New Separation Algorithms for the Simple Plant Location Problem

Laura Galli[*] Adam N. Letchford[†] Sebastian J. Miller[†]

Draft, 24th August 2015

Abstract

The *Simple Plant Location Problem* (SPLP) is a well-known $\mathcal{NP}$-hard combinatorial optimisation problem with applications in logistics. Several families of valid inequalities (a.k.a. cutting planes) have been derived for the SPLP. On the other hand, very little attention has been paid to *separation algorithms* (i.e., algorithms to generate inequalities when needed). We close this gap in the literature by presenting several new polynomial-time separation algorithms, both exact and heuristic, each of which separates over a family of inequalities that contains an exponentially large number of facet-defining members. Computational results are also given to show which inequalities are useful in practice.

Keywords: facility location; combinatorial optimisation; branch-and-cut

1 Introduction

The *Simple Plant Location Problem* (SPLP), also known as the *Uncapacitated Facility Location Problem*, is a well-known combinatorial optimisation arising in logistics applications. We are given a set I of facilities and a set J of clients. The (non-negative) cost of opening facility $i \in I$ is denoted by f_i, and the (non-negative) cost of assigning client $j \in J$ to facility $i \in I$, assuming that the facility is open, is c_{ij}. The task is to decide which facilities to open, and to assign each client to an open facility, at minimum cost.

The SPLP is well-known to be $\mathcal{NP}$-hard in the strong sense [5, 28]. For detailed surveys on the SPLP and related problems, see, e.g., [19, 28, 29, 34]. In this paper, we concentrate on *exact* algorithms for the SPLP, i.e., algorithms for solving instances to proven optimality.

[*]Dipartimento di Informatica, Università di Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy. E-mail: `Laura.Galli@di.unipi.it`

[†]Department of Management Science, Lancaster University Management School, Lancaster LA1 4YX, United Kingdom. E-mail: `A.N.Letchford@lancaster.ac.uk`, `sebmiller64@hotmail.co.uk`

Balinski [5] formulated the SPLP as an integer program. Let m denote $|I|$ and n denote $|J|$. For each $i \in I$, define a binary variable y_i, taking the value 1 if and only if facility i is opened. For each $i \in I$ and $j \in J$, define a binary variable x_{ij}, taking the value 1 if and only if client j is assigned to (open) facility i. The formulation is then:

$$\begin{array}{lclr}
\min & \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{i \in I} f_i y_i & & (1)\ \texttt{?eq:obj?} \\
\text{s.t.} & \sum_{i \in I} x_{ij} = 1 & (j \in J) & (2)\ \boxed{\texttt{eq:assign}} \\
 & x_{ij} \leq y_i & (i \in I, j \in J) & (3)\ \boxed{\texttt{eq:VUB}} \\
 & (x, y) \in \{0,1\}^{mn+m}. & & (4)\ \boxed{\texttt{eq:bin}}
\end{array}$$

We call (2) *assignment constraints* and (3) *variable upper bounds* (VUBs).

Early success in exact algorithms for the SPLP was based on heuristics for solving the dual of the LP relaxation of this formulation (e.g., [9, 21, 26]). Later on, some success was also had with Lagrangian relaxation (e.g., [7, 8, 23]). At present, the leading exact algorithms are the one in [32], which is based on inter-communicating primal and dual heuristics, and the one in [31], which is based on problem reduction tests, followed by standard LP-based branch-and-bound.

Surprisingly, nobody has developed a modern *branch-and-cut* algorithm for the SPLP (see, e.g., [14, 17, 27] for introductions to branch-and-cut). This gap in the literature is not due to lack of knowledge of the associated family of polyhedra, which has been studied in depth [10, 15, 16, 18, 20, 22, 25]. Rather, it seems to be due to a lack of good *separation algorithms*, i.e., algorithms for generating inequalities when needed. Indeed, we are aware of only four papers that even mention separation algorithms for the SPLP [1, 11, 12, 13], and, of those, only two present computational results [1, 13].

To address this issue, we present in this paper several new exact and heuristic separation algorithms for the SPLP. Each of these algorithms has the desirable property of separating over a family of inequalities that has an exponentially large number of facet-defining members. We also present extensive computational results, to ascertain which inequalities and algorithms tend to be most useful in practice.

The rest of the paper is structured as follows. The literature is reviewed in Section 2. In Section 3, we present two new exact separation algorithms. In Section 4, we present a general-purpose separation heuristic, that can be used to generate several different kinds of valid inequalities. In Section 5, we present two more separation algorithms, which use ideas from disjunctive programming. The computational results are in Section 6. Finally, some concluding remarks are in Section 7.

Throughout the paper, we let $P(m,n)$ denote the convex hull of all pairs $(x,y) \in \mathbb{R}^{(m \times n)+m}$ that satisfy (2)–(4). We sometimes write $x(i,j)$ for x_{ij} and $y(i)$ for y_i. Moreover, sometimes we write $x(E)$ for $\sum_{\{i,j\} \in E} x_{ij}$, $x(S:T)$ for $\sum_{i \in S} \sum_{j \in T} x_{ij}$ and $y(S)$ for $\sum_{i \in S} y_i$.

2 Literature Review

⟨se:literature⟩ Now we review the relevant literature. Since the literature on the SPLP is vast, we focus here on papers concerned with polyhedra and/or separation, and refer the reader to [19, 28, 29, 34] for surveys of other approaches.

2.1 Valid inequalities

⟨sub:lit1⟩ To our knowledge, the papers that present valid inequalities for $P(m, n)$ are [10, 15, 16, 18, 20, 22, 25]. For the sake of brevity, we mention only families of inequalities for which separation algorithms (exact or heuristic) have been devised, either by other authors or ourselves.

Cornuéjols *et al.* [18] presented the following result. Let p and q be integers satisfying $1 < q < p < m$ and $p \leq n$, and suppose that p is not a multiple of q. Let $s_1, \ldots, s_p$ be distinct location indices, let $t_1, \ldots, t_p$ be distinct client indices, and take indices modulo p, so that, for example, s_{p+1} is identified with s_1. Then, the following inequality is valid for $P(m, n)$:

$$\sum_{i=1}^{p} \sum_{j=i}^{i+q-1} x(s_i, t_j) \;\leq\; \sum_{i=1}^{p} y(s_i) + p - \lceil p/q \rceil. \qquad (5)\ \boxed{\texttt{eq:circulant}}$$

In [22], such inequalities are called *circulant* inequalities.

Circulant inequalities define facets of $P(m, n)$ if and only if $p = q + 1$ [20, 25]. The circulant inequalities with $q = 2$ and p odd are called *odd cycle* inequalities [15, 20]. They take the form:

$$\sum_{i=1}^{p} \big(x(s_i, t_i) + x(s_i, t_{i+1}) \big) \;\leq\; \sum_{i=1}^{p} y(s_i) + \lfloor p/2 \rfloor. \qquad (6)\ \boxed{\texttt{eq:OC}}$$

Odd cycle inequalities define facets if and only if $p = 3$. Following [22], we will call these special odd cycle inequalities *3-cycle* inequalities.

The circulant inequalities were generalised by Aardal [1]. As before, let p and q be integers satisfying $1 < q < p \leq m$ and $p \leq n$, with p not a multiple of q. Let $T \subseteq J$ be any client set with $|T| = p$, and let $S \subseteq I$ be any location set such that $|S| \geq \lceil p/q \rceil$. Let G be any bipartite graph with node sets S and T, such that each node in S has degree q in G. Finally, let E denote the set of edges of G. Then the following inequality

$$x(E) \;\leq\; y(S) + p - \lceil p/q \rceil \qquad (7)\ \boxed{\texttt{eq:pq}}$$

is valid for $P(m, n)$. Following [22], we will call inequalities of this kind *(p, q)* inequalities.

In our own paper [22], the (p, q) inequalities were both strengthened and generalised. First, it was shown that the (p, q) inequalities are dominated by the inequalities

$$x(E) \leq r y(S) + k(q - r), \qquad (8)\ \boxed{\texttt{eq:AMIR}}$$

where k denotes $\lfloor p/q \rfloor$ and r denotes $p \bmod q$. These stronger inequalities are called *aggregated mixed-integer rounding* (AMIR) inequalities. (Note that an AMIR inequality is a (p,q) inequality if and only if $r = 1$.) Second, the condition that each node is S must have degree q in G is relaxed to the condition that the degree of each such node must be any positive multiple of q. Letting $d(i)$ denote the degree of $i \in S$ in G, we have the *generalised (p,q)* inequality

$$x(E) \ \leq \ \sum_{i \in S} \frac{d(i)}{q} y_i \ + p - \lceil p/q \rceil, \tag{9}$$?eq:gen-pq?

and the *generalised AMIR* inequality

$$x(E) \ \leq \ r \sum_{i \in S} \frac{d(i)}{q} y_i \ + k(q-r). \tag{10}$$?eq:gen-AMIR?

It is shown in [22] that, for any fixed pair p and q, the number of facet-defining AMIR inequalities grows exponentially with m and n, and so does the number of facet-defining generalised AMIR inequalities that are not AMIR inequalities. It is also pointed out that each circulant inequality (5) is equivalent to or dominated by an AMIR inequality of the form:

$$\sum_{i=1}^{p} \sum_{j=i}^{i+q} x(s_i, t_j) \leq r \sum_{i=1}^{p} y(s_i) + k(q-r). \tag{11}$$?eq:SCI?

These special AMIR inequalities are called *strengthened* circulant inequalities.

Cho *et al.* [16] presented the following result. Let j_1, j_2 and j_3 be three client indices, and let S_{12}, S_{13}, S_{23} and S_{123} be disjoint subsets of I. (We permit S_{123} to be empty.) Then the following inequality is valid and facet-defining:

$$\begin{aligned} &2\,x(S_{12} : \{j_1, j_2\}) \ + \ 2\,x(S_{13} : \{j_1, j_3\}) \ + \ 2\,x(S_{23} : \{j_2, j_3\}) \ + \\ &\quad x\big(S_{123} : \{j_1, j_2, j_3\}\big) \ \leq \ 2 \ + \ 2\,y(S_{12} \cup S_{13} \cup S_{23}) \ + \ y(S_{123}). \end{aligned} \tag{12}$$ eq:3C

Following [22], we will call inequalities of this type *3-client* inequalities.

Cho *et al.* [16] also showed that, given any facet-defining inequality that has binary coefficients for the x and y variables, one can obtain another such inequality by taking one or more facilities and "replicating" them. This basically means creating copies of the facility and setting the coefficients of the x and y variables for the copies equal to the coefficients of the x and y variables for the original. This leads to "replicated" odd cycle inequalities, circulant inequalities, and so on. However, replicating a (p,q) or generalised (p,q) inequality just leads to another inequality of the same type [22].

Figure 1, adapted from [22], shows a hierarchy of fifteen key families of inequalities. An arrow from one family to another means that the latter is

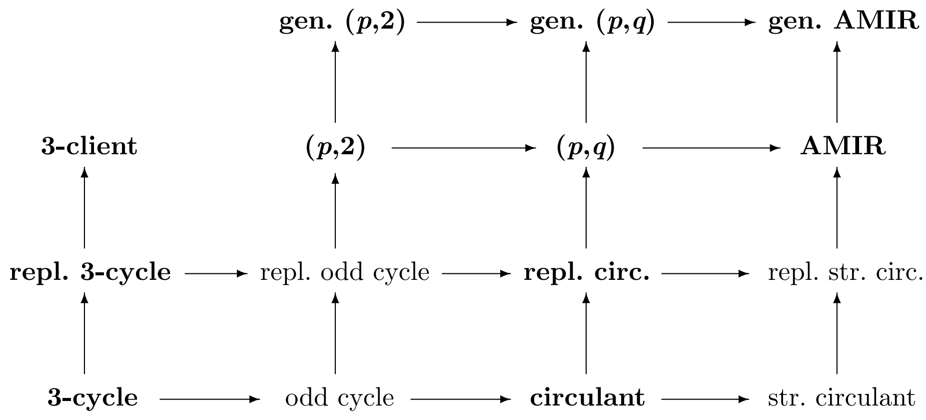

Figure 1: Hierarchy of inequalities for the SPLP.

a proper generalisation of, or dominates, the former. The prefixes "repl.", "gen." and "str." stand for "replicated", "generalised" and "strengthened", respectively, and "circ." stands for "circulant". If a family is in bold font, it means that it is known to include facet-defining members that do not belong to any of the subfamilies listed in the figure. See [22] for details.

2.2 Separation

⟨sub:lit2⟩ Finally, we mention the existing separation algorithms.

Caprara & Fischetti [11] showed that the separation problem for the *odd cycle* inequalities (6) amounts to finding a minimum-weight odd cycle in a (suitably labelled) graph with $\mathcal{O}(mn)$ nodes and $\mathcal{O}(m^2n)$ edges. Using results in [6, 24], this problem in turn can be reduced to $\mathcal{O}(mn)$ shortest path problems in a graph of twice the size. Unfortunately, even if we had a linear-time algorithm to solve these shortest-path problems, the total running time would be $\mathcal{O}(m^3n^2)$. Though polynomial, this is clearly much too slow to be useful in practice.

In their other paper [12], Caprara and Fischetti showed that odd cycle separation can in fact be done much more quickly. They show that one can construct in $\mathcal{O}(m^2n)$ time a (suitably labelled) graph, say $\tilde{G}$, with m nodes, in such a way that a violated odd cycle inequality exists if and only if there exists an odd cycle in $\tilde{G}$ with weight less than 1. By solving m shortest-path problems, one can check if such an odd cycle exists in $\mathcal{O}(m^3)$ time. This approach is fast enough to be of practical use, as shown by Caprara & Salazar [13], in the context of a generalisation of the SPLP called the *Index Selection Problem.* We remark that this faster odd cycle separation algorithm actually returns m odd cycle inequalities, one for each node of $\tilde{G}$, each of which can be checked for violation.

Finally, Aardal [1] gives a separation heuristic for the (p, q) inequalities (7), in the context of a different generalisation of the SPLP, known as the *Capacitated* Plant Location Problem. The heuristic begins by creating a bipartite graph with vertex set $I \cup J$ and an edge for each positive x variable. Then, each fractional y variable is used as a 'seed' for a set S. For each seed, the sets S, T and E are constructed in a greedy way. We have not seen an analysis of the running time, but the computational results given in [1] suggest that it is reasonable in practice.

3 Two New Exact Separation Algorithms

⟨se:exact⟩ In this section, we present two new exact separation algorithms. In Subsection 3.1, we present an algorithm for the 3-client inequalities. In Subsection 3.2, we give a more efficient algorithm for the odd cycle inequalities.

Throughout this section, we let (x^*, y^*) denote the fractional point to be separated, I_f denote the set of fractional y variables, E_f denote the set of fractional x variables, and J_f denote the set of clients with at least one fractional x variable. That is, $I_f = \{i \in I : 0 \leq y_i^* < 1\}$, $E_f = \{(i,j) : i \in I,\, j \in J,\, 0 < x_{ij}^* < 1\}$ and $J_f = \{j \in J : \exists i \in I : (i,j) \in E_f\}$.

3.1 An exact algorithm for 3-client inequalities

⟨sub:sep-3C⟩ We deal with the 3-client inequalities (12) first, since they are the easiest to handle.

⟨th:3C-sep⟩ **Theorem 1** *The separation problem for the 3-client inequalities (12) can be solved exactly in $\mathcal{O}(mn^3)$ time.*

Proof. There are $\mathcal{O}(n^3)$ possible ways to select the client triple $\{j_1, j_2, j_3\}$. For each such triple, we can find the location sets S_{12}, S_{13}, S_{23} and S_{123} that minimise the slack of the inequality (or, equivalently, maximise the violation) as follows. For each $i \in M$, compute the following four quantities:

$$\begin{gathered} 2\left(x^*(i,j_1) + x^*(i,j_2) - y_i^*\right) \\ 2\left(x^*(i,j_1) + x^*(i,j_3) - y_i^*\right) \\ 2\left(x^*(i,j_2) + x^*(i,j_3) - y_i^*\right) \\ x^*(i,j_1) + x^*(i,j_2) + x^*(i,j_3) - y_i^*. \end{gathered}$$

If at least one of these quantities is positive, then place i in S_{12}, S_{13}, S_{23} or S_{123}, respectively, according to which of the quantities is the largest. Otherwise, do not place i in any of those sets. Since determining these sets for a given client triple takes only $\mathcal{O}(m)$ time, the total running time is $\mathcal{O}(mn^3)$, as stated. □

Although a running time of $\mathcal{O}(mn^3)$ is rather high, it should be noted that the algorithm can return up to $\binom{n}{3}$ violated inequalities, rather than just one, in a single call. Moreover, it is possible to speed up the algorithm, by exploiting the sparsity of the LP solution. This is shown in the following lemma and theorem.

⟨le:3client⟩ **Lemma 1** *If a 3-client inequality is violated, then $j_1, j_2, j_3 \in J_f$.*

Proof. First, we show that, if $x^*(i, j_1) = 1$ for some $i \in S_{12}$, then the 3-client inequality cannot be violated. To see this, observe that, if we sum together the equation $-2x(i, j_1) = -2$, the assignment constraints (2) for j_1 and j_2 and the VUBs (3) for client j_3 and all locations in S_{13}, S_{23} and S_{123}, the resulting inequality dominates the 3-client inequality.

By symmetry, the same applies to j_1 and S_{13}. That is, if $x^*(i, j_1) = 1$ for some $i \in S_{13}$, then the 3-client inequality cannot be violated.

Next, we show that, if $x^*(i, j_1) = 1$ for some $i \in S_{123}$, then the 3-client inequality cannot be violated. To see this, observe that, if we sum together the equation $-x(i, j_1) = -1$, the VUB $x(i, j_1) \leq y_i$, the assignment constraints for j_1, j_2 and j_3, the VUBs for client j_1 and all locations in S_{12} and S_{13}, the VUBs for client j_2 and all locations in S_{12} and S_{23}, and the VUBs for client j_3 and all locations in S_{13} and S_{23}, the resulting inequality dominates the 3-client inequality.

Third, we show that, if $x^*(i, j_1) = 0$ for all $i \in S_{12} \cup S_{13} \cup S_{123}$, then the 3-client inequality cannot be violated. To see this, observe that, if we sum together the equations $-2x(i, j_1) = 0$ for all $i \in S_{12} \cup S_{13} \cup S_{123}$, the assignment constraints for j_1 and j_2, the VUBs for client j_2 and all locations in S_{123}, and twice the VUBs for client j_2 and all locations in S_{12} and S_{13}, we obtain an inequality that dominates the 3-client inequality.

Thus, $x^*(i, j_1)$ must be fractional for at least one index i, and therefore j_1 must lie in J_f. By symmetry, the same applies to j_2 and j_3. □

Proposition 1 *The separation problem for the 3-client inequalities (12) can be solved exactly in $\mathcal{O}\left(|J_f|^2 |E_f|\right)$ time.*

Proof. Due to the VUBs (3), the first of the four quantities in the proof of Theorem 1 can be positive only if both $x^*(i, j_1)$ and $x^*(i, j_2)$ are positive. By Lemma 1, this implies that (i, j_1) and (i, j_2) belong to E_f. So, it is only worth computing the first quantity if this holds. By symmetry, analogous statements hold for the second and third quantities. As for the fourth quantity, it can only be larger than all of the first three quantities if $x^*(i, j_1)$, $x^*(i, j_2)$ and $x^*(i, j_3)$ are all positive. By Lemma 1, this implies that (i, j_1), (i, j_2) and (i, j_3) all belong to E_f. So, it is only worth computing the fourth quantity if that holds.

To exploit these facts, we begin by computing, for each client $j \in J_f$, a list of the locations $i \in I$ for which $(i, j) \in E_f$. We let $d(j)$ denote the *degree* of client $j \in J_f$, which is the size of the associated list. This computation takes $\mathcal{O}(|E_f|)$ time in total. Once this is done, we can compute the four quantities more quickly for each given client triple $\{j_1, j_2, j_3\}$, by only considering locations that are in the relevant lists. The total time taken over all triples is:

$$\sum_{\{j_1,j_2,j_3\} \subseteq J_f} \mathcal{O}\big(d(j_1) + d(j_2) + d(j_3)\big) = \mathcal{O}\left(|J_f|^2 \sum_{j \in J_f} d(j)\right) = \mathcal{O}\left(|J_f|^2 |E_f|\right)$$

as stated. □

3.2 A faster exact algorithm for odd cycle inequalities

⟨sub:sep-OC⟩ Now we turn our attention to the odd cycle inequalities (6). As mentioned in Subsection 2.2, the exact odd cycle separation algorithm of Caprara and Fischetti [12] runs in $\mathcal{O}(m^3 + m^2 n)$ time, and returns m odd cycle inequalities, which may or may not be violated.

We now show how to speed up the exact odd cycle separation algorithm, by exploiting the sparsity of the LP solution.

⟨le:OC-frac⟩ **Lemma 2** *If an odd cycle inequality (6) is violated, then all of the variables involved must be fractional at* (x^*, y^*).

Proof. Suppose that $y^*(s_p) = 1$. In this case, the odd cycle inequality cannot be violated, since it is implied by the equation $-y_p = -1$, the assignment constraints (2) for clients $t_1, t_3, \ldots, t_p$, and the VUBs (3) for the pairs (s_{i-1}, t_i) and (s_i, t_i) for $i = 2, 4, \ldots, p-1$.

Now suppose that $x^*(s_p, t_1) = 0$. Again, the odd cycle inequality cannot be violated, since it is implied by the equation $x^*(s_p, t_1) = 0$, the assignment constraints for clients $t_2, t_4, \ldots, t_{p-1}$, the VUBs for the pairs (s_i, t_i) for $i = 1, 3, \ldots, p$ and the VUBs for the pairs (s_i, t_{i-1}) for $i = 2, 4, \ldots, p-1$.

By symmetry, none of the y variables involved can take the value 1, and none of the x variables involved can take the value 0. The VUBs then imply that all variables involved must be fractional. □

⟨th:OC⟩ **Theorem 2** *The exact odd cycle separation algorithm in [12] can be modified so that it runs in* $\mathcal{O}\big(|I_f|^3 + \min\{|I_f|, |J_f|\}\, |E_f|\big)$ *time, and returns, for each* $i \in I_f$*, the odd cycle inequality with least slack (or most violation) among all odd cycle inequalities with* $i \in \{s_1, \ldots, s_p\}$.

Proof. We recall some facts about the algorithm in [12]. The key to the algorithm is that the odd cycle inequalities can be written in the form:

$$\sum_{k=1}^{p} \big(1 + y(s_k) + y(s_{k+1}) - 2x(s_k, t_k) - 2x(s_{k+1}, t_k)\big) \;\geq\; 1.$$

Accordingly, the algorithm starts by constructing a complete graph, say $\tilde{G}$, with vertex set I, in which the weight of the edge $\{i, i'\}$ is:

$$\min_{j \in J} \big(1 + y^*(i) + y^*(i') - 2x^*(i,j) - 2^*x(i',j)\big).$$

Then, a violated odd cycle inequality exists if and only if there exists in $\tilde{G}$ an odd cycle (i.e., a cycle containing an odd number of edges), with total weight less than 1. The algorithm then uses the fact (from [6, 24]) that finding a minimum weight odd cycle in a graph passing through a given node can be reduced to a shortest-path problem in a graph twice the size of the original. Doing this for all $i \in I$ yields m odd cycles, whose weights are then checked to see if any are less than 1.

Now, due to Lemma 2, the node set of $\tilde{G}$ can be set to I_f rather than I. The same lemma implies that both (s_k, t_k) and (s_{k+1}, t_k) must lie in E_f for all k, which implies that $\tilde{G}$ need not be complete. Specifically, if we let $\tilde{E}$ denote the set of edges in $\tilde{G}$, we need include the edge $\{i, i'\}$ in $\tilde{E}$ only if there exists at least one client $j \in J_f$ such that both (i, j) and (i', j) lie in E_f.

We now show that $\tilde{E}$ has only $\mathcal{O}\big(\min\{|I_f|, |J_f|\}\,|E_f|\big)$ edges and can be constructed in $\mathcal{O}\big(\min\{|I_f|, |J_f|\}\,|E_f|\big)$ time. We consider two cases. First, suppose that $I_f < J_f$. Scan each location $i \in I_f$ to find all clients $j \in J_f$ such that $\{i, j\} \in E_f$. For each such j, include the edge $\{i, i'\}$ in $\tilde{E}$ if and only if $\{i', j\} \in E_f$. There are $|E_f|$ suitable pairs $\{i, j\}$, and processing each pair takes $\mathcal{O}(|I_f|)$ time, leading to $\mathcal{O}(|I_f|\,|E_f|)$ time in total. Second, suppose that $J_f < I_f$. For each client $j \in J_f$, include the edge $\{i, i'\}$ in $\tilde{E}$ if and only if both $\{i, j\}$ and $\{i', j\}$ are in E_f. The number of suitable pairs i, i' for a given j is the square of the degree of j, defined in the proof of Lemma 1. The sum of the squared degrees over all j is $\mathcal{O}(|J_f|\,|E_f|)$. Note that, in either case, we can compute the weights of the edges in $\tilde{E}$ as we go along.

Once the reduced version of $\tilde{G}$ has been created, we solve I_f shortest-path problems in $\tilde{G}$, which takes $\mathcal{O}\big(|I_f|^3\big)$ time. Finally, we output one odd cycle inequality for each of the $|I_f|$ shortest-path calls. □

We remark that this improved algorithm can still return up to $|I_f|$ violated inequalities in a single call.

4 A General-Purpose Separation Heuristic

⟨se:heuristic⟩ In this section, we present a general-purpose separation heuristic, that can be applied to several different families of inequalities. The key to this heuristic, presented in Subsection 4.1, is that, if the client set T is fixed, then one can solve the separation problem exactly and quickly for the given families. In Subsection 4.2, we present heuristics for choosing the set T.

4.1 Exact separation for fixed T

⟨sub:given-T⟩ The following four propositions show how to solve the separation problem for various inequalities when T (and optionally q) is fixed.

⟨pr:fix-T1⟩ **Proposition 2** *For fixed T and q, the separation problems for (p, q) and AMIR inequalities can be solved in $\mathcal{O}(mn)$ time.*

Proof. Since T is fixed, p is also fixed. If p is either divisible by q or smaller than q, we stop. Otherwise, finding the most violated (p, q) inequality, if one exists, amounts to finding a set $S \subseteq I$ and an edge set E that maximises

$$\sum_{\{i,j\} \in E} x^*_{ij} - \sum_{i \in S} y^*_i.$$

To find these sets, we do the following for each $i \in I$. Let $j_1, \ldots, j_q$ be the q clients in T that have the largest value of x^*_{ij}. If

$$\sum_{k=1}^{q} x^*(i, j_k) > y^*_i,$$

then insert i into S and insert the edges $(i, j_1), \ldots, (i, j_q)$ into E. This takes $\mathcal{O}(n)$ time for each $i \in I$, which is $\mathcal{O}(mn)$ time in total.

AMIR inequalities are handled in an analogous way. The only difference is that, for a given i, we check whether

$$\sum_{k=1}^{q} x^*(i, j_k) > r y^*_i,$$

before deciding to place i into S and $(i, j_1), \ldots, (i, j_q)$ into E. □

We remark that, even for fixed T and q, the number of facet-defining AMIR inequalities can be exponential in both m and n. (E.g., for fixed T with $|T| = 3$, there can be exponentially many replicated 3-cycle inequalities.)

⟨pr:fix-T2⟩ **Proposition 3** *For fixed T, the separation problems for (p, q) and AMIR inequalities can be solved in $\mathcal{O}(mn + |E_f| \log n)$ time.*

Proof. We start by sorting the clients in T, for each $i \in I$, in non-decreasing order of x^*_{ij}. If d_i represents the number of edges in E_f that are incident on i, then sorting for a given facility i takes $\mathcal{O}(n + d_i \log d_i)$ time. Thus, sorting for all $i \in I$ takes $\mathcal{O}(mn + |E_f| \log n)$ time. For a given i, let the sorted clients be labelled $j_{i1}, \ldots, j_{ip}$.

Now, for all $i \in I$ and for $c = 1, \ldots, p$, let $X(i, c)$ denote the partial sum $\sum_{\ell=1}^{c} x^*(i, j_{i\ell})$. Computing these partial sums takes $\mathcal{O}(mn)$ time.

In what follows, we will use the real variable X, which will keep track of the quantity $\sum_{e \in E} x^*_e$ throughout the rest of the separation algorithm, and the real variable Y, which will keep track of $\sum_{i \in S} y^*_e$.

For (p, q) inequalities, we then do the following. Initially S is empty, and X and Y are zero. Set q to the smallest integer that does not divide p. For all $i \in I \setminus S$, check whether $X(i, q) > y^*_i$. If so, add i to S and y^*_i to Y. Set X to $\sum_{i \in S} X(i, q)$. If $X > Y + p - \lceil p/2 \rceil$, set

$$E = \bigcup_{i \in S} \bigcup_{c=1}^{q} \{(i, j_{ic})\},$$

output the violated (p, q) inequality and stop. If $q = p - 1$, stop. Otherwise, set q to the next smallest integer that does not divide p, and repeat.

One can check that the procedure described in the previous paragraph runs in $\mathcal{O}(mn)$ time. Thus, the running time of the whole procedure is dominated by that of the sorting routine.

The procedure for AMIR inequalities is similar, except that we insert i into S if $X(i, q) > r y^*_i$, and we check whether $X > rY + k(q - r)$ to decide whether the AMIR inequality is violated. □

⟨pr:fix-T3⟩ **Proposition 4** *For fixed T and q, the separation problems for the generalised (p, q) and generalised AMIR inequalities can be solved in $\mathcal{O}(mn + |E_f| \log n)$ time.*

Proof. We start by sorting the clients in T and then computing the partial sums $X(i, c)$, as described in the proof of the previous proposition. Then, for a given $i \in I$, we now have the option of placing $q, 2q, \ldots$ edges into E. The best multiple, say cq, is the one that maximises $X(i, cq) - c y^*_i$ (in the case of generalised (p, q) inequalities) or $X(i, cq) - r c y^*_i$ (in the case of generalised AMIR inequalities). One can find the best value of c in $\mathcal{O}(\log |T|/q) = \mathcal{O}(\log n)$ time via binary search. If the corresponding quantity $X(i, cq) - c y^*_i$ is positive for the given i, then i should be placed into S. The total time of the binary search procedure is $\mathcal{O}(m \log n)$, which is dominated by that of the initial sorting. □

Proposition 5 *For fixed T, the separation problems for the generalised (p, q) and generalised AMIR inequalities can be solved in $\mathcal{O}(mn \log n)$ time.*

Proof. As in the proof of the previous two propositions, we start by sorting the clients in T and then computing the partial sums $X(i, c)$. We then proceed as in the proof of Proposition 3, except that, for a given value of q and a given $i \in I$, we use the binary search procedure described in the proof of Proposition 4 to check whether i should be placed into S. The total time of the binary search procedure is now $\mathcal{O}(mn \log n)$, which dominates that of the initial sorting. □

4.2 Finding suitable candidates for T

⟨sub:find-T⟩ To put the results in the previous subsection to practical use, we need heuristics for generating suitable candidates for the client set $T \subset J$. We propose three such heuristics.

The first heuristic is based on the exact odd cycle separation algorithm described in Subsection 3.2. As explained in that subsection, the algorithm generates $|I_f|$ candidate odd cycle inequalities, each of which may or may not be violated. Associated with each such inequality is a client set $\{t_1, \ldots, t_p\}$, that we can use as a candidate for T. (In practice, one can do the following for each of the $|I_f|$ odd cycle inequalities. If the inequality is violated already, then output it. Otherwise, take the candidate set T and search for a violated (generalised) (p, q) or AMIR inequality. In this way, no time is wasted in redundant odd cycle calculations.)

The second heuristic is based on the following observation. Suppose that $q = 2$, and consider a pair of clients $\{j_1, j_2\} \subset J_f$. From the proof of Proposition 2, it seems worthwhile putting t_1 and t_2 into T simultaneously if there exists a facility i such that $x^*(i, j_1) + x^*(i, j_2) - y_i^*$ is large. We use this idea as follows. We construct a graph whose node set is J_f and which initially has no edges. We then call a pair $\{j_1, j_2\} \subset J_f$ a *potential edge* if there exists a facility i with $\{i, j_1\}, \{i, j_2\} \in E_f$. We sort the potential edges in non-increasing order of

$$\max_{i \in I_f} \{x^*(i, j_1) + x^*(i, j_2) - y_i^*\} .$$

Then we add the potential edges to the graph, in sorted order, recording any newly created connected and biconnected components as we go. Each such component, if it contains at least 3 nodes, forms a candidate for T.

We remark that the argument used in the proof of Theorem 2 can be used to show that the potential edges can be found in $\mathcal{O}\big(\min\{|I_f|, |J_f|\}\, |E_f|\big)$ time. The sorting of the potential edges then takes $\mathcal{O}(|J_f|^2 \log |J_f|)$ time. The heuristic does however yield $\mathcal{O}(|J_f|)$ candidates for T.

The third heuristic is simply to take a previously-generated (generalised) (p, q) or AMIR inequality that has small slack in the optimal solution to the current LP relaxation, and take the associated set T. For this, it helps to store the set T corresponding to each inequality generated.

5 Separation via Disjunctive Arguments

⟨se:disjunctive⟩ In this section, we show that two more separation results can be obtained via *disjunctive* arguments. (See, e.g., [2, 3, 4, 14, 17] for details on the disjunctive approach to integer programming, and [30] for applications of the approach to the design of separation algorithms for combinatorial optimisation problems.) Unfortunately, the results in this section are likely to be of theoretical interest only, since they rely on the solution of large LPs.

5.1 AMIR separation for fixed S

?⟨sub:sep-S⟩? Our next separation result is concerned with AMIR inequalities in which the facility set S is fixed. To prove it, we will need the following two known results, due to Balas [2] and Tardos [33], respectively:

⟨th:balas⟩ **Theorem 3 (Balas, 1979)** *Let $P = \{x \in \mathbb{R}^n : \; Ax \leq b\}$ be a polytope (i.e., a bounded polyhedron), where $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$. Suppose it is known that all integer points inside P satisfy a disjunction of the form*

$$\bigvee_{k=1}^{q} (f^k \cdot x \leq d_k), \tag{13}$$ eq:disjunction

where $f^k \in \mathbb{Q}^n$ and $d_k \in \mathbb{Q}$ for $k = 1, \ldots, q$. Also let x^* be a point that lies in P. Then, to test whether x^* violates an inequality that is implied by the inequalities $Ax \leq b$ and the disjunction (13), it suffices to solve the following LP:

$$\begin{array}{rll} \max & x^* \cdot \alpha & \\ s.t. & \alpha \geq A^T \lambda^k + \mu_k f^k & (k = 1, \ldots, q) \\ & b^T \cdot \lambda^k + d^k \mu_k \leq 1 & (k = 1, \ldots, q) \\ & \alpha \in \mathbb{R}^n & \\ & \lambda^k \in \mathbb{R}^m_+ & (k = 1, \ldots, q) \\ & \mu \in \mathbb{R}^q_+. & \end{array}$$

If $(\alpha^*, \lambda^*, \mu^*)$ is an optimal solution of this LP, and $x^* \cdot \alpha^* > 1$, then $\alpha^* \cdot x \leq 1$ is the desired violated inequality.

⟨th:tardos⟩ **Theorem 4 (Tardos, 1986)** *There exists a strongly polynomial algorithm (i.e., one that uses only addition, subtraction, multiplication and comparisons, and works with integers whose encoding length is bounded by a polynomial in the input size) for LPs whose constraint coefficents are all in $\{0, 1, -1\}$.*

We will also need the following proposition:

⟨pr:disj⟩ **Proposition 6** *The AMIR inequality (8) is implied by the assignment constraints (2), the VUBs (3), the non-negativity inequalities $x_{ij} \geq 0$ for $i \in I$ and $j \in J$, and the disjunction:*

$$\big(y(S) \leq k\big) \; \vee \; \big(y(S) \geq k + 1\big), \tag{14}$$ eq:disj1

where, as usual, k denotes $\lfloor p/q \rfloor$.

Proof. As usual, let r denote $p \bmod q$. We consider two cases:

1. $y(S) \geq k+1$. Multiplying by r and re-arranging yields the inequality $0 \leq r\,y(S) - r(k+1)$. Adding to this the valid inequality $x(E) \leq p$, which is implied by the assignment constraints (2) and non-negativity, we obtain $x(E) \leq r\,y(S) + p - r(k+1) = r\,y(S) + k(q-r)$.

2. $y(S) \leq k$. This implies that $0 \leq (r-q)y(S) + (q-r)k$. Adding to this the valid inequality $x(E) \leq q\,y(S)$, which is implied by the VUBs (3) for all $\{i,j\} \in E$, we obtain $x(E) \leq r\,y(S) + k(q-r)$.

So, if either of the two terms in the disjunction holds, the AMIR inequality is satisfied. □

Now we can present our next separation result:

Theorem 5 *Let S be fixed. There exists a strongly polynomial algorithm that solves the separation problem for a family of valid linear inequalities for $P(m,n)$ that includes all of the AMIR inequalities (8) with the given set S.*

Proof. Let (x^*, y^*) be a feasible solution to the LP relaxation of the SPLP. From Proposition 6, if (x^*, y^*) violates an AMIR inequality for the given S, it violates a disjunctive inequality that is implied by the constraints in the LP relaxation and a disjunction of the form (14), with k set to $\lfloor y^*(S) \rfloor$. Theorem 3 then implies that the separation problem for such disjunctive inequalities can be formulated as an LP of polynomial size. Moreover, since the constraints (2) and (3) have coefficients in $\{0, -1, +1\}$, the LP satisfies the condition mentioned in Theorem 4. As a result, it can be solved with a strongly polynomial algorithm. □

Now, recall from Subsection 2.1 that circulant inequalities with $q = p-1$ are examples of facet-defining AMIR inequalities. If we fix the facilities $s_1, \ldots, s_p$ in such a circulant inequality, the number of possible candidates for the clients $t_1, \ldots, t_p$ is $n!/(n-p)!$. Therefore, the separation algorithm described in this subsection separates over a family of valid inequalities that includes at least $n!/(n-|S|)!$ facet-defining members. If $|S|$ is permitted to vary, the number of such members is exponential in n.

5.2 Separation for replicated circulant inequalities

?⟨sub:sep-rc⟩? Our last separation result is concerned with replicated circulant inequalities. We begin by noting that these inequalities can be written in the form:

$$\sum_{i=1}^{p} \sum_{j=i}^{i+q-1} \sum_{k \in S_i} x(k, t_j) \leq \sum_{i=1}^{p} \sum_{k \in S_i} y_k + p - \lceil p/q \rceil, \qquad (15)$$

eq:RCI

where $S_1, \ldots, S_p$ are disjoint sets of facilities. We will also need the fact that the replicated circulant inequalities are dominated by replicated *strengthened* circulant inequalities, which take the form:

$$\sum_{i=1}^{p} \sum_{j=i}^{i+q} \sum_{k \in S_i} x(k, t_j) \leq r \sum_{i=1}^{p} \sum_{k \in S_i} y_k + k(q-r). \tag{16}$$ eq:RSCI

We will also need the following lemma.

Lemma 3 *For any client $j \in J$, any feasible solution to the SPLP satisfies the following disjunction:*

$$(x_{1j} = 1) \vee \cdots \vee (x_{mj} = 1), \tag{17}$$ eq:multiple-choice

where '$\vee$' represents logical 'or'.

Proof. Trivial. □

We remark that Balas [2] calls a disjunction of the type (17) a *multiple choice* disjunction. The following two propositions show how to derive replicated circulant inequalities from these disjunctions.

⟨pr:fixy⟩ **Proposition 7** *Suppose that a pair $(x^*, y^*) \in [0,1]^{(m \times n)+m}$ is feasible for the LP relaxation. If it also satisfies $y_\ell = 1$ for some $\ell \in \bigcup_{i=1}^{p} S_i$, then it satisfies the replicated strengthened circulant inequality (16).*

Proof. Assume without loss of generality that $t_j = j$ for $j = 1, \ldots, p$, and that $\ell \in S_1$. We define the following index set:

$$\bar{S} = \bigcup_{k=1}^{\lfloor p/q \rfloor} \{kq+1, \ldots, kq+r\},$$

and note that $|\bar{S}| = r\lfloor p/q \rfloor$. The proof then works in four steps. First, due to the VUBs (3), we have that (x^*, y^*) must satisfy:

$$\sum_{j \in S} \sum_{i=j-q+1}^{j} \sum_{k \in S_i} x_{ij} \ \leq \ r \sum_{i=2}^{p} \sum_{k \in S_i} y_k. \tag{18}$$ eq:fixy1

Second, due to the assignment constraints (2) for $j \in \{1, \ldots, p\} \setminus \bar{S}$, together with the non-negativity on x, we have that (x^*, y^*) must satisfy:

$$\sum_{j \in \{1,\ldots,p\} \setminus \bar{S}} \sum_{i=j-q+1}^{j} \sum_{k \in S_i} x_{kj} \ \leq \ k(q-r). \tag{19}$$?eq:fixy2?

Third, since $\ell \in S_1$ by assumption, (x^*, y^*) must satisfy:

$$0 \ \leq \sum_{k \in S_1} y_k - r. \tag{20}$$ eq:fixy3

Finally, summing together (18)–(20), we obtain:

$$\sum_{j=1}^{p} \sum_{i=j-q+1}^{j} \sum_{k \in S_i} x_{kj} \leq r \sum_{i=1}^{p} \sum_{k \in S_i} y_k + p - r - r\lfloor p/q \rfloor.$$

This is equivalent to (16), since $r = p - q\lfloor p/q \rfloor$ by definition. □

⟨pr:multiple-choice⟩ **Proposition 8** *Every replicated circulant inequality (15) is implied by the inequalities defining the LP relaxation, together with a single multiple-choice disjunction of the form (17).*

Proof. Again, we can assume without loss of generality that $t_j = j$ for all j in (15). Consider the disjunction that is obtained by setting $j = q$ in (17). Suppose that $x_{kq} = 1$ for some $k \in S_1$. Then $y_k = 1$. Proposition 7 then implies that the replicated strengthened circulant inequality (16) is satisfied. Since this inequality dominates the replicated circulant inequality (15), the latter inequality is also satisfied. By symmetry, the same is true if $x_{kq} = 1$ for some $k \in S_1 \cup \cdots \cup S_q$.

So, suppose that $x_{kq} = 0$ for $k \in S_1 \cup \cdots \cup S_q$. This implies that $X \leq p-1$, or, equivalently, $0 \geq X - p + 1$. If we multiply the inequality $0 \geq X - p + 1$ by $(q-1)/q$ and add it to the inequality $Y \geq X/q$ (which is implied by the VUBs), we obtain $Y \geq X - p + (p+q-1)/q$. This dominates the replicated circulant inequality $Y \geq X - p + \lceil p/q \rceil$. □

Now we can present our final separation result.

Theorem 6 *There exists a family of valid linear inequalities for $P(m, n)$, including all of the replicated circulant inequalities (15), whose associated separation problem can be solved in strongly polynomial time.*

Proof. For a fixed j, consider all valid inequalities for $P(m, n)$ that are implied by the inequalities that define the LP relaxation, together with the jth multiple-choice disjunction (17). By Proposition 8, these disjunctive inequalities include all replicated circulant inequalities such that $j \in T$. Theorem 3 then implies that the separation problem for those inequalities can be formulated as an LP of polynomial size. Moreover, since the constraints (2) and (3) have coefficients in $\{0, -1, +1\}$, the LP satisfies the condition mentioned in Theorem 4. As a result, it can be solved with a strongly polynomial algorithm. Running that algorithm for $j = 1, \ldots, n$ yields the desired separation algorithm. □

Note that replicated circulant inequalities are facet-defining if and only if $q = p - 1$. From this one can deduce that the number of facet-defining replicated circulant inequalities grows exponentially with both m and n. Therefore, the separation algorithm described in this subsection separates over a family of valid inequalities that has an exponential number of facet-defining members.

6 Computational Experiments

⟨se:experiments⟩ **Perhaps use instances of Hansen *et al.*, 2007 and/or Kratica *et al.*, 2001. There are some other families available in 'UFLlib'.**

6.1 Exact algorithms

We could compare exact algorithms for 3-cycle, odd cycle, replicated 3-cycle and 3-client inequalities. One might expect 3-client inequalities to close more of the integrality gap than the odd cycle inequalities.

6.2 Heuristic algorithms

We could use the exact algorithm for odd cycle inequalities as a benchmark, and see how much is gained when we use the client sets in heuristics for replicated odd cycle, (generalised) $(p, 2)$, (generalised) (p, q) and (generalised) AMIR inequalities.

6.3 Algorithms based on disjunctions

Likely to be of theoretical use only, since the cut-generating LPs are huge. But we could perhaps give it a try, for completeness.

6.4 Combinations of algorithms

Try to find a combination of the algorithms that works well in practice. A sensible order might be:

1. Connected components for (generalised?) $(p, 2)$.
2. Odd cycles for (generalised?) $(p, 2)$.
3. Connected components for (generalised?) AMIR.
4. Odd cycles for (generalised?) AMIR.
5. Exact 3-client.

7 Conclusion

⟨se:end⟩ Although the SPLP has been studied in depth from a polyhedral point of view, only three separation algorithms were known until the writing of this paper. We have presented four new separation algorithms, each of which separates over a family of inequalities that contains an exponentially large number of facet-defining inequalities. We also presented a faster exact algorithm for odd cycle inequalities.

Among the separation algorithms that we have tested...

... seem to be most useful in practice.

There are several questions for future research. The first is whether there exists a fully combinatorial polynomial-time separation routine, not based on the solution of Linear Programs, for any family of valid inequalities that includes either the replicated odd cycle inequalities or the circulant inequalities. Second, one could also look for polynomial-time separation routines for the other inequalities mentioned in [10, 15, 16, 20]. Finally, it would be interesting to know whether any of the inequalities discussed in this paper, or variants of them, could somehow be used to obtain new or improved approximation algorithms for the SPLP.[1]

References

A98 [1] K. Aardal (1998) Capacitated facility location: separation algorithms and computational experience. *Math. Program.*, 81, 149–175.

B79 [2] E. Balas (1979) Disjunctive programming. *Ann. Discr. Math.*, 5, 3–51.

BCC93 [3] E. Balas, S. Ceria & G. Cornuéjols (1993) A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Math. Program.*, 58, 295–324.

BP02 [4] E. Balas & M. Perregaard (2002) Lift-and-project for mixed 01 programming: recent progress. *Discr. Appl. Math.*, 123, 129–154.

B65 [5] M. Balinski (1965) Integer programming: methods, uses, computation. *Mgmt. Sci.*, 12, 254–313.

BM86 [6] F. Barahona & A.R. Mahjoub (1986) On the cut polytope. *Math. Program.*, 36, 157-173.

B93 [7] J.E. Beasley (1993) Lagrangian heuristics for location problems. *Eur. J. Opl Res.*, 65, 383–399.

BVA12 [8] C. Beltran-Royo, J. Vial, A. Alonso-Ayuso (2012) Semi-lagrangian relaxation applied to the uncapacitated facility location problem. *Comput. Opt. & Appl.*, 51, 387–409.

BK77 [9] O. Bilde & J. Krarup (1977) Sharp lower bounds and efficient algorithms for the simple plant location problem. *Ann. Discr. Math.*, 1, 79–88.

CLM02 [10] L. Cánovas, M. Landete & A. Marín (2002) On the facets of the simple plant location packing polytope. *Discr. Appl. Math.*, 124, 27–53.

[1] There is a new paper by Shi Li which uses residual capacity inequalities to get a better approximation for the capacitated p-median problem.

CF96 [11] A. Caprara & M. Fischetti (1996) $\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts. *Math. Program.*, 74, 221–235.

CF96b [12] A. Caprara & M. Fischetti (1996) Odd cut-sets, odd cycles, and 0-1/2 Chvátal-Gomory cuts. *Ricerca Operativa*, 26, 51–80.

CS96 [13] A. Caprara & J.J. Salazar–González (1996) A branch-and-cut algorithm for a generalization of the uncapacitated facility location problem. *Top*, 4, 135–163.

CBD10 [14] D.-S. Chen, R.G. Batson & Y. Dang (2010) *Applied Integer Programming: Modeling and Solution.* Wiley.

CJPR83a [15] D.C. Cho, E.J. Johnson, M.W. Padberg & M.R. Rao (1983a) On the uncapacitated location problem I: valid inequalities and facets. *Math. Oper. Res.*, 8, 579–589.

CJPR83b [16] D.C. Cho, E.J. Johnson, M.W. Padberg & M.R. Rao (1983b) On the uncapacitated location problem II: facets and lifting theorems. *Math. Oper. Res.*, 8, 590–612.

CCZ15 [17] M. Conforti, G. Cornuéjols & G. Zambelli (2015) *Integer Programming.* Graduate Texts in Mathematics, vol. 271. Springer.

CFN77 [18] G. Cornuéjols, M.L. Fischer & G.L. Nemhauser (1977) On the uncapacitated location problem. *Ann. Discr. Math.*, 1, 163–177.

CNW90 [19] G. Cornuéjols, G.L. Nemhauser & L.A. Wolsey (1990) The uncapacitated facility location problem. In: P.B. Mirchandani & R.L. Francis (eds.), *Discrete Location Theory*, pp. 1–54. New York: Wiley.

CT82 [20] G. Cornuéjols & J.-M. Thizy (1982) Some facets of the simple plant location polytope. *Math. Program.*, 23, 50–74.

E78 [21] D. Erlenkotter (1978) A dual-based procedure for uncapacitated facility location. *Oper. Res.*, 26, 992–1009.

GLM15a [22] L. Galli, A.N. Letchford & S.J. Miller (2015) New valid inequalities and facets for the simple plant location problem. *Working paper*, Department of Management Science, Lancaster University, UK.

GR89 [23] R. Galvão & L. Raggi (1989) A method for solving to optimality uncapacitated location problems. *Ann. Oper. Res.*, 18, 225–244.

GS86 [24] A.M.H. Gerards & A.J. Schrijver (1986) Matrices with the Edmonds-Johnson property. *Combinatorica*, 6, 365–379.

G80 [25] M. Guignard (1980) Fractional vertices, cuts and facets of the simple plant location problem. *Math. Program. Study*, 12, 150–162.

K89 [26] M. Körkel (1989) On the exact solution of large-scale simple plant location problems. *Eur. J. Opl Res.*, 39, 157–173.

KV12 [27] B. Korte & J. Vygen (2012) *Combinatorial Optimization: Theory and Algorithms* (5th Edn.) Algorithms and Combinatorics vol. 21. Springer.

KP83 [28] J. Krarup & P.M. Pruzan (1983) The simple plant location problem: survey and synthesis. *Eur. J. Opl Res.*, 12, 36–81.

LL97 [29] M. Labbé & F. Louveaux (1997) Location problems. In: M. Dell'Amico, F. Maffoli & S. Martello (eds.), *Annotated Bibliographies in Combinatorial Optimization*, pp. 261-281. Chichester: Wiley.

L01 [30] A.N. Letchford (2001) On disjunctive cuts for combinatorial optimization. *J. Comb. Opt.*, 5, 299–315.

LM14 [31] A.N. Letchford & S.J. Miller (2014) An aggressive reduction scheme for the simple plant location problem. *Eur. J. Oper. Res.*, 234, 674–682.

PFM14 [32] M. Posta, J. Ferland & P. Michelon (2014) An exact cooperative method for the uncapacitated facility location problem. *Math. Program. Comput.*, 6, 199–231.

T86 [33] É. Tardos (1986) A strongly polynomial algorithm to solve combinatorial linear programs. *Oper. Res.*, 34, 250–256.

V11 [34] V. Verter (2011) Uncapacitated and capacitated facility location problems. In: H.A. Eiselt & V. Marianov (eds.), *Foundations of Location Analysis*, pp. 25–38. International Series in Operations Research & Management Science, vol. 155. Berlin: Springer.