x_{uv}^t \geq z_j^t \qquad \forall W \subseteq S \backslash \{r\},\ j \in W \cap F \neq \emptyset,\ t \in T \qquad (7)$$

$$x^t_{kl}, y^t_j, z^t_i \in \{0,1\} \qquad \forall (k,l) \in A,\ j \in R,\ i \in F,\ t \in T \tag{8}$$

$$\tilde{x}^t_e, \tilde{z}^t_i \in \{0,1\} \qquad \forall e \in E,\ i \in F,\ t \in T \tag{9}$$

Constraints (1) model the fact that a customer is served only if there is a facility connected to it. Constraints (2) enforce that a facility is open if it is used to serve a customer. Inequalities (3) and (4) ensure that we *open* edges and facilities as soon as they are *used*. Constraint set (5) expresses the minimum demand coverage requirement for each time period. Inequalities (6) enforce the continuance of service for each customer (i.e., if customer j was served in period $t \in T$, it also need to remain served in all consecutive periods). Finally, the exponentially large constraint set (7) ensures that, in each time period, all open facilities are connected to the root node. The inequalities in constraint set (7) enforce that for every subset $W \subseteq S$ that includes a facility j and does not include the root node r, at least one of the arcs in the set of all incoming arcs in W must be *used* if facility j is open. These inequalities correspond to the directed cutset inequalities in the Steiner tree formulation [9, 12].

Instead of enforcing at least one arc in each directed cut that separates a chosen facility from the root node, as done by constraints (7), we may model the connectivity constraints by enforcing at least one arc in every directed cut that separates a chosen customer from the root node. This leads to the following alternative formulation for the incremental ConFL problem:

$$(CUT_R): \quad \max\ \mathbf{f}(x,y,z)$$

$$(x,y,z) \text{ satisfies } (1)-(6)$$

$$\sum_{(u,v)\in\delta^-(W)} x^t_{uv} \ge y^t_j \qquad \forall W \subseteq V\backslash\{r\},\ j \in W \cap R,\ t \in T \tag{10}$$

2.1 Valid Inequalities

In this section we provide two new families of valid inequalities that can strengthen the previous two models. The third group of constraints presented here are several degree-inequalities that were very useful throughout our computations.

Cover Inequalities:

The minimum coverage constraints (5) imply a set of *cover inequalities* that can be defined for each single period $t \in T$. We call a subset of facilities $I^t \subset F$ a *cover* if its complement, $\bar{I}^t = F \setminus I^t$, cannot serve enough customers to satisfy the minimal demand requirements for the time period t. We denote by $COV^t \subseteq 2^F$ the family of all covers for period t. We call an inclusion-wise minimal such facility set I^t a *minimal cover*. In other words, I^t is a minimal cover if $\bar{I}^t$ cannot satisfy the minimum demand requirement of period t even if all the facilities in $\bar{I}^t$ are open, but for any $i \in I^t$ the facility set $J = \bar{I}^t \cup \{i\}$ would allow to serve enough customers to meet the minimum coverage constraint. In such a case, obviously at least one facility from I^t needs to be opened. Consequently, the following set of *cover inequalities* are valid for all solutions of (CUT_F) and (CUT_R):

$$\sum_{i \in I^t} z_i^t \geq 1 \quad \forall t \in T,\ I^t \in COV^t \tag{11}$$

It is easy to verify that the cover inequality for any non-minimal cover I^t is dominated by the cover inequality for any minimal cover $I^t_{\min} \subseteq I^t$. Furthemore, any non-minimal cover I^t can be easily turned into a minimal cover by iteratively removing all those facilities, whose removal still results in a cover.

It is also not difficult to construct examples where the addition of cover inequalities (11) strengthens the LP relaxations of (CUT_F) and (CUT_R). These inequalities are similar to the cover inequalities studied for knapsack constraints.

The separation of cover inequalities is a modification of the knapsack problem, and hence it is an NP-hard problem. Our separation algorithm for cover inequalities is described in Section 3.2.

Cut-Set-Cover Inequalities:

The set of cover inequalities (11) also implies the following exponentially large family of cut-set inequalities, that we will refer to as *cut-set-cover inequalities*:

$$\sum_{uv \in \delta^-(W)} x_{uv}^t \geq 1 \quad \forall t \in T,\ I^t \in COV^t, W \subseteq S \setminus \{r\}, I^t \subseteq W \tag{12}$$

These inequalities state that, in each period $t \in T$, we have to establish a path between the root and at least one of the facilities from the set I^t. Once the corresponding covers I^t become known, the separation of these new inequalities can be done in polynomial time by means of a maximum flow algorithm, see Section 3.2.

Again, it is not difficult to show that the addition of the cut-set-cover inequalities (12) strengthens the LP relaxations of (CUT_F) and (CUT_R).

In-Arc Inequalities:

The requirement that, in each time period, the root node is connected to any open facility, implies the following *in-arc inequalities*:

$$z_i^t \leq \sum_{(j,i)\in\delta^-(i)} x_{ji}^t \qquad \forall i \in F,\, t \in T \tag{13}$$

$$x_{ik}^t \leq \sum_{(j,i)\in\delta^-(i):j\neq k} x_{ji}^t \qquad \forall (i,k) \in A_S,\, i \neq r,\, t \in T \tag{14}$$

Inequalities (13) imply that there is at least one arc entering any chosen facility. Inequalities (13) ensure that there is at least one arc entering any facility or Steiner node if there is an arc leaving that node.

Note that these inequalities are implied by the cut inequalities (7) or (10), but not vice versa. However, there is only a polynomial number of inequalities of type (13) and (14), which makes these inequalities very useful in practical computations [9, 10].

Furthermore, we add the inequalities

$$\sum_{(j,i)\in\delta^-(i)} x_{ji}^t \leq 1 \qquad \forall i \neq r,\, t \in T \tag{15}$$

to the LP relaxations of (CUT_F) and (CUT_R). The inequalities ensure that the indegree of every node except the root node is at most 1. These inequalities may cut off feasible solutions but as there are no capacity constraints associated with the facilities and edges, there always *exists an optimal* solution of incremental ConFL that satisfies these inequalities. Adding these inequalities to the formulations substantially reduced the solution times in our experiments.

3 Separation Algorithms

In this section we explain separation algorithms for the cover inequalities and the three groups of cut-set inequalities described above.

3.1 Separation of Cut-Set Inequalities

We now present the separation routine to generate cut inequalities of type (7). Let $\hat{x}^t$ and $\hat{z}^t$ by the values of the arc variables and of the facility variables of the current optimal LP solution. In order to find a violated inequality of type (7), we compute for each time period $t \in T$ and each facility node $j \in F$ a minimum r-j-cut in the digraph $G(S, A_S)$ with arc capacities $\hat{x}^t$, solving the corresponding maximum flow problem. Let $\Gamma(r, j)$ be the set of arcs in the minimum cut obtained from this maximum flow computation. If the corresponding maximum flow value is less than $\hat{z}_i^t$, the corresponding cut inequality

$$\sum_{(u,v)\in \Gamma(r,j)} x^t_{uv} \geq z^t_j \tag{16}$$

is violated and we add this inequality to the current formulation.

The separation of the customer based cutset inequalities (10) is carried out analogously. We now consider the entire digraph $G(V, A)$ with capacities $\hat{x}^t$ given by the LP solution's arc variable values and solve the maximum flow problem with the root node r as the source and the customer node j as the sink for each customer $j \in R$ and each time period. Again, let $\Gamma(r, j)$ be the arcs of the corresponding minimum cut. If the maximum flow value is less than y^t_i, we add the violated cut

$$\sum_{(u,v)\in \Gamma(r,j)} x^t_{uv} \geq y^t_i \ . \tag{17}$$

3.2 Separation of Cover and Cut-Set-Cover Inequalities

Let $t \in T$ and let $\hat{z}^t$ be the values of the facility variables in the current LP solution. In order to find a cover I^t for which the corresponding cover inequality (11) is violated, we introduce variables $\alpha_i \in \{0, 1\}$ for all $i \in F$ indicating which facilities are contained in I^t and $\beta_j \in \{0, 1\}$ for all $j \in R$ indicating which customers can be served by any of the facilities not in I^t. Clearly, a cover I^t that maximizes the violation of inequality (11) corresponds to an optimal solution of the following integer program:

$$\min \sum_{i\in F} \hat{z}^t_i \alpha_i \tag{18}$$

$$\sum_{j\in R} d_j \beta_j \leq D^t - \epsilon \tag{19}$$

$$\beta_j \geq 1 - \alpha_i \qquad \forall (i, j) \in A_R \tag{20}$$

$$\alpha_i, \beta_j \in \{0, 1\} \qquad \forall i \in F, j \in R \tag{21}$$

Inequalities (20) guarantee that all clients that have at least one neighboring facility not in I^t are served, while constraint (19) ensures that the total demand of all served clients is strictly less than the demand required to meet the coverage constraint. Together, these constraints ensure that, for any integer solution of (18) - (21), the set of facilities i with $\alpha_i = 1$ forms a cover. Note that the objective value of a solution of (18) - (21) is equal to the left hand side of the corresponding cover inequality for the current LP solution. Finding a violated cover inequality thus is equivalent to finding a time period $t \in T$ and a solution of (18) - (21) with objective value strictly less than 1. In our implementation, we solve this integer program for all $t \in T$.

To separate the cut-set-cover inequalities for a given cover I^t, we create an artificial sink node l and connect the nodes in I^t to l. We then compute a maximum r-l flow in the graph $G(S \cup \{l\}, A_S \cup I^t \times \{l\})$ with capacities $\hat{x}^t$ for the arcs in A_S and capacity 1 for the artificial arcs in $I^t \times \{l\}$. If the maximum flow value is less than 1, we add the violated cut-set-cover inequality

$$\sum_{(u,v) \in \Gamma(r,l)} x_{uv}^t \geq 1 \tag{22}$$

where $\Gamma(r,l)$ is the arc set of a corresponding minimum cut.

4 Experiments

Benchmark Instances

In Gollowitzer and Ljubić [9], a set of instances for connected facility location was generated by combining a set of benchmark instances for the Uncapacitated Facility location (UFL) problem from the UflLib [1] with instances of the Steiner tree problem (STP) from the OR-library [4]. The ConFL input graphs are generated in the following way: first f nodes of the STP instance are selected as potential facility locations (where f denotes the number of facilities in the corresponding UFL instance), and the node with index 1 is selected as the root. The number of facilities, the number of customers, opening costs and assignment costs are provided in UFL files. STP files provide edge-costs and additional Steiner nodes.

We consider a set of 32 instances obtained by combining four UFL instances `mp1,mp2` and `mq1,mq2` (of the size 200×200 and 300×300, respectively) with eight STP instances $\{$`c,d`$\}$`n`, for $n \in \{5, 10, 15, 20\}$. These instances define the core networks with between 500 and 1000 nodes and with up to 25,000 edges.

We extend these instances to include demands and time periods. We generate demands uniformly between 20 and 40 for each customer and we consider a time horizon $\mathcal{T} = 5$. In the test instances generated in [9], the facility set F and customers R induce a complete bipartite graph. We desire a more sparse setting for our demand satisfaction and the cover set inequalities. Therefore, we only considered the connections of the first 20 closest facilities for each customer. Such obtained instances contain up to 1300 nodes and 45,000 edges. Finally, the minimum coverage required for time period t is defined as

$$D^t = \frac{\sum_{j \in R} d_j}{1.25(T - t)} \text{ for } t \in \{0, 1, 2, 3, 4\} \text{ and } T = 5.$$

The experiments were performed on an Intel Core2 Quad 2.66 Ghz systems with 2GB RAM. Each run was carried out on a single processor.

4.1 Branch and Cut Implementation

To test the effectiveness of the presented formulations and inequalities, we implemented a branch and cut algorithm using CPLEX 12.2 and Python API, a commercial integer programming solver with a branch and cut framework.

The integer linear programs initially contain all variables and the constraints (1) – (6). The cut inequalities (7) and (10), the cover inequalities (11), and the cut-set-cover inequalities (12) are applied in a standard cutting plane approach, iteratively adding those inequalities that are violated by the current fractional solution.

We add all indegree constraints (15) to the initial LP formulation. We generate a cut pool with all the in-arc inequalities (13) and (14), which are added at the root node if they are violated. We then call the maximum flow separation routine that generates the inequalities (7). This separation consists of randomly selecting 50 terminals at every time period and generating the violated cuts. We restrict the number of calls to the separation routine at every node by 10, to enable branching and avoid multiple calls to the separation routines. In addition to the above, the separation routine is called at node depth of multiples of 10 and at every occasion an incumbent is rejected. The intuition behind this scheme is that it would provide us with a balance between the time spent in generating the cuts and branching, as branching helps us reducing the search space (due to the priority strategies described below). The enhanced cuts and customer cuts are combined in the same separation routine. Each test run was limited to 2000 CPU seconds and the optimality gap at this point of time is reported in the results.

Branching:

The assignment variables x^t_{ij}, when branched (set to 0 or 1), does not affect the search space as much as the facility variables z^t_i. So we give them the highest priority in the branching. This was also observed in [12], but unlike the connected facility location problem, in our incremental version of the problem, we also have uncertainty in determining the set of customers to be served at each time period. So, we provide them with the next highest priority in branching.

Separation routine:

We observed that the cuts generated by the maximum flow algorithm when the root is treated as source tend to generate cuts that are closer to the root node and there will be edges repeated in the various minimum cuts generated for various terminals. In order to avoid this, we treat the root as the sink and the facilities as the source. This was appropriately captured in the primal heuristic and the in-arc inequalities as well. We also perform nested cuts, wherein we resolve maximum flow for the same facility by setting the capacity of the edges in current minimum cutset to 1. The cover (11) and cut-set-cover inequalities (12) rely on solving an integer program at every call of the separation routine, which is run for every time period. The integer program terminates if the elapsed running time is over 100 seconds or if the objective value drops below 1. We use this exact separation to test the impact of these inequalities on the lower bound and in the event they are useful they will be replaced with heuristic methods similar to the techniques used to generate cover inequalities for knapsack constraints.

Primal Heuristics:

We also implemented and tested a naive primal heuristic. After our initial runs we decided to turn off the CPLEX heuristics as this was leading to poor performance. The primal heuristic rounds up all the z variables that indicate the usage of a facility as well as the y variables, which indicate the service to a customer. We run a minimum cost flow algorithm with a linear cost estimator with the open facilities (rounded up values) as sinks and the root node as source to generate our Steiner tree.

4.2 Results

Our preliminary computational study has shown that CUT_R formulation is not competitive against the CUT_F model, due to the size of the support graph and the large number of cut-set inequalities that need to be separated. This is also consistent with the results obtained by Gollowitzer and Ljubić [9] for the single-period ConFL.

Therefore, in our computational study, we compared the performance of the following two branch and cut settings:

- CUT_F formulation,
- CUT_F+ formulation extended by cover inequalities (11) and cut-set-cover inequalities (12).

For each of the two settings, we report on the following values given in Table 1: the overall percentage gap obtained after the time limit of 2000 seconds calculated as Gap $= (UB - LB)/LB$, where UB is the best obtained upper bound, and LB is the global lower bound; the number of all constraints separated throughout the execution of the algorithm, denoted by “Cuts”; the number of branch and bound nodes, denoted by “B&B”.

Table 1 Comparison of two branch and cut settings: plain CUT_F model vs. CUT_F extended by cover and cut-set-cover inequalities.

		CUT_F				CUT_F + (11) + (12)			
Instance	best LB	best UB	Gap[%]	cuts	B&B	best UB	Gap[%]	cuts	B&B
c10-mp1	164,136	166,691	1.53	2103	170	166,670	1.52	695	104
c10-mp2	160,278	168,347	4.79	1781	99	168,258	4.74	297	20
c10-mq1	346,866	386,409	10.23	457	11	386,054	10.15	161	0
c10-mq2	348,929	387,501	9.95	559	14	387,088	9.86	136	0
c15-mp1	165,004	166,900	1.14	1539	107	166,985	1.19	212	15
c15-mp2	161,333	168,487	4.25	1508	56	168,525	4.27	299	16
c15-mq1	352,583	386,520	8.78	567	15	386,326	8.73	140	0
c15-mq2	348,640	387,614	10.05	422	10	387,452	10.02	138	0
c20-mp1	155,919	167,227	6.76	334	6	167,184	6.74	121	0
c20-mp2	154,157	168,658	8.60	360	3	168,656	8.60	137	0
c20-mq1	349,075	386,640	9.72	298	0	386,540	9.69	55	0
c20-mq2	348,628	387,792	10.10	311	0	387,681	10.07	60	0
c5-mp1	162,521	166,466	2.37	1927	190	166,320	2.28	590	75
c5-mp2	158,230	168,042	5.84	1630	45	167,892	5.76	347	19
c5-mq1	346,924	386,236	10.18	869	15	385,491	10.00	133	0
c5-mq2	348,453	387,330	10.04	817	15	386,744	9.90	143	0
d10-mp1	164,160	166,945	1.67	2008	45	166,706	1.53	126	10
d10-mp2	155,885	168,381	7.42	1902	21	168,167	7.30	291	15
d10-mq1	342,278	386,234	11.38	508	6	385,844	11.29	121	0
d10-mq2	348,389	387,584	10.11	642	9	387,062	9.99	110	0
d15-mp1	158,402	167,103	5.21	813	15	167,005	5.15	176	6
d15-mp2	158,835	168,398	5.68	1103	15	168,468	5.72	193	1
d15-mq1	346,494	386,579	10.37	407	0	386,258	10.29	95	0
d15-mq2	346,129	387,683	10.72	415	6	387,458	10.67	118	0
d20-mp1	155,821	167,168	6.79	256	0	167,208	6.81	139	0
d20-mp2	154,141	168,661	8.61	291	0	168,675	8.62	175	0
d20-mq1	348,738	386,621	9.80	180	0	386,580	9.79	62	0
d20-mq2	348,365	387,801	10.17	135	0	387,772	10.16	16	0
d5-mp1	163,074	166,680	2.16	2666	50	166,216	1.89	1387	225
d5-mp2	163,182	167,967	2.85	2495	104	167,634	2.66	1172	165
d5-mq1	346,120	386,141	10.36	1230	15	385,396	10.19	185	0
d5-mp1	344,089	387,304	11.16	1541	15	386,517	10.98	166	0

Comparing the number of inserted cuts by the two approaches, we observe that the inclusion of coverage-related cuts (i.e., (11) and (12)) reduces the overall number of cuts generated within a given time limit. This can easily be explained by the large separation times needed to solve the integer program (18)-(21). Despite the reduced number of separated inequalities, in 27 out of 32 instances we obtained reduced duality gaps when the coverage-related inequalities were used. This indicates the strength of the coverage-related cuts, but also the trade-off between their strength and their separation time.

We also observe that due to the branching and separation strategies that we choose, there is no direct correlation between the usage of coverage-related constraints and the number of branch and bound nodes.

5 Conclusions

In this work we introduce a new combinatorial optimization problem that models the design of fiber-to-the-curb networks over time. The problem is a multi-period version of the connected facility location problem that has been intensively studied in the literature in the last decade. Besides two mixed integer programming models, we also introduce two new families of valid inequalities derived from the incremental coverage constraints over time. We provide separation algorithms needed to detect the new coverage-related inequalities within a cutting plane framework. The problem is then solved by means of a branch and cut algorithm that makes use of the cut-set inequalities and the new coverage-related constraints. In the (preliminary) computational study we show that the new inequalities are useful for small and/or sparser instances, where the obtained duality gaps can be significantly reduced. For larger instances, it turns out that the there is a trade-off between the separation time of the coverage-related family of inequalities and the obtained improvement of the quality of lower bounds.

In a future work we intend to investigate the performance of the branch and cut algorithm on a larger set of benchmark instances. We also want to study the influence of the minimum coverage rate D^t to the quality of lower bounds of the proposed models. Further problem-related inequalities will be derived as well. One of the problems addressed by our computational results is the computational inefficiency of the integer program needed to separate the coverage-related inequalities. To overcome this problem, one needs to develop more efficient exact or heuristic approaches for the separation. Finally, it will be also interesting to compare decomposition based approaches (e.g., Lagrangian or Benders decomposition) with the proposed branch and cut framework.

References

[1] UflLib. URL http://www.mpi-inf.mpg.de/departments/d1/projects/benchmarks/UflLib/.

[2] M. Albareda-Sambola, E. Fernández, Y. Hinojosa, and J. Puerto. The multi-period incremental service facility location problem. *Computers & Operations Research*, 36:1356–1375, May 2009.

[3] M. G. Bardossy and S. Raghavan. Dual-Based Local Search for the Connected Facility Location and Related Problems. *INFORMS Journal on Computing*, 22(4):584–602, 2010.

[4] J. E. Beasley. OR-library. URL http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html.

[5] D. Bienstock, O. Raskina, I. Saniee, and Q. Wang. Combined network design and multiperiod pricing: Modeling, solution techniques, and computation. *Operations Research*, 54(2):261–276, 2006.

[6] N. Christofides and P. Brooker. Optimal expansion of an existing network. *Mathematical Programming*, 6:197–211, 1974.

[7] P. J. Doulliez and M. R. Rao. Optimal network capacity planning: A shortest-path scheme. *Operations Research*, 23(4):810–818, 1975.

[8] F. Eisenbrand, F. Grandoni, T. Rothvoß, and G. Schäfer. Connected facility location via random facility sampling and core detouring. *Journal of Computer and System Sciences*, 76:709–726, 2010.

[9] S. Gollowitzer and I. Ljubić. MIP models for connected facility location: A theoretical and computational study. *Computers & Operations Research*, 38(2):435–449, 2011.

[10] T. Koch and A. Martin. Solving Steiner tree problems in graphs to optimality. *Networks*, 32:207232, 1998.

[11] B. Lardeux and D. Nace. Multi-period network design of optical transmission networks. In *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, pages 935–940, 2007.

[12] I. Ljubić. *A hybrid VNS for connected facility location*, volume 4771 of *Lecture Notes in Computer Science*, pages 157–169. Springer, 2007.

[13] S. H. Owen and M. S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111(3):423–447, 1998.

[14] U. H. Suhl and H. Hilbert. A branch-and-cut algorithm for solving generalized multiperiod Steiner problems in graphs. *Networks*, 31(4): 273–282, 1998.

[15] A. Tomazic and I. Ljubić. A GRASP algorithm for the connected facility location problem. In *Proceedings of 2008 International Symposium on Applications and the Internet (SAINT)*, pages 257–260, 2008.

[16] S. V. Ukkusuri and G. Patil. Multi-period transportation network design under demand uncertainty. *Transportation Research Part B: Methodological*, 43(6):625 – 642, 2009.

[17] N. Zadeh. On building minimum cost communication networks over time. *Networks*, 4(1):19–34, 1974.