Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 467–472

ISBN 978-83-60810-51-4

Novel Presolving Techniques for the Connected Facility Location Problem

Alessandro Tomazic
Institute for Statistics and Mathematics
Vienna University of Economics and Business
Vienna, Austria
alessandro.tomazic@wu.ac.at

***Abstract*—We consider the connected facility location problem (ConFLP), a useful model in telecommunication network design. First we introduce the extended connected facility location problem which generalizes the ConFLP by allowing pre-opened and pre-fixed facilities. This new concept is advantageous for applying complex sequences of reduction tests. By such an analysis of the solution space we anticipate solution dependencies in favor of following optimization methods. Besides transferring existing techniques designed for the facility location problem, the Steiner tree problem and the group Steiner tree problem, specific new reduction methods are introduced. The presented concepts based on graph theoretic formulations are also of theoretical interest. Additionally, we propose an efficient self-adaptive presolving strategy based on test dependencies and test impacts respectively. A computational study shows that the number of edges could be reduced up to 85% and the number of nodes up to 36% respectively on instances from the literature.**

***Keywords*-connected facility location; presolving; network design; Steiner tree;**

I. Introduction

A. Motivation

THE *connected facility location problem* (ConFLP) is a highly useful model for the application to problems arising in the design of telecommunication networks. For instance the (partial) replacement of existing out-of-date copper based networks by modern fiber-optic cables can be handled as a ConFLP. We are given customers, that need to be connected to a central distributor by a tree-shaped network. In commonly used *Fiber-To-The-Curb* (FTTC) architectures, potential switching locations are given to which the customers may by connected by an existing copper infrastructure. Any choice of switch installations results in a set of terminals that have to be connected to the distributor using new fiber-optic technology. The practical objective is to minimize the overall installation costs for cables and switching devices.
The ConFLP is an NP-hard [1] optimization problem and therefore especially challenging in real world applications involving large instances. To ease the computational burden for algorithms to compute an optimal or heuristic solution, the application of problem presolving procedures is not just effective, but also unavoidable in many cases. The term *presolving* is used in turn which emphasizes the solution-oriented character of the applied techniques, compared to simple preprocessing methods. Such an analysis of the solution space may result in a remarkable reduction of the problem size. For certain instances no further methods need to be applied since the techniques used for presolving completely reduced them to trivial ones and therefore solves it to optimality. Exact and heuristical methods take advantage of the anticipation of preprocessable dependencies. Certainly a reduced number of provided variables is likely to accelerate the enumeration in exact branch & bound algorithms.

Partially supported by the Austrian Research Promotion Agency, FFG, within Bridge 2 programme (812973)

B. Contribution

We developed several presolving techniques for the ConFLP, transferred existing literature ideas for related problems and embedded these methods into a self-adaptive overall strategy. This algorithmical framework is based on problem and test specific reduction dependencies that we also present in this paper. In our studies we did not limit ourselves to problem reductions that necessarily result into a ConFLP instance again. Instead we generalized the ConFLP by the introduction of the extended connected facility location problem (EConFLP). This model is very convenient for transferring presolving information during the reduction process and allows the flexible integration of practical side constraints at the same time. The ideas are also of theoretical interest and variations may be considered for related problems. Computational results show that these methods can be effective.

C. Related work

The ConFLP model combines two classical problems in combinatorial optimization. On the one hand the *Steiner tree problem* (STP) asks for a tree of minimal edge costs that connects given *terminals* by optional use of *Steiner* nodes for the interlinkage [2]. On the other hand the ConFLP generalizes the well known *(Uncapacitated) facility location problem* (FLP). To solve the FLP selected facilities are installed at potential sites and each given customer is assigned to exactly one of them. The objective is to minimize the sum of the installation costs and the assignment costs. The ConFLP has been introduced by Karger and Minkoff [1] who gave the first approximation algorithm of constant factor. Solution

approaches for the ConFLP in the literature include exact MIP-based methods [3], greedy random adaptive search [4] and a dual based heuristic [5]. The design of effective reduction methods was already carried out for related network design problems. Introductory work on preprocessing techniques can ba found in [6]. The STP was considered by Duin et al. [7] and Polzin et al. ([8], [9]). Ferreira et al. [10] developed sophisticated tests for the reduction of instances of the *group Steiner tree problems* (GSTP). Ideas from these works serve as a starting point for some of our elaborations presented in this paper. More recently, the extensive application of presolving techniques by Letchford et al. [11] enabled the solution of peviously unsolved FLP instances to optimality. The authors combined complex lower and upper bounding procedures to an effective *aggressive preprocessing* scheme, that reduces instances sufficiently for MIP solvers.

D. Problem definition

Given an undirected connected graph $G = (V, E)$, a nontrivial partition (F, C) of V identifying *facilities* and *customers*, nonnegative *edge costs* c and nonnegative *opening costs* for the facilities, the *connected facility location problem* consists of finding a connected subgraph $G' = (V', E')$ of G, such that

i.) each customer is adjacent to exactly one facility in G',
ii.) the subgraph of G' induced by the set of facilities in V' is connected, and
iii.) the total cost, defined as the sum of the edge costs and the costs for opening facilities,

$$\sum_{v \in N_{V'}(C)} p_v \; + \; \sum_{e \in E'} c_e \quad ,$$

is minimized.

For any node $u \in V$, by $N_{V'}(u)$ we denote the set of its neighboring nodes in V', and for any subset $X \subseteq V$, we set $N_{V'}(X) = \cup_{u \in X} N_{V'}(u)$. Due to the problem definition above and the non-negativity of costs c and p there exists an optimal solution such that G' is a tree. Some facilities in F may be used as pure Steiner nodes, in which case no opening costs need to be paid for them. If in a solution a facility v is adjacent to a customer u, we call v an *open facility* and we say that u *is supplied* by v, or that u is *assigned to* v. The set of *potential facilities* $F_p = N_F(C)$ contains the nodes in F that allow facility installations.

II. The Extended ConFLP (EConFLP)

In our presolving studies we did not limit ourselves to problem reductions that necessarily result into a ConFLP instance. Instead we generalize the understanding of such an instance by the following properties:

- A facility might be labeled as *open facility*, i.e. it has to be open in a solution.
- A facility might be labeled as *network facility*, i.e. it has to be part of a solution, either open or not. We do not allow a facility to be open and a network facility at the same time.
- An edge connecting two facilities might be labeled as *network edge*, i.e. it has to be part of the facility network in a solution. This implies each of the two ends to be network facilities if not opened yet.
- Facilities may belong to *groups*. A group specifies facility sets in which at least one node belongs to an optimal solution.

In order to solve the resulting EConFLP, algorithms might need to be modified with respect to the additional restrictions. Alternatively a ConFLP instance could easily be obtained from an EConFLP instance. The corresponding transformation into the ConFLP looks as follows:

i.) For each open facility v, introduce an artificial customer $\tilde{v}$ and connect it to v with $c_{\tilde{v}v} = 0$. Herewith in any ConFLP solution the facility v will be opened.
ii.) For each network facility v introduce an auxiliary facility v' and connect it to v by an edge of zero cost where $p_{v'} = 0$. Then proceed with v' as for an open facility described above. Note that v' needs to be introduced since we do not necessarily open v.
iii.) For each group g, introduce an artificial customer v_g and connect it by zero cost edges to the nodes in g.

III. Presolving Techniques

Our presolving methodology tries to reduce the initial problem stepwise. To refer to the current reduced structures we use tilde, (e.g. $\tilde{F}_p$). Furthermore we will use the *distance function* d_{uv} which returns the length of a shortest path between two nodes u to v in the current graph with respect to c. To restrict the function to a subgraph induced by the nodes $X \subseteq V$ we write d^X_{uv}. For the set of open and network facilities, we also say *solution* or *fixed facilities*. Given two disjoint node sets $S_1, S_2 \subseteq V[G]$, we define a *minimal* $\{S_1, S_2\}$-*cut* as a partition $\{R_{S_1}, R_{S_2}\}$ of $V[G]$ such that $S_1 \subseteq R_{S_1}$, $S_2 \subseteq R_{S_2}$ and the number of edges connecting R_{S_1} and R_{S_2} is minimized. After computing a solution for the reduced problem, a corresponding solution for the original problem needs to be constructed. This is achieved by the successive reversion of the modifications in the current problem and the corresponding solution adaptations in the opposite order of reduction. Corresponding restoration rules can easily be derived from the reduction steps so that we do not elaborate them here.

A. Presolving the facility subgraph

As we already observed, once the set of open facilities is known, the problem reduces to the Steiner tree problem on the facility subgraph G_F. Therefore, the traditional reduction procedures for the STP, most of them originally proposed by Duin and Volgenant [7], can easily be extended to the ConFLP. We now show how to generalize these tests for the ConFLP: we apply them on the subgraph $\tilde{G}_{\tilde{F}}$, with network facilities and open facilities treated as terminal nodes.

1) Degree 1 and 2 facilities: Every facility $v \in \tilde{F}$ with $deg_{\tilde{V}}(v) = 1$ that is not a potential facility can be deleted. If v is an open or network facility, we additionally fix its neighbor (if not an open facility yet) as a network facility.
For a facility $v \in \tilde{F}$ with $deg_{\tilde{V}}(v) = 2$ that is not a potential facility, an open facility or network facility, we can apply the following: delete v and insert an edge connecting its two former neighbors. Set the edge cost to the sum of the costs of the removed edges. If this edge already exists, then set its weight to the minimum of the new cost and its original value. This is possible because either none of the two edges incident with v is part of an optimal solution or both.

2) Shortest paths: An edge $e = uv \in \tilde{E}_{\tilde{F}}$ that is not a network edge is dispensable if $c_e \geq d^{\tilde{F}}_{uv}$, because e can always be replaced by a shortest path connecting u and v without increasing the objective value.

3) Network edges: For a network edge $e = uv \in \tilde{E}_{\tilde{F}}$ with not both ends being potential suppliers, i.e. $\{u, v\} \not\subseteq \tilde{F}_p$, u and v can be contracted, say to z. Then z becomes a solution facility if neither u nor v were fixed in the solution before. If one of the ends was an open facility, then we assign its opening costs to z and open it. If u or v was a network facility, then z is added to the network facilities.

4) Nearest node: Consider a solution facility v and let $x = \text{argmin}_{u \in N_{\tilde{F}}(v)} c_{vu}$ and $z = \text{argmin}_{u \in N_{\tilde{F}}(v) \setminus \{x\}} c_{vu}$. We add vx to our solution as a network edge if a solution facility v' exists such that

$$c_{vx} + d^{\tilde{F}}_{xv'} \leq c_{vz} \quad .$$

Note that the nearest facility x may be a solution facility and in this case the test corresponds to the *adjacent solution facilities test* for x and v.

5) Node nearer to solution facility: An edge $uv \in \tilde{E}_{\tilde{F}}$ will not appear in an optimal solution if a solution facility $x \notin \{u, v\}$ exists such that

$$max(d^{\tilde{F}}_{xu}, d^{\tilde{F}}_{xv}) \leq c_{uv} \quad .$$

This is possible, because instead of using the edge uv in the network we could always connect the two ends to a solution facility without paying additional costs. Note that in the case of one end being a terminal, this test corresponds to the *nearest node test*.

6) Bottleneck degree m: We consider a facility v that is not a potential facility with $m = deg_{\tilde{F}}(v) \geq 3$. Such a facility will have either degree two or will not belong to an optimal solution if the following property holds:

$$\sum_{u \in N_{\tilde{F}}(v)} c_{vu} \geq ST(K, \tilde{G}_{\tilde{F}} \setminus \{v\}), \ \forall K \subseteq N_{\tilde{F}}(v), \ |K| \geq 3,$$

where $ST(K, H)$ denotes the cost of an optimal Steiner tree connecting subset K of terminals on the graph H. Since solving the Steiner Tree subproblem would be way to expensive, we just apply a heuristic, namely the well known *Shortest Path Heuristic* for the STP (see [2] or [4]). In order to do so, for each pair of neighbors of $x, z \in N_{\tilde{F}}(v)$, we either insert a new edge $e = xz$ (if it does not exist) and set its cost to $s = c_{xv} + c_{vz}$ or we update the current edge weight to $c_{xz} = \min(c_{xz}, s)$. In the worst case we pay the price of adding $\binom{m}{2} - m$ edges to the problem for a single facility deletion which explains why this test is just applicable for small values of m. In a dense facility network we conversely may not need to add any edges but we might want to change the cost structure, besides the facility deletion.

7) Adjacent solution facilities: Let two adjacent facilities u and v be part of a solution, either as network or open facilities. If $c_{vu} \leq c_{vx} \ \forall x \in F$ then the edge e_{vu} can be added to our solution as a network edge. To prove this, assume that the condition holds for v but an optimal solution S exists that does not contain vu. For connectivity reasons S contains a path in $\tilde{F}$ using an edge vx ($x \neq u$) from v to u. So S could be improved by using vu instead of vx what contradicts with the optimality of S.

8) Facility cuts: In [9], Polzin and Daneshmand present a decomposition concept for the STP based on the detection of *node separator* subsets of low cardinality, i.e. subsets of nodes whose removal separates the terminals of the graph $\tilde{G}_{\tilde{F}}$. We extend this concept to the set of *edge separators*, by introducing additional presolving steps for the newly detected groups induced by these cuts. For solution facilities t_1 and t_2 we compute a minimal t_1-t_2-cut (S, T) in $\tilde{G}_{\tilde{F}}$. If there is just a single edge e connecting S and T we can label e as a network edge, since it will belong to any feasible solution. Additionally we consider the induced node separator sets $Q_S = \{v \in S : N_{\tilde{F}}(v) \cap T \neq \emptyset\}$ and $Q_T = \{v \in T : N_{\tilde{F}}(v) \cap S \neq \emptyset\}$. We add Q_S and Q_T to the set of groups, but control these additions by a parameter that limits the size of added groups. Some of the presolving techniques for the group Steiner tree problem can be transferred and applied to the concept of *EConFLP* as defined above. Recall that, given a graph $\tilde{G}_{\tilde{F}}$ with non-negative edge costs, and a collection $\mathcal{R}$ of subsets of $\tilde{F}$, called *groups*, the GSTP is to find a minimum-cost subtree of $\tilde{G}_{\tilde{F}}$ that contains at least one node from each group $R \in \mathcal{R}$. We consider the groups $R \subset \tilde{F}$, that arise from different tests introduced within this paper. Apart from such groups, we can initially add a group R_u for each customer u consisting of its potential suppliers $N_{\tilde{F}}(u)$.

9) Node nearer to group: This test is a generalization of the *node nearer to terminal test* for the GSTP. An edge $e = uv \in \tilde{E}_{\tilde{F}}$ will not appear in an optimal solution if a group $R \not\supseteq \{u, v\}$ exists such that

$$max(d^{\tilde{F}}_{ru}, d^{\tilde{F}}_{rv}) \leq c_e \quad \forall r \in R \quad .$$

This is possible, because instead of using the edge e in the network we could always connect the two ends with any node in R without additional costs. The special case of considering groups of cardinality one leads to the *node nearer to terminal test*.

10) Group cuts: For two disjoint groups R_1 and R_2, we compute a minimal (R_1, R_2)-cut in $G_{\tilde{F}}$, say (S, T). If there is just a single edge connecting S and T we can label it as a network edge, since it will belong to any solution. Additionally we consider the induced node sparator sets $R_S = \{v \in S :$

$N(v) \cap T \neq \emptyset\}$ and $R_T = \{v \in T : N(v) \cap T \neq \emptyset\}$. We add R_S and R_T to the set of groups, but limit the total number of added groups. This generalizes the *facility cut test* since facilities fixed in a solution are just groups of cardinality one. Here we also mix the two types by allowing singleton groups.

B. Presolving the facility-customer subgraph

Adapting presolving tests for the facility location problem we get the following applying to the ConFLP.

1) Degree 1 customers: For a customer $u \in \tilde{C}$ with $deg_{\tilde{G}}(u) = 1$ we remove u and force the facility v to be part of the facility network as an open facility.

2) Customer domination: For two customers u and v with $N_{\tilde{V}}(u) \subset N_{\tilde{V}}(v)$ we can delete an edge $e = vx$ with $x \notin N_{\tilde{V}}(u)$ if $c_e \geq c_{vz}\ \forall z \in N_{\tilde{V}}(u)$. The reason for this is that at least one of the potential facilities of u will be opened and therefore even if facility x was already open it would be cheaper to let v be supplied by the facility supplying u.

3) Network facility: We consider the network facilities that are potential suppliers in order to open them or exclude alternative potential suppliers. Consider such a facility v and a potential customer u, i.e. $v \in N_{\tilde{V}}(u)$. We may delete an edge ux $(x \neq v)$ if $c_{xu} \geq c_{uv} + p_v$. So opening v and supplying u by v would be cheaper than supplying u by x, even if x was open.

4) Open facility: In the case that we have fixed a facility v to be supplying in our solution we should check whether we can exclude other facilities from being potential suppliers for its potential customers. So for a customer u adjacent to v an edge $e = ux$ $(x \neq v)$ can be deleted if $c_{vu} \leq c_e$.

C. Presolving the whole graph

Finally, in this section we propose tests that apply to the EConFLP concept, involving the whole graph $\tilde{G}$, considering solution facilities, network edges and groups as well.

1) Facility-customer distance: For a potential facility v we can delete a supply edge $e = vu$ $(u \in \tilde{C})$ if a path P_{vu} in $\tilde{G}_{\tilde{F}\cup\{u\}}$ from v to u not containing e exists such that $d'_{vu} \leq c_{vu}$, where d'_{vu} denotes the length of the path P_{vu} plus the opening costs of the potential facility z on that path if not opened yet. Thereby, the weights of network edges are discarded:

$$d'_{vu} = \begin{cases} \sum_{e \in P_{vu}} c_e + p_z & \text{z closed} \\ \sum_{e \in P_{vu}} c_e & \text{z open} \end{cases} \quad (zu \in P_{vu}) \quad .$$

This test checks if it would be cheaper to add the whole path to the solution and possibly open the facility z than using the edge vu.

2) Solution-facility-customer distance: In this test we consider a potential facility v and one of its supply edges $e = vu$ $(u \in \tilde{C})$. Let F_S be the set of current solution facilities. We delete e if for a $x \in F_S$ a path P_{ux} in $\tilde{G}_{\tilde{F}\cup\{u\}}$ from u to x not containing e exists such that $d'_{u,x} \leq c_{vu}$. Here d' is the function used in the *facility-customer distance test*. In contrast to the *facility-customer distance test* we just try to supply and connect the customer u to any facility in the existing facility subnetwork without exceeding certain supply edge costs. In the case that v is already open or a network facility, this test covers the *facility-customer distance test*.

3) Group-customer distance: We extend the *terminal-customer distance test* to groups. The additional requirement for a supply edge deletion is the existence of the mentioned path for all the nodes of at least one group.

4) Potential facility leaves: Let a potential facility v have $deg_{\tilde{F}}(v) = 1$. If v is part of an optimal solution, then - provided it is not the only facility to do so - surely its incident edge $vx \in \tilde{E}_{\tilde{F}}$ in the facility network will be used in this solution. In the case that x is not a potential facility itself, we can contract x and v and set the opening costs of the resulting potential facility as $p_v + c_e$. Otherwise if x is a potential facility and there exists a group not containing v or one other solution facility, then the opening cost of v can be increased by c_e to provoke the success of the *network facility test*.

5) Groups: We remove groups that contain solution facilities, since this is redundant information. Additionally we transform groups of cardinality 1 to network facilities. Multiple and empty groups are dynamically removed when removing a facility.

D. The overall presolving strategy

The proposed techniques are all of polynomial time complexity. Although the STP is well known to be in the class of NP-hard optimization problems we use a polynomial method to solve the problem heuristically. Therefore the overall procedure based on reduction success is also polynomial. The correctness follows from the validity of the single reductions and the transformation of the EConFLP into the ConFLP. However the effort for carrying out test on a problem instance varies. For instance a degree-testing can be done in significantly less time than running numerous minimal cut computations. Therefore we tried to minimize the number of tests that we identified as computationally more expensive in our experiments. The latter ones are mostly the tests that require solving a non-trivial subproblem, e.g. finding minimal cuts or multiple shortest paths. So we first apply the less elaborate tests as long as this results in a problem reduction. Afterwards we once run the more time consuming procedures and perform a restart if a problem modification was detected. Since we still observed many redundant test runs we focus on concrete test interactions to minimize unnecessary iterations. One can observe some complex relationships between different reduction types. Figure 1 describes the potential changes of the problem structure with respect to the tests performed. An arc depicts the potentially successful presolving tests after a certain problem modification. Figure 2 shows the impacts of the modifications of the input graph on specific presolving steps. An arc describes a problem modification consequence of the corresponding reduction type. Our idea was to make the testing scheme adaptive. Therefore a prioirity is assigned to each reduction method. Initially these values are chosen to set up a basic ordering. We simply enumerate the tests from based

on our computational study from easy to hard. So the tests are called according to their priority. Whenever a test did alter the current problem, we increment the priority of all the other tests that may depend on the performed problem modification. To keep the computational effort low, we still apply the 2-phase division that first works all the easy tests.

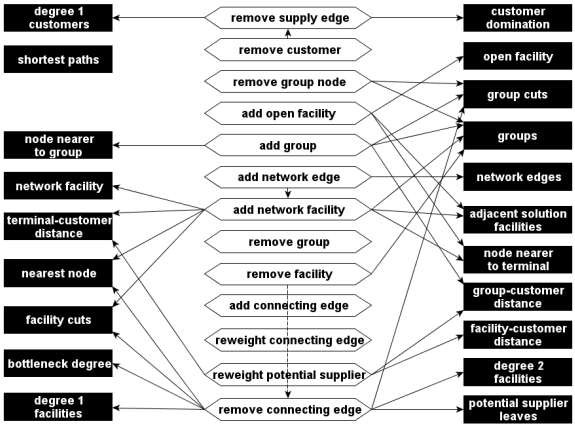


Fig. 1: Arc AM indicates that test A may result in a problem modification of type M.

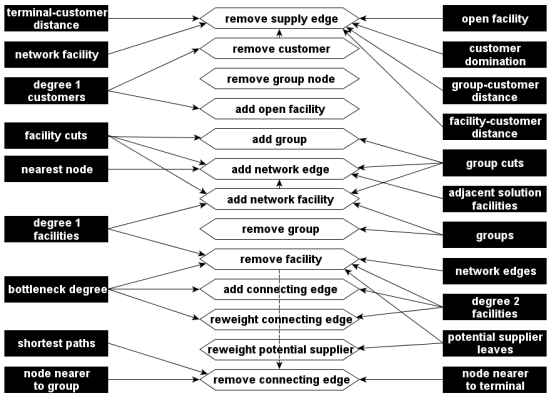


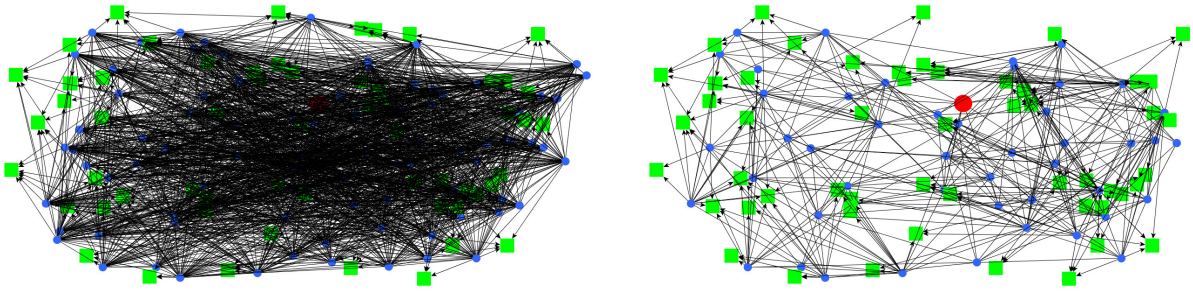
Fig. 2: Arc MB indicates that a modification of type M may have an impact on the success of test B.

IV. Computational Results

We ran our scheme on the ConFLP instances used in [4]. The problems consist of a random facility network with randomly added customer assignments . The following parameters were adjusted to create the problem classes: the probability of creating an edge in the facility network, $G[F]$ (p_eF), the probability of creating a supply edge (p_eC) and the probability of defining a facility node as a potential supplier (pF_p). Edge weights are randomly assigned ranging from 50 to 100 and opening costs from 150 to 200 respectively. Our C++ implementation of the presolving algorithm was tested on an Intel Core 2 Duo E4300 machine with 1.8 GHz, 3.25 GB RAM. We used the following default parameter setting: in the *bottleneck degree test* we set $m = 3$; the maximal size of the added groups was set to 2 and number of groups was limited to 8. The simple categorization of tests into two complexity classes already speeds up the overall testing procedure. The number of easy test loops is about twice the number of hard test repetitions. Moreover it saves computational effort to exclude hard tests from a test loop, if no need can be detected by the dependencies illustrated in the previous section. The results shown in Table I are average values of 3 random instances per group. The computation times did not exeed 5 minutes per instance. The graphs having a sparse facility network ($p_eF = 0.18$) can be preprocessed with less effort than others. The benefit of the applied methods obviously depends on the density of the customer-facility network (p_eC). If the bipartite subgraph is complete, the algorithm stops after less than 2 seconds without any reduction. On the other hand if both networks are sparse ($p_eF = p_eC = 0.18$) and not all facilities are potential suppliers ($pF_p = 0.3$), we are able to significantly reduce the size of the inputs. We obtain graphs whose numbers of nodes and edges are reduced by 36% and 61% respectively on average. We can also observe that the sparsity of the graph G is not a sufficient condition for a successful reduction. If $F_p = F$, i.e. $pF_p = 1$, the presolving is not able to remove nodes, even for a very sparse graph $G[F]$. The worst results are obtained for graphs having a complete bipartite structure and $F_p = F$. This can be explained by the fact that many tests originally designed for the STP are not directly applicable to potential suppliers. Since our benchmark instances obey a uniform structure, in which every node has almost the same degree, simple degree tests that lead facility removals have no effect at all. In extreme cases, when the degree of every customer equals $|F_p|$, the number of edges could be reduced by only 5.1%. Finally, the most remarkable reductions concerning the number of edges (between 71.6% and 85%) are obtained for graphs having complete facility networks.

TABLE I: Average presolving effectiveness on 39 instances ($|F| = 100$,$|C| = 100$) with the relative reductions r_V and r_E on V end E. Each instance class consists of 3 random instances.

Orig. ConFLP instance						Presolved EConFLP instance				
p_eF	p_eC	pF_p	$\|E\|$	$\|E_F\|$	$\|E_C\|$	$\|E\|$	$\|F\|$	$\|C\|$	r_V	r_E
0.18	0.18	0.3	1131	590	540	435	33	93	36.5	61.5
0.18	0.18	1.0	2406	594	1811	1811	100	100	0.0	24.7
0.18	0.55	0.3	2215	587	1628	1702	49	100	25.3	23.2
0.18	0.55	1.0	6062	575	5486	5486	100	100	0.0	9.5
0.18	1.00	0.3	3575	575	3000	3103	57	100	21.2	13.2
0.18	1.00	1.0	10572	572	10000	10029	100	100	0.0	5.1
0.55	0.18	0.3	3070	2538	531	647	87	94	9.0	78.9
0.55	0.18	1.0	4347	2538	1808	1808	100	100	0.0	58.4
0.55	0.55	0.3	4179	2524	1654	1815	72	100	13.8	56.6
0.55	0.55	1.0	8023	2528	5495	5495	100	100	0.0	31.5
0.55	1.00	0.3	5518	2518	3000	3154	66	100	17.0	42.8
1.00	0.18	0.3	5506	4950	556	828	96	98	3.0	85.0
1.00	0.55	0.3	6569	4950	1619	1869	80	100	9.75	71.6



(a) (b)

Fig. 3: Presolving effect on a random instance.

V. Conclusions

In this paper we provide techniques for presolving instances of the ConFLP embedded in an algorithmic framework. We extend the concept of a the ConFLP by allowing terminal facility nodes (being open or not) and groups of facilities among which at least one needs to be included in the solution. Afterwards we also describe how such an extended ConFLP instance can be reversed into a ConFLP. The new extended ConFLP concept enables the transfer of several known tests for (group) Steiner tree problems and the facility location problem. Furthermore, we propose a bunch of new presolving ideas for the ConFLP structure itself and test all of them computationally. The overall methodology is based on identified problem modification dependencies. Our algorithmic framework is tested on a set of benchmark instances from the literature showing that the proposed presolving is especially beneficial for graphs obeying a sparse edge structure with respect to the edges connecting only facilities, facilities and customers, or both. We observe that increasing the number of potential suppliers decreases the effectivity of the presolving procedures. maximal degree m for the bottleneck test, are determined manually, by running only a small number of sample instances. One possible way to improve these features is to use intelligent learning techniques to train these so-called *control values*.

References

[1] D. Karger and M. Minkoff, "Building Steiner Trees with Incomplete Global Knowledge," in *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, 2000, pp. 613 –623.

[2] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem*, ser. ADM. Amsterdam, Netherlands: North-Holland, 1992, vol. 53.

[3] S. Gollowitzer and I. Ljubic, "MIP Models for Connected Facility Location: A Theoretical and Aomputational Study," *Computers & OR*, vol. 38, no. 2, pp. 435–449, 2011.

[4] A. Tomazic and I. Ljubic, "A GRASP Algorithm for the Connected Facility Location Problem," in *SAINT*. IEEE Computer Society, 2008, pp. 257–260.

[5] M. G. Bardossy and S. Raghavan, "Dual-based local search for the connected facility location and related problems," *INFORMS Journal on Computing*, vol. 22, no. 4, pp. 584–602, 2010.

[6] D.-Z. Du and P. M. Pardalos, *Handbook of Combinatorial Optimization*. Springer London, Limited, 2005.

[7] C. W. Duin and A. Volgenant, "Reduction Tests for the Steiner Problem in Graphs," *Networks*, vol. 19, pp. 549–567, 1989.

[8] T. Polzin and S. V. Daneshmand, "Extending Reduction Techniques for the Steiner Tree Problem," in *Lecture notes in computer science*. Springer Berlin/Heidelberg, 2001.

[9] T. Polzin and S. Daneshmand, "Practical Partitioning-Based Methods for the Steiner Problem," in *WEA*, ser. Lecture Notes in Computer Science, C. Àlvarez and M. J. Serna, Eds., vol. 4007. Springer, 2006, pp. 241–252.

[10] C. E. Ferreira and F. M. de Oliveira Filho, "New Reduction Tests for the Group Steiner Tree Problem," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 1176–1188, 2006.

[11] A. Letchford, "An Aggressive Reduction Scheme for the Simple Plant Location Problem," *submitted*, 2011.