A branch-and-cut-and-price approach for a Two-level Hierarchical Location Problem

Bernardetta Addis* Giuliana Carello* Alberto Ceselli◇

*Dipartimento di Elettronica e Informazione, Politecnico di Milano
Via Ponzio 34, 20133 Milano, Italia

◇Dipartimento di Tecnologie dell'Informazione, Università degli Studi di Milano
via Bramante 65, 26013 Crema, Italia

Abstract

In many telecommunication networks a given set of client nodes must be served by different sets of facilities, which must be located and dimensioned in designing the network. We provide a compact and an extended formulation for that problem and we design an exact branch-and-cut-and-price optimization algorithm. We test our approach on a set of instances derived from the facility location literature.

Keywords: *location, column generation, branch-and-cut-and-price, telecommunications*

1 Introduction

In many telecommunication networks different sets of network facilities, provided with different equipments and carrying out different tasks, are needed. In designing such networks the set of client nodes is usually given, while the facilities of different kinds must be located and dimensioned. For instance in IP networks *access* nodes, which are origin and destination of traffic demands, must be connected to the backbone nodes, the *core* nodes, through *edge* nodes. The same happens in the fiber-to-the-home network where nodes representing clients must be connected to *cabinet* nodes which collect traffic and send it to *central offices*. The design of such IP networks has been recently heuristically tackled in [1], while to the best of our knowledge, no optimization approach has been proposed so far for fiber-to-the-home networks. The optimal design of networks with the above structure can be seen as a Facility Location Problem in which two different sets of facilities are considered. Facilities in the same sets are similar and are said to belong to the same level. As in the Single Source Facility Location Problem, each client must be assigned to one facility of the mid level set. Besides, each mid level facility must be assigned to one of the high level set. The network turns out to be a star-star one. Similar network location problems have already been considered in the literature. In [2] a Two-level Simple Plant Location Problem with uncapacitated facilities is tackled. In [3] the Two-echelon, Single Source Capacitated Facility Location Problem is considered: each client must be served by one and only one facility which, in turn, is assigned to one and only one depot. Depots are not capacitated. A branch and bound is proposed for the problem, based on Lagrangian Relaxation. In [4] the location of two different types of nodes, concentrators and routers, is considered. Both concentrators and routers are capacitated. Each terminal in the network must be assigned to one and only one concentrator, which in turn must be assigned to one router. Besides installation, allocation and operational costs related to concentrators and routers are considered, which depend linearly on the amount of capacity used. Both lower and upper bounds are proposed for the problem. The lower bound is based on lagrangian relaxation, while the upper bound is provided through a tabu search heuristic.

In all the previous papers only location decisions are considered; however, in order to deal with practical applications location and dimensioning have to be optimized simultaneously.

Thus, in this paper we consider the problem of both locating and dimensioning capacitated facilities in a star-star network. We denote the problem as the *Two-level Hierarchical Capacitated Facility Location Problem* (TLHCFLP). TLHCFLP is *NP*-hard, as it generalizes classical Facility Location Problem. We provide a compact and an extended formulation for the TLHCFLP. We design an exact optimization algorithm, which exploits the extended formulation within a branch-and-cut-and-price approach. We propose to use Very Large Scale Neighborhood search techniques to obtain good integer solutions. We test our approach on a set of instances derived from the literature on the Single-Source Capacitated Facility Location Problem.

The paper is organized as follows: in Section 2 the problem is described and the formulations, both compact and extended, are presented. The proposed approach is described in Section 3. Finally computational results in Section 4 and remarks in Section 5 end the paper.

2 Problem description and formulations

In TLHCFLP each client node $i \in \mathcal{I}$ must be connected to a mid level facility which in turn must be connected to a high level facility. Both kinds of facilities must be located: the sets of candidates sites are denoted with $\mathcal{J}$ and $\mathcal{K}$, respectively. Each mid level facility must be equipped with a device chosen in a set $\mathcal{T}$ of available devices.

Placing a mid level facility in a site $j \in \mathcal{J}$ and a high level facility in a site $k \in \mathcal{K}$ implies installation costs c_j and g_k, respectively. Each client $i \in \mathcal{I}$ has a demand a_i. Assigning client i to a mid level facility located in $j \in \mathcal{J}$ and a mid level facility located in j to a high level facility located in $k \in \mathcal{K}$ implies connection costs d_{ij} and l_{jk}, respectively. Each device $t \in \mathcal{T}$ has a capacity b_t and an installation cost f_t; the b_t coefficient represents also the demand to be served by a high level facility to a mid level facility equipped with device t. To model economies of scale of practical applications we suppose that devices identified by higher indices have higher capacity b_t, higher cost f_t and that the cost per unit of capacity f_t/b_t is decreasing with respect to the capacity. Dimensioning of high level facilities is not considered: each high level facility provides a capacity B.

In the TLHCFL four decisions have to be taken: to open or not a high level facility in each $k \in \mathcal{K}$ (binary variables z_k), to open or not a facility equipped with device $t \in \mathcal{T}$ in each $j \in \mathcal{J}$ (binary variables y_{jt}), to assign or not a mid level facility in $j \in \mathcal{J}$ equipped with device $t \in \mathcal{T}$ to a high level facility in $k \in \mathcal{K}$ (binary variables w_{jtk}), to assign or not a client $i \in \mathcal{I}$ to a mid level facility located in $j \in \mathcal{J}$ (binary variables x_{ij}).

An ILP model for TLHCFL is the following:

$$\min \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} d_{ij} x_{ij} + \sum_{j \in \mathcal{J}} \sum_{t \in \mathcal{T}} \sum_{k \in \mathcal{K}} l_{jk} w_{jtk} + \sum_{j \in \mathcal{J}} \sum_{t \in \mathcal{T}} (c_j + f_t) y_{jt} + \sum_{k \in \mathcal{K}} g_k z_k \tag{1}$$

$$\sum_{j \in \mathcal{J}} x_{ij} \geq 1, \forall\ i \in \mathcal{I} \tag{2}$$

$$\sum_{i \in \mathcal{I}} a_i x_{ij} \leq \sum_{t \in \mathcal{T}} b_t y_{jt}, \forall\ j \in \mathcal{J} \tag{3}$$

$$\sum_{t \in \mathcal{T}} y_{jt} \leq 1, \quad \forall j \in \mathcal{J} \tag{4}$$

$$\sum_{k \in \mathcal{K}} w_{jtk} \geq y_{jt}, \forall\ j \in \mathcal{J}, \forall\ t \in \mathcal{T} \tag{5}$$

$$\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} b_t w_{jtk} \leq B z_k, \forall\ k \in \mathcal{K} \tag{6}$$

$$x_{ij}, w_{jtk}, y_{jt}, z_k \in \{0,1\} \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T} \tag{7}$$

The objective function (1) aims at minimizing the total installation and connection cost. Constraints (2) force each client to be assigned to a mid level facility, while constraints (5) force each open mid level facility to be assigned to a high level facility. Inequalities (3) guarantee that each mid level facility is equipped with a device providing enough capacity to serve the demand of the connected clients, while inequalities (4) guarantee that each mid level facility is equipped with at most one device. Finally, inequalities (6) guarantee that each high level facility has enough capacity to serve its assigned mid level facilities. The model is completed with integrality conditions (7).

Model (1) − (7) has a polynomial number of variables and constraints, and is therefore suitable to be optimized by general purpose ILP solvers. Instead, we propose to formulate the problem with an extended model involving an exponential number of variables, and solve it by ad-hoc integer programming techniques. In particular, our formulation has two parts: an extended part which models the mid level location problem, and a discretized part which models the high level location problem.

In the extended part we consider, for each site $j \in \mathcal{J}$ and for each device $t \in \mathcal{T}$, each subset of clients which can be assigned to a facility in site j equipped with device t without violating constraints (3); we call these subsets *clusters* of clients, and we indicate as $\mathcal{S}_{jt}$ the set of all clusters which refer to site j and device t. Moreover we describe with coefficients u_{is} the incidence vector of each set s, that is u_{is} is one if i is included in cluster s and zero otherwise. The cost of a cluster $s \in \mathcal{S}_{jt}$ is given by $C_s = c_j + f_t + \sum_{i\in s} d_{ij}$. For all $j \in \mathcal{J}$, for all $t \in \mathcal{T}$ and for all $s \in \mathcal{S}_{jt}$ we introduce a binary variable v_s, which takes value one if a mid level facility equipped with device t is built in j, and cluster s is assigned to that facility, zero otherwise. These v variables replace the x and y variables, while w and z variables keep the same role as above.

In the discretized part we refine our model by exploiting a technique introduced in [10]. Let Q be the set of values which can actually represent the demand of a set of mid level facilities assigned to the same high level facility. We substitute each variable z_k with a set of binary variables z_k^q forall $q \in Q$; each variable z_k^q takes value 1 if a high level facility is built in k and serves exactly q demand units, 0 otherwise. Therefore $z_k = \sum_{q\in Q} z_k^q$ and $\sum_{q\in Q} z_k^q \leq 1$. The whole problem can be reformulated as follows:

$$\min \sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}_{jt}} C_s v_s + \sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}} l_{jk} w_{jtk} + \sum_{k\in\mathcal{K}} g_k z_k \tag{8}$$

$$\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}_{jt}} u_{is} v_s \geq 1, \forall\ i \in \mathcal{I} \tag{9}$$

$$\sum_{k\in\mathcal{K}} w_{jtk} - \sum_{s\in S_{jt}} v_s \geq 0, \forall j \in \mathcal{J}, \forall t \in \mathcal{T} \tag{10}$$

$$\sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}} w_{jtk} \leq 1, \forall j \in \mathcal{J} \tag{11}$$

$$\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} b_t w_{jtk} = \sum_{q\in Q} q z_k, \forall k \in \mathcal{K} \tag{12}$$

$$\sum_{q\in Q} z_k \leq 1, \forall k \in \mathcal{K} \tag{13}$$

$$v_s, w_{jtk}, z_k^q \in \{0,1\} \quad \forall j \in \mathcal{J}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \forall s \in \mathcal{S}_{jt}, \forall q \in Q \tag{14}$$

As for the compact model, the objective function (8) aims at minimizing the total installation and connection cost. Inequalities (9) guarantee that each client is included in a selected cluster. Constraints (10) force each open mid level facility to be assigned to a high level facility. Constraints (11) and (12) keep the same meaning as (4) and (6), while constraints (13) are introduced during the discretization step.

It is not hard to show that, when integrality conditions are relaxed, model (8) – (14) corresponds to the Dantzig-Wolfe reformulation of model (1) – (7) when constraints (3) are convexified, and the high level part of the model is discretized.

Moreover, let N be a lower bound on the number of demand units which have to be provided by mid level facilities; N can be computed by simply dividing the demand of all clients by the capacity of the smallest device. Then

$$\sum_{k\in K}\sum_{q\in Q} q z_k^q \geq N.$$

Hence, for all $p \in Q$, we can divide both sides of the inequality by p, round up to the nearest integer the left-hand-side coefficients, still obtaining a valid inequality. Since in this way the left-hand-side never takes fractional values, we can round also the right-hand-side term up to the nearest integer, obtaining the following set of inequalities:

$$\sum_{k\in K}\sum_{q\in Q} \lceil \frac{q}{p} \rceil z_k^q \geq \lceil \frac{N}{p} \rceil.$$

Besides including these $|Q|$ inequalities, we strenghten the final model with the following constraints: for all $j \in J$ and $k \in K$

$$w_{jkt} \leq \sum_{q\in Q} z_k^q$$

that is, when no high level facility is built in k, no mid level facility can be assigned to k.

3 Branch and price and cut

We indicate as master problem (MP) the continuous relaxation of model (8) – (14), and we compute this relaxation to obtain a dual bound for the problem. Since this model contains an exponential number of variables in problem dimension, we solve this relaxation using column generation techniques. Our method works as follows.

Initialization. We consider a restricted master problem (RMP) including all the constraints, but only z and w variables and columns encoding clusters containing just one client. We enrich this initial RMP with the solutions obtained by three runs of a simple greedy heuristic, which works as follows. First, mid level facilities are opened in a randomly selected set of sites; the choice on which device to install at each facility is delayed to subsequent steps. Then, each client is assigned to the nearest facility. Finally, the cheapest device providing enough capacity to serve the sum of the demands of the assigned clients is installed on each facility. Although a complete integer solution would require to assign mid level facilities to high level facilites, the procedure above is enough to generate good initial clusters.

Column generation. We iteratively solve the RMP and exploit the optimal dual solution to search for new columns having negative reduced cost; the problem of finding such columns is called the pricing problem. Let λ_i and μ_{jt} be the dual variables associated respectively to each constraint of the set (9) and to each constraint of the set (10). For each $j \in J$ and for each $t \in T$, consider the cluster s associated to a value τ_{jt} computed as follows:

$$\tau_{jt} = \min_{s\in S_{jt}} \{c_j + f_t - \sum_{i\in\mathcal{I}} (\lambda_i - d_{ij})\, u_{is} + \mu_{jt}, \text{s.t.} \sum_{i\in\mathcal{I}} a_i u_{is} \leq b_t\}$$

We obtain each τ_{jt} value by solving a 0-1 knapsack problem using an ad-hoc algorithm [8]. The problem of finding the most negative reduced cost column consists in finding the column associated to the value $\sigma = \min_{j\in J, t\in T}\{\tau_{jt}\}$. If $\sigma \geq 0$, the current RMP is optimal for MP, and therefore gives a valid dual bound for the whole problem. Otherwise, all the columns having $\tau_{jt} \leq 0$, which are encoded by the coefficients u_{is}, are inserted into the RMP; the new RMP is solved, a new vector of dual variables is obtained and the process is iterated.

Stabilization. Column generation methods are known to suffer stability problems, yielding poor convergence performace. In order to overcome such problems we adapted the stabilization method described in [7], obtaining good results with a modest additional computational effort.

Cut generation. When column generation is over, we search for valid inequalities which are violated by the current fractional solution. In fact, while cuts on the v column variables are difficult to handle in a branch-and-price-and-cut scheme, cuts involving w and z variables can improve the quality of the bound without affecting the structure of the pricing problem. We search for such violated inequalities in a pool of general purpose cuts, as sketched out in Section 4 Whenever new cuts are found, the column generation process is repeated to ensure the dual bound to be valid.

Such price-and-cut loop is repeated until neither new cuts nor new columns are generated.

Branching. Indeed, the MP optimal solution might not be integral, even at the end of the price-and-cut loop. In this case we first compute the fractional solution of the compact formulation (1) – (7) corresponding to the current fractional solution of MP by setting

$$x_{ij} = \sum_{t \in \mathcal{T}} \sum_{s \in \mathcal{S}_{jt}} u_{is} v_s \quad \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}$$

$$y_{jt} = \sum_{s \in \mathcal{S}_{jt}} u_{is} v_s \quad \forall j \in \mathcal{J},\ \forall t \in \mathcal{T}.$$

and keeping the z and w variables at their value. Then we complete the computation by exploring a branching tree, which is composed by five levels.

At the first level we select the most fractional z variable and perform binary branching by fixing its value to 1 in one branch and to 0 in the other branch.

When all the z variables take integer values, we branch on the (4) constraints: we select the index $\bar{j}$ for which $\sum_{t \in \mathcal{T}} y_{jt}$ is nearest to 0.5. In fact, in an integer solution $\sum_{t \in \mathcal{T}} y_{jt}$ is 1 if a facility is placed in j or 0 otherwise. We impose in one branch that $\sum_{t \in \mathcal{T}} y_{\bar{j}t} = 1$ and in the other branch that $\sum_{t \in \mathcal{T}} y_{\bar{j}t} = 0$. Both conditions can simply be enforced by fixing to zero either slack variables or columns in the MP; in the second branch we remove columns related to $\bar{j}$ from the RMP and generate no more column of each set $\mathcal{S}_{\bar{j}t}$.

When all the values $\sum_{t \in \mathcal{T}} y_{jt}$ are 0 or 1, we select the index $\bar{j}$ for which the largest number of $y_{\bar{j}t}$ variables have value greater than 0. That is the site in which the assigned demand is split between the largest number of possible devices. Given a capacity b_k for which half of the fractional $y_{\bar{j}t}$ values have $b_t \leq b_k$, we impose $\sum_{t \in \mathcal{T} | b_t \leq b_k} y_{\bar{j}t} = 0$ in one branch and $\sum_{t \in \mathcal{T} | b_t > b_k} y_{\bar{j}r} = 0$ in the other branch. As in the previous level, these constraints are handled by just removing suitable columns from the RMP and not generating any new column in the sets $\mathcal{S}_{\bar{j}t}$.

When also each y_{jt} variable takes integer value, we perform branching on the $\sum_{j \in \mathcal{J}} x_{ij} = 1$ constraint by searching for the client $\bar{i}$ which is (fractionally) assigned to the largest number of facilities. We split $\mathcal{J}$ in two sets $\mathcal{J}^l$ and $\mathcal{J}^r$, each containing at least one site j for which $x_{\bar{i}j} > 0$, and we create two branches imposing respectively $\sum_{j \in \mathcal{J}^l} x_{\bar{i}j} = 0$ and $\sum_{j \in \mathcal{J}^r} x_{\bar{i}j} = 0$. As before, these conditions are easy to enforce by just removing items in the pricing subproblems.

Finally, when also these variables take integer value, we perform branching on the $\sum_{k \in \mathcal{K}} w_{jtk} \geq y_{jt}$ constraints by searching for the mid level facility $\bar{j}$ which is (fractionally) assigned to the largest number of high level facilities, and we create two branches as those applied in level four.

We remark that these branching decisions requires no changes in the structure of the pricing problem, which remains a simple KP through all the enumeration process.

4 Computational results

We implemented a preliminary prototype of our algorithm (BCP) in C++, using SCIP [6] as a framework for building branch-and-price-and-cut algorithms, CPLEX 11.0 dual simplex solver for solving LP subproblems and an adaptation of MINKNAP algorithm [8] for solving the KP subproblems. For this prototype version, the general purpose cut generation of SCIP was turned on, but no additional valid inequality had been experimented.

We considered CPLEX 11.0 ILP solver (CPX), with default parameter settings, as a benchmark method. In fact, CPX relies on a state-of-the-art branch-and-cut method, which includes general purpose cut generation and primal heuristics, and which proved to be effective in solving many capacitated location problems [5]. In [1], the authors test their heuristics using instances involving up to 75 clients, 10 candidate mid level facilities and 5 candidate high level facilities. Instead, in order to test our method also on larger instances, we considered a benchmark of 72 instances adapted from those in the literature [5], in which the number of clients range from 50 to 200, the number of facilities from 10 to 50 and the number of high level facilities from 10 to 50. We considered 5 types of device to be available at each node.

Our experiments ran on a Centrino Core2 3 GHz workstation with 2GB of RAM.

In a first set of experiments we compared the performances of CPX and BCP in providing primal and dual bounds at the root node [1].

The column generation method allows to find dual bounds which are tighter than CPX in all but 7 instances. On the average, BCP gives dual bounds which are about 5.41% tighter than CPX. On the other hand, BCP requires an average CPU time of 25.61s to complete the computation, against an average CPU time of 7.78s needed by CPX; furthermore, it provides worse primal bounds on almost all the instances.

In a second set of experiments we let CPX and BCP optimize the whole problem, imposing a time limit of 7200s of CPU time to each computation. CPX was able to solve to optimality 43 of the instances in the dataset (that is about 60%), while BCP solved 63 of them (that is 87.5%). On the instances which both methods were able to solve to optimality BCP was about 4% faster than CPX. Instead, on the instances which neither BCP nor CPX could solve to optimality CPX had a gap between best found primal and dual bound which is on the average about 1% smaller; in fact, in some instances the primal solution found by CPX after two hours of computation is substantially better than that found by BCP.

In a third set of experiments we tried to improve the primal solutions obtained by BCP on the root node with a preliminary implementation of a Very Large Scale Neighborhood Search procedure (VLSN) adapted from [9]. On the root node the gap between the best know primal solution and the primal solution found by BCP is on the average 9.10%; the primal solutions found by CPX are on the average 3.04% worse than the best known, while the solutions found by VLSN are on the average 2.77% worse than the best known. Moreover the best known primal solution is actually found by VLSN on 32 of the dataset instances.

5 Conclusions

In this work we consider a two-level facility location problem, proposing two models. The first one includes a polynomial number of variables and constraints. The second one contains a column for each feasible cluster of clients, modeling the mid level location problem, and embeds a reformulation by discretization of the compact part of the previous model for the high level location problem. We present an optimization algorithm (BCP), in which columns corresponding to these clusters are generated, and the bound is strenghtened by adding cuts on the high level assignment and location variables. BCP is completed with a stabilization device and a five-level branching policy. BCP can handle the second model in an effective way, since the column generation process shows good convergence properties, and effective cuts can be generated on the compact part of the model.

We present preliminary computational results on a dataset of instances which is based on single-source location problems from the literature. We compare BCP with CPLEX 11.0 ILP solver (CPX). Given a time limit of two hours, BCP allows to solve a larger number of instances to proven optimality. Furthermore, on the root node BCP gives a better dual bound at the expense of an higher computational effort; on the other hand CPX is able to get better primal bounds.

[1] A table reporting the detailed results of these experiments is available at http://www.dti.unimi.it/ $\sim$ ceselli/tlcfl.html

Since the largest part of the CPU time is spent in solving LP subproblems, multiple pricing strategies can improve the performances of BCP, as well as the introduction of ad-hoc cuts on the compact part.

We are currently experimenting the effect of embedding a VLNS algorithm in BCP. First computational results are encouraging, as VLSN is able to find primal bounds on the root node which are substantially better than those of CPX.

References

[1] S. Chamberland. An Efficient Algorithm for Designing Reliable IP Networks with an Access/Edge/Core Hierarchical Structure. *Communication at Networks 2008*.

[2] P. Chardaire, J.-L. Lutton, and A. Sutter. Upper and lower bounds for the two-level simple plant location problem. *Annals of Operations Research*, 86:117–140, 1999.

[3] S. Tragantalerngsak, J. Holt, and M. Ronnqvist. An exact method for the two-echelon, single-source, capacitated facility location problem. *European Journal of Operational Research*, 123:473–489, 2000.

[4] A.A.V. Ignacio, V.J.M.F. Filho, and R.D. Galvao. Lower and upper bounds for a two-level hierarchical location problem in computer networks. *Computers and Operations Research*, 35:1982–1998, 2008.

[5] A. Ceselli, F. Liberatore, and G. Righini. A computational evaluation of a general branch-and-price framework for capacitated network location problems. *Annals of Operations Research*, available online, 2008.

[6] T. Achterberg. Constraint Integer Programming. *PhD Thesis, Technische Universität Berlin*, available online, 2007.

[7] E. Uchoa. A new stabilization method for column generation. *Column Generation Workshop*, conference presentation, Aussois, 2008.

[8] D. Pisinger. A minimal algorithm for the 0-1 knapsack problem. *Operations Research*, 45:758–767, 1997.

[9] R.K. Ahuja, J.B. Orlin, S. Pallottino, M.P. Scaparra, and M.G. Scutellà. A multi-exchange heuristic for the single source capacitated facility location problem. *Management Science*, 50(6):749:760, 2004.

[10] L. Gouveia, F. Saldanha-da-Gama. On the capacitated concentrator location problem: a reformulation by discretization. *Computers and Operations Research*, 33(5):1242–1258, 2006.