A New Extended Formulation with Valid Inequalities for the Capacitated Concentrator Location Problem

Massimo Di Francesco
Dipartimento di Matematica e Informatica, Università di Cagliari, 09124 Cagliari, Italy
Email: mdifrance@unica.it

Manlio Gaudioso
Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica,
Università della Calabria, 87036 Rende, Italy
Email: manlio.gaudioso@unical.it

Enrico Gorgone
Dipartimento di Matematica e Informatica, Università di Cagliari, 09124 Cagliari, Italy
Indian Institute of Management Bangalore, 560076 Bangalore, India
Email: egorgone@unica.it

Ishwar Murthy
Indian Institute of Management Bangalore, 560076 Bangalore, India,
Email: ishwar@iimb.ac.in

We present a new disaggregated formulation of the Capacitated Concentrator Location Problem (CCLP) using the notion of cardinality of terminals assigned to a concentrator. This formulation consists of O(*mnn*) variables and constraints, where *m* denotes the number of concentrators and *n* the number of terminals, respectively. We prove that this extended formulation is stronger than the traditional one. We also present two classes of inequalities exploiting the cardinality effect of the extended formulation. The first class is a generalization of the well-known Cover and (1, *k*)-Configuration inequalities, which collectively are stronger than the original Cover and (1, *k*)-Configuration inequalities. The second class, called the 2-Facility Cardinality Matching Inequality, holds for the uncapacitated version of the Concentrator Location Problem and can be lifted to become a strong inequality for CCLP. We solve the LP relaxation of the extended formulation and use separation heuristics to identify and sequentially add the previous valid inequalities to improve the lower bound. This approach is embedded in a branch-and-bound. We test our solution approach on a large set of benchmark problems. The experimentation shows that we can identify the optimal solution at the root node in most of the problem instances with up to 50 concentrators and 50 terminals. For larger sized test problems with up to 100 concentrators and 1000 terminals, the branch-and-cut procedure using the disaggregated formulation outperforms the branch-and-cut procedure applied to the traditional formulation both in terms of CPU and the required number of branch-and-bound nodes.

Keywords: Integer Programming, Concentrator Location, Valid Inequalities.

1. Introduction

The Capacitated Concentrator Location Problem (CCLP) is a classic problem in network design and has relevant applications in computer networks. The problem is best described as the optimal design of a layered network where a central node is to be connected to a set of terminals through a set of satellite nodes or concentrators [1]. This problem takes on greater importance due to its equivalence to the Single

Source Capacitated Facility Location Problem (SSCFLP), which has applications is logistics and supply chain. Given a set $M = \{1,\ldots, m\}$ of potential concentrator locations and a set $N = \{1,\ldots, n\}$ of terminals, a subset of locations in M have to be selected where concentrators will be opened with each terminal in N being assigned to exactly one of the concentrators that are set up. The cost of setting up a concentrator in location i is f_i, while its associated capacity to provide computing resource to terminals is C_i. The cost of serving each terminal j from concentrator location i is c_{ij}, while the terminal's computing demands is d_j. Without loss of generality, one can assume $d_j \leq C_i$. CCLP is a decision problem in which a set of concentrators in M are opened and each terminal in N is assigned to one of the open concentrators. Such a decision must ensure that terminals are assigned to the opened concentrators such that their capacity is respected. The problem then is to determine the decision that minimizes the sum of the cost of setting up concentrators and the cost of assignment of terminals to concentrators. Let y_i = 1 if concentrator $i \in M$ is set up, 0 otherwise. Similarly, let x_{ij} = 1 if terminal j is assigned to concentrator i, 0 otherwise. The standard integer programming formulation of CCLP is

$$\textbf{(P}_{\textbf{x-y}}\textbf{)} \qquad Minimize \qquad F(x,y) = \sum_{i\in M}\sum_{i\in N} c_{ij}x_{ij} + \sum_{i\in M} f_i\, y_i$$

s.t.

$$\sum_{i\in M} x_{ij} = 1 \qquad \forall j \in N \qquad (1)$$

$$x_{ij} \leq y_i \qquad \forall i \in M, j \in N \qquad (2)$$

$$\sum_{j\in N} d_j x_{ij} \leq C_i y_i \qquad \forall i \in M \qquad (3)$$

$$x_{ij}, y_i \in \{0,1\} \qquad \forall i \in M, j \in N \qquad (4)$$

In **(P$_{\text{x-y}}$)**, (1) (known as *semi-assignment constraints*) specifies that each terminal in N is to be assigned to exactly one concentrator. Constraints (2), known as Variable Upper Bound (VUB) constraints, ensure that if a concentrator i is not opened, terminal j cannot be assigned to it. Knapsack constraints (3) enforce the capacity restriction on each concentrator i. CCLP is known to be NP-Hard and therefore there has been of considerable interest in developing solutions that can solve large instances of it in reasonable time.

1.1 Literature Review

While several articles have been published on solution procedures to solve CCLP, they broadly fall under two categories. The first and perhaps the most prevalent has been the use of Lagrangian relaxation. Boffey [4], Pirkul [15] and Lo and Kershenbaum [11] were one of the earliest to address the solution to CCLP and their variants. Pirkul [15], followed by Pirkul and Nagarajan [16], Sridharan [18], Holmberg et al. [9] and Cortinhal and Captivo [7], all approached this problem by using Lagrangian relaxation. Starting with the formulation **(P$_{\text{x-y}}$)**, constraints (1) were dualized, giving rise to a sub-problem that involves solving a series of knapsack problem. While solving knapsack problems is not

"easy", it is well known that dualizing (1) instead of (2), provides tighter lower bounds, and is therefore overall more beneficial. There exist algorithms that solve the knapsack problems optimally whose computational effort is pseudo-polynomial in the worst case. Martello et al. [13] provide an excellent review of pseudo-polynomial algorithms to solve the 0-1 knapsack problem. It needs to be noted here that since the knapsack relaxation is embedded in a branch-and-bound to solve CCLP, such exact methods will still exhibit exponentially bounded effort in the worst case. The papers listed above all used sub-gradient optimization to solve the Lagrangian dual. Their papers differed in the primal heuristic methods used to generate good upper bounds. However, all the approaches essentially used information provided by the Lagrangian dual to obtain good feasible solutions. Here, it is worth mentioning the work of Celani et al. [5], who developed a fast dual-ascent procedure to solve the Lagrangian dual, instead of employing sub-gradient optimization. Chen and Ting [6] supplemented the Lagrangian relaxation approach with Ant Colony method to generate upper bounds. Similarly, Barcelo et al [2], combined Column generation with Lagrangian relaxation to solve CCLP.

Another category of papers focused on the polyhedral structure of formulations of CCLP, introducing strong formulations along with their attendant valid inequalities. These then result in a branch-and-cut approach to find an exact solution to CCLP. Shima [17] considered a generalization of CCLP where in each location i, K concentrator options are possible, each with different costs and capacities. Starting with a formulation similar to $\mathbf{(P_{x\text{-}y})}$, they replaced the y variables by its complement ($z = 1 - y$). The resulting formulation in x and z variables transforms the constraints similar to (3) into knapsack constraints with a constant right-hand-side. They then introduced a form of lifted cover inequalities, which are incorporated into a branch-and-cut framework. Labbè and Yaman [10] introduced the Quadratic Capacitated-Concentrator Location (QCL) Problem, whose formulation involves constraints which are quadratic. They studied the polytope of the resulting formulation and developed strong inequalities for it. They then incorporated these inequalities as cuts in a branch-and-cut algorithm. Yang et a. [19] considered the formulation $\mathbf{(P_{x\text{-}y})}$ for CCLP. In their approach, they introduced Lifted Cover Inequalities (LCI) and Fenchel cutting planes (FCI) that arise from (3). They implemented exact separation algorithms for both. Further, they implemented a cut-and-solve approach, wherein the branching is done on a sum of variables, akin to a GUB constraint. Thus, a sub-problem is partitioned into a *sparse* problem and a *dense* problem. The sparse problem being small is solved exactly and therefore fathomed. Subsequent branching, if needed, is only done on the *dense* problem. Gouveia and Saldanha-da-Gama [8] considered a special case of CCLP in which the demands $d_j = 1$ for all $j \in N$. Thus, constraint (3) in $\mathbf{(P_{x\text{-}y})}$ amounts to each concentrator i being capable of handling at most Q terminals. They further considered an extension wherein several capacity options can be chosen at each concentrator location. For both problems, they presented an extended formulation that disaggregates the y variables in $\mathbf{(P_{x\text{-}y})}$ into various cardinalities, each representing the number of terminals assigned to it. They also presented "$\leq$" and "$\geq$" inequalities for their extended formulation. Finally, Ahuja et al. [1] focus entirely on reliable heuristics to find a good feasible solution to CCLP.

1.2 Contributions of this Paper

In this paper, we first present a new extended formulation of CCLP that induces cardinality of terminals assigned to concentrator locations by appropriately disaggregating x and y variables in **($\mathbf{P_{x\text{-}y}}$)** (similar to but not the same as in [8], as discussed later in the paper). The disaggregated formulation consists of $O(mn^2)$ variables and constraints. While the disaggregated formulation is larger than that in **($\mathbf{P_{x\text{-}y}}$)**, it is stronger, i.e., its LP relaxation provides a tighter lower bound than that provided by **($\mathbf{P_{x\text{-}y}}$)**. We next present two classes of inequalities that are specific to the disaggregated formulation. First, generalizations of the well-known *Cover* and (1-*k*) *Configuration* inequalities for the disaggregated formulation are presented. In an accompanying working paper [12], it is shown that these generalizations when added to the disaggregated formulation, provide a tighter lower bound than when their counterparts are added to **($\mathbf{P_{x\text{-}y}}$)**. This is because several of the *Cover* and (1-*k*) *Configuration* inequalities in the disaggregated formulation imply stronger knapsack inequalities in **($\mathbf{P_{x\text{-}y}}$)**. The second class of inequalities, called the 2-*Facility Cardinality Matching* inequality, holds for the un-capacitated version of the Concentrator Location Problem and can be lifted to become a strong inequality for CCLP. We provide details on how they can be lifted easily. At the root node, first the disaggregated formulation is solved, followed by the addition of violated valid inequalities that are identified by appropriate separation heuristics. If no more violated inequalities can be identified and the LP solution is still fractional, then branching takes place. Thus, our procedure can be characterized as a branch-and-cut approach, wherein cuts are added only at the root node.

The rest of this paper is organized as follows. In section 2, the disaggregated formulation is formally presented. We also show why the LP relaxation of the disaggregated formulation is tighter than that of **($\mathbf{P_{x\text{-}y}}$)**. In section 3, we present the two classes of valid inequalities alluded to earlier: a) generalizations of Cover and (1, *k*) Configuration inequalities, and b) 2-Facility Cardinality Matching inequalities. We also describe briefly, separation heuristics for each. In section 4, we present a detailed computational study that assesses the effectiveness of our approach in terms of strength of the bounds and time required to solve the problems to optimality. We provide concluding remarks in section 5.

2. Extended Formulation of CCLP

Associated with each concentrator location $i \in M$, let K_i denote the maximum number of terminals that the concentrator at location i has the capacity to handle. We disaggregate the model **($\mathbf{P_{x\text{-}y}}$)** by separating each location into K_i location-cardinality combinations. Note that for each $i \in M$, K_i can be determined as follows. The demands d_j are first sorted from smallest to largest. K_i then represents the maximum number of terminals that can be accommodated so that the accumulation does not exceed C_i. The following binary variables are now defined:

$$y_{ik_i} = \begin{cases} 1, & if\ k_i\ terminals\ are\ assigned\ to\ concentrator\ i, \\ 0, & otherwise \end{cases},$$

$$z_{ijk_i} = \begin{cases} 1, & if\ terminal\ j\ is\ assigned\ to\ concentrator\ i\ with\ cardinality\ k_i, \\ 0, & otherwise. \end{cases}$$

The extended formulation is:

($\mathbf{P_{z\text{-}y}}$)

$$Minimize \qquad F(z,y) = \sum_{i\in M}\sum_{j\in N}\sum_{k_i=1}^{K_i} c_{ij} z_{ijk_i} + \sum_{i\in M}\sum_{k_i=1}^{K_i} f_i y_{ik_i}$$

s.t.

$$\sum_{i\in M}\sum_{k_i=1}^{K_i} z_{ijk_i} = 1 \qquad \forall j \in N, \qquad (5)$$

$$z_{ijk_i} \le y_{ik_i} \qquad \forall i \in M, j \in N, k_i = 1, \dots\dots, K_i, \qquad (6)$$

$$\sum_{j\in N} d_j z_{ijk_i} \le C_i y_{ik_i} \qquad \forall i \in M, k_i = 1, \dots\dots, K_i, \qquad (7)$$

$$\sum_{j\in N} z_{ijk_i} = k_i y_{ik_i} \qquad \forall i \in M, k_i = 1, \dots\dots, K_i, \qquad (8)$$

$$\sum_{k_i=1}^{K_i} y_{ik_i} \le 1 \qquad \forall i \in M, \qquad (9)$$

$$z_{ijk_i}, y_{ik_i} \in \{0,1\} \qquad \forall i \in M, j \in N, k_i = 1, \dots\dots, K_i \qquad (10)$$

In **($\mathbf{P}_{z\text{-}y}$)**, (5) represents the *semi-assignment* constraints across concentrator locations and cardinality. Constraints (6), known as *variable upper bound* (VUB) constraints ensure that if a concentrator i with designated cardinality k_i is not setup, then terminal j associated with cardinality k_i cannot be assigned to it. Constraints (7) represents the knapsack constraints, defined for each concentrator and cardinality. *Cardinality* constraints (8) enforce the requirement that if a concentrator of a specified cardinality k_i is used, then the number of terminals assigned to it must equal to k_i. Constraints (9) ensure that at most one cardinality type is used for each concentrator.

It is important to highlight here the difference between our formulation to that presented by Gouveia and Saldanha-da-Gama [8] for CCLP. To begin with, our formulation addresses the most general case of CCLP, while theirs considers a special case of CCLP wherein the knapsack constraint (3) is replaced by a requirement that each concentrator can accommodate at most Q terminals. Consequently, in Gouveia and Saldanha-da-Gama's [8] formulation, the binary variables x_{ij} as defined in **($\mathbf{P}_{x\text{-}y}$)** are preserved, while the variables y_i are disaggregated into Q number of variables y_{ik_i}, one for each k_i = 1,…., Q. The constraints (2) and (3) in **($\mathbf{P}_{x\text{-}y}$)** are replaced by constraints, $\sum_{j\in N} x_{ij} \le \sum_{k_i=1}^{Q} k_i y_{ik_i}$ and (9) in **($\mathbf{P}_{z\text{-}y}$)** for each $i \in M$. To model the same problem, our formulation in **($\mathbf{P}_{z\text{-}y}$)** specializes to one in which knapsack constraints (7) are removed, and K_i is replaced by a common upper bound of Q. The principal difference then is that we disaggregate the x_{ij} variables as well.

It is worth noting that **($\mathbf{P}_{z\text{-}y}$)** and **($\mathbf{P}_{x\text{-}y}$)** are strictly <u>not</u> equivalent. Every feasible solution in **($\mathbf{P}_{z\text{-}y}$)** translates to an equivalent solution in **($\mathbf{P}_{x\text{-}y}$)** with the same objective function value, while the converse is not true. Specifically, **($\mathbf{P}_{x\text{-}y}$)** allows a concentrator i to be setup (y_i =1) without any terminal being assigned to it. Such a solution is not feasible in **($\mathbf{P}_{z\text{-}y}$)**. However, assuming that, without loss of generality

$f_i \geq 0$, the optimal solution to **($\mathbf{P_{x\text{-}y}}$)** will always be one in which if a concentrator is selected, at least one terminal will be assigned to it. This ensures that the optimal solution obtained from solving **($\mathbf{P_{z\text{-}y}}$)** and **($\mathbf{P_{x\text{-}y}}$)** have the same optimal value.

We next show that the LP relaxation of **($\mathbf{P_{z\text{-}y}}$)** provides a tighter lower bound than the LP relaxation of **($\mathbf{P_{x\text{-}y}}$)**. Consider the following LP polyhedra:

$LP(x, y) = \{(x, y) \in R^{mn+m} \mid (1)\text{-}(3),\ x \geq 0,\ y \geq 0, \text{ are satisfied}\},$ (11)

$LP(z, y) = \{(z, y) \in R^{mn^2+mn} \mid (5)\text{-}(9),\ z \geq 0,\ y \geq 0, \text{ are satisfied}\}.$ (12)

For each $(z, y) \in LP(z, y)$, consider the aggregation

$$x_{ij} = \sum_{k_i=1}^{K_i} z_{ijk_i} \qquad \forall i \in M, j \in N. \qquad (13a)$$

$$y_i = \sum_{k_i=1}^{K_i} y_{ik_i} \qquad \forall i \in M, j \in N, \qquad (13b)$$

Accordingly, let

$LP_a(x, y) = \{(x, y) \in R^{mn+m} \mid (13) \text{ are satisfied for each } (z, y) \in LP(z, y)\}.$ (14)

Proposition 1. $LP_a(x, y) \subset LP(z, y)$.

Proof: We first show that $LP_a(x, y) \subseteq LP(x, y)$. Consider a solution $(z', y') \in LP(z, y)$. By aggregating z' and y' as described in (13a) and (13b), we obtain a $(x', y') \in LP_a(x, y)$. As well, constraints (5)-(7) get aggregated to become constraints (1)-(3), respectively, which along with $x' \geq 0$ and $y' \geq 0$, are satisfied. Hence, $(x', y') \in LP(x, y)$ and $LP_a(x, y) \subseteq LP(x, y)$.

We now identify instances in which $(x, y) \in LP(x, y)$, but $(x, y) \notin LP_a(x, y)$. Consider a $(\hat{x}, \hat{y}) \in LP(x, y)$, where for some pair $(i_1, i_2) \in M$, $\hat{y}_{i_1} = \hat{y}_{i_2} = 1$. Let $J(i_1) \subset N$ and $J(i_2) \subset N$ denote the set of terminals assigned to i_1 and i_2, respectively. Further, let i) $\sum_{j \in J(i_1)} d_j < C_{i_1}$, ii) $\sum_{j \in J(i_2)} d_j < C_{i_2}$, but there exists a $j_1 \in \{N\text{-}J(i_1)\text{-}J(i_2)\}$ such that $\sum_{j \in J(i_1)} d_j + d_{j_1} > C_{i_1}$, but $\sum_{j \in J(i_2)} d_j + d_{j_2} > C_{i_2}$. To complete the solution $(\hat{x}, \hat{y})$, $\hat{x}_{i_1 j} = 1$ for $j \in J(i_1)$, $\hat{x}_{i_1 j_1} = \Delta = (C_{i_1} - \sum_{j \in J(i_1)} d_j)/d_{j_1}$, $\hat{x}_{i_2 j} = 1$ for $j \in J(i_2)$ and $\hat{x}_{i_2 j_1} = 1 - \Delta$. Note that $0 < \Delta < 1$. By contradiction, suppose that $(\hat{x}, \hat{y}) \in LP_a(x, y)$. Then, $\hat{y}_{i_1} = \sum_{k_i=1}^{K_{i1}} \hat{y}_{i_1 k_i} = 1$. Let $|J(i_1)| = k_1$. Since the number of positive $\hat{x}_{i_1 j}$ variables in $(\hat{x}, \hat{y})$ is k_1+1, it follows from (6) and (8), that $\sum_{k_i=k_1+2}^{K_{i1}} \hat{y}_{i_1 k_i} = 0$. Observe that with regards to i_1, $(\hat{x}, \hat{y})$ satisfies knapsack constraint (3) exactly. Therefore,

$$\sum_{j \in J(i_1)} d_j \left(\sum_{k_i=1}^{k_1+1} \hat{z}_{i_1 j k_i}\right) + d_{j_1} \left(\sum_{k_i=1}^{k_1+1} \hat{z}_{i_1 j_1 k_i}\right) = C_{i_1} \left(\sum_{k_i=1}^{k_1+1} \hat{y}_{i_1 k_i}\right). \qquad (15)$$

It is clear from VUB constraint (6) that the only way (15) would hold is if each of the knapsack constraints (7) associated with i_1 and each k_i=1,…k_1+1 is satisfied exactly. Further, due to (6), the only way cardinality constraint (8) for $k_i = k_1$+1 can be satisfied is if $\hat{y}_{i_1 k_1+1} = \hat{z}_{i_1 j, k_1+1} = \Delta$ for each

$j \in \{J(i_1), j_1\}$. However, since $\sum_{j \in J(i_1)} d_j + d_{j_1} > C_{i_1}$, knapsack constraint (7) associated with $k_i = k_1+1$ is violated. This is suitably illustrated in Figure 1 below. Hence, $(\hat{x}, \hat{y}) \notin LP_a(x, y)$. □

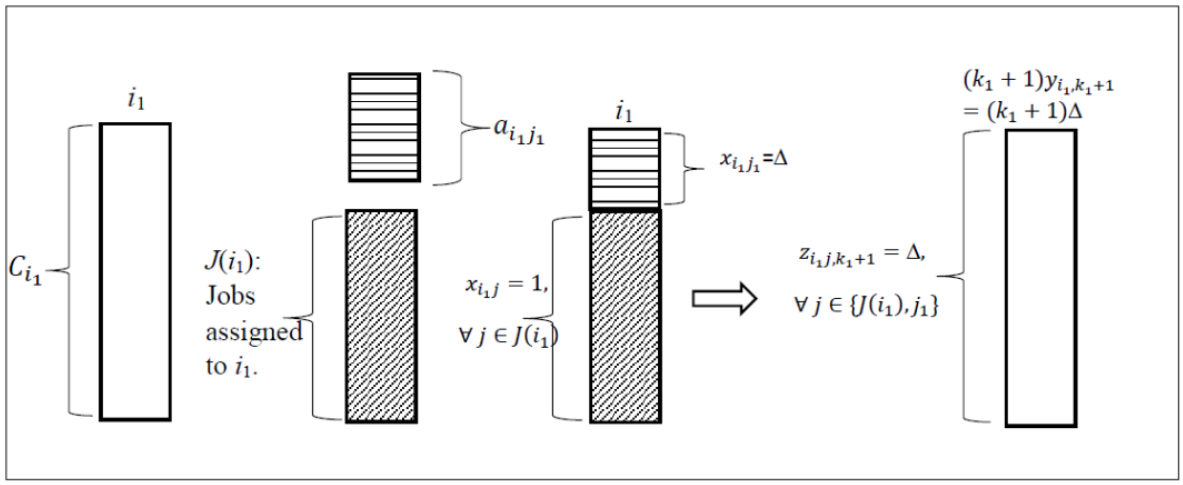

Figure 1. Illustration of an instance where $LP_a(x, y) \subset LP(z, y)$.

3. Valid Inequalities for ($P_{z\text{-}y}$)

In this section, we present three classes of inequalities, i) Cardinality Constrained Cover, ii) (1, $\hat{p}_k$)-Configuration inequalities, and iii) 2-Facility Cardinality Matching inequalities. We also present separation heuristics for each one of them.

3.1. Cardinality Constrained Cover inequalities

We first present a generalization of the Cover inequality for **($P_{z\text{-}y}$)**, followed by a generalization of the (1, k)-Configuration inequality. Let

$H(x, y) = Conv\{(x, y) \in R^{mn+m} |$ (1)-(4)$\}$ and (16)

$H(z, y) = Conv\{(z, y) \in R^{w} |$ (5)-(10), $w = \sum_{i=1}^{m}(n + 1)K_i$. (17)

A cover inequality that is valid for $H(x, y)$ is defined as follows. Consider a subset $S \subseteq N$ with $|S| = s$ such that,

(i) for all $R_i \subset S$, with $|R_i| = r_i$, $\sum_{j \in R_i} d_j \leq C_i$,
(ii) however, for all $R_{i+1} \subseteq S$, $|R_{i+1}| = r_i+1$, $\sum_{j \in R_{i+1}} d_j > C_i$.

Given the above conditions, a (s, r_i)-cover inequality that is valid for $H(x, y)$ is:

$$\sum_{j \in S} x_{ij} \leq r_i y_i. \quad (18)$$

Definition 1. Consider a knapsack inequality, $\sum_{j\in N} d_j x_j > C$. For some $D \subseteq N$, and $0 \le k \le |D|$, $V^*(D, k) = Min\{\sum_{j\epsilon D} d_j x_j \,|\, \sum_{j\epsilon D} x_j = k, x\epsilon\{0,1\}^{|D|}\}$.

Given that k items have to be selected, $V^*(D, k)$ represents the cumulative 'demand' if k smallest items are selected out of set D. Now consider a set S defined in conditions (i) and (ii) above leading to (18) and $k_i \ge 2$. For some integer $\hat{r}_{ik_i} < Min\ \{k_i, s\}$ that satisfies the conditions

a) for some $\hat{R}_{ik_i} \subset S$, $|\hat{R}_{ik_i}| = \hat{r}_{ik_i}$, $\sum_{j\in\hat{R}_{ik_i}} d_j + V^*\left(N - \hat{R}_{ik_i}, k_i - \hat{r}_{ik_i}\right) \le C_i$, (19)

b) but for all $\hat{R}_{ik_i+1} \subseteq S$, $|\hat{R}_{ik_i+1}| = \hat{r}_{ik_i}+1$, $\sum_{j\in\hat{R}_{ik_i+1}} d_j + V^*\left(N - \hat{R}_{ik_i+1}, k_i - \hat{r}_{ik_i} - 1\right) > C_i$, (20)

the following $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality is valid for $H(z, y)$:

$$\sum_{j\in S} z_{ijk_i} \le \hat{r}_{ik_i} y_{ik_i}. \tag{21}$$

In [12], the strength of the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality is discussed in depth, including conditions under which (21) is a facet of $H(z, y)$. Suffice it to say, two properties of the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality stand out vis-à-vis the (s, r_i)-cover inequality. First, for a given S there is one (s, r_i)-cover inequality, while several $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities can be constructed, one for each $k_i \ge 2$. Thus, $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities are a lot more ubiquitous. Second, given that $V^*\left(N - \hat{R}_{ik_i+1}, k_i - \hat{r}_{ik_i} - 1\right) \ge 0$, it follows that $\hat{r}_{ik_i} \le r_i$ and (21) dominates (18). The following property shows that the addition of all $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities to $LP(z, y)$ will provide a tighter lower bound than the addition of all (s, r_i)-cover inequalities to $LP(x, y)$.

Proposition 2. Let i) $LP_{a\text{-}cov}(z, y) = \{(z, y)\in R^w |\ (z, y)$ satisfies (5)-(9), $z \ge 0$, $y \ge 0$, and $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality (21) for every $i\in M$, $S \subseteq N$ and $k_i\}$ where $w = \sum_{i=1}^{m}(n+1)K_i$, and ii) $LP_{a\text{-}cov}(x, y) = \{(x, y)\in R^{mn+m} |\ (x, y)$ satisfies (13) for each $(z, y)\in LP_{a\text{-}cov}(z, y)$. Then, $LP_{a\text{-}cov}(x, y) \subset LP(x, y) \cap \{(x, y)$ satisfies (s, r_i)-cover inequality (18) for every $i\in M$, $S \subseteq N\}$.

Proof: Let $LP_{cov}(z, y) = \{(z, y)\in R^w |\ (z, y)$ satisfies the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality (21) for every $i\in M$, $S \subseteq N$ and $k_i\}$ and $LP_{cov}(x, y) = \{(x, y)\in R^{mn+m} |\ (x, y)$ satisfies (13) for each $(z, y)\in LP_{cov}(z, y)$, $x\ge 0$, $0\le y\le 1\}$. By definition, $LP_{a\text{-}cov}(z, y) = LP_a(z, y) \cap LP_{cov}(z, y)$ and therefore $LP_{a\text{-}cov}(x, y) = LP_a(x, y) \cap LP_{cov}(x, y)$. Aggregating constraints in (21) over k_i gives $\sum_{j\in S} x_{ij} \le \sum_{k_i=1}^{K_i} \hat{r}_{ik_i} y_{ik_i}$. Due to (9) and the fact that $\hat{r}_{ik_i} \le r_i$, it follows that $\sum_{k_i=1}^{K_i} \hat{r}_{ik_i} y_{ik_i} \le r_i y_i$. Therefore, $LP_{cov}(x, y) \subseteq \{(x, y)\in R^{mn+m} |\ (x, y)$ satisfies (s, r_i)-cover inequality (18) for every $i\in M$, $S \subseteq N\}$. From Proposition 1, $LP_a(x, y) \subset LP(z, y)$. Therefore, $LP_{a\text{-}cov}(x, y) \subset LP(x, y) \cap \{(x, y)$ satisfies (s, r_i)-cover inequality (18) for every $i\in M$, $S \subseteq N\}$. □

The following example illustrates Proposition 2.

Example 1. Consider a concentrator i with $C_i = 60$ with demands d_j in sorted order being [15, 15, 14, 14, 13, 12, 8, 8, 7, 5, 4, 4, 4, 3, 3]. For $S = \{1,\ldots, 6\}$, a (6, 4)-cover inequality is $\sum_{j\in S} x_{ij} \le 4y_i$. Intuitively, the (6, 4)-cover inequality enforces the condition that at most 4 items in S can be accommodated. For a given S, several $(s, k_i, \hat{r}_{ik_i})$-*cover* distinct inequalities can be configured. Observe that $V^*(N - S, 4)$=14, $V^*(N - S, 5)$=18 and $V^*(N - S, 6)$=23, implying that the 4, 5 and 6 'smallest'

terminals consume resources of amounts, 14, 18 and 23, respectively. From this we obtain $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities: *i*) $\sum_{j\in S} z_{ij8} \leq 3y_{i8}$ and *ii*) $\sum_{j\in S} z_{ij9} \leq 2y_{i9}$. Intuitively, inequalities in *i*) (and *ii*)) describe the fact that if 8 (or 9) terminals have to be assigned to *i*, then at most 3 (or 2) terminals in *S* can be accommodated, each of which are stronger than the (6, 4)-cover inequality. This is illustrated in Figure 2 below.

We now illustrate possible solutions $(z, y)\in LP(z, y)$ solutions that violate one or more of the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities above. More importantly, the corresponding solution $(x, y)\in LP_a(x, y)$ satisfies all the pertinent (s, r_i)-cover inequalities. Consider the partial (z, y) solution: $y_{i8} = 0.5$, $z_{i3,8} = 0.1$, $z_{i4,8} = z_{i5,8} = z_{i6,8} = 0.5$, $z_{i11,8} = 0.4$, $z_{i12,8} = z_{i13,8} = z_{i14,8} = z_{i15,8} = 0.5$, $y_{i9} = 0.5$, $z_{i2,9} = 0.125$, $z_{i3,9} = z_{i4,9} = 0.5$, $z_{i,9,9} = 0.375$, $z_{i,10,9} = z_{i,11,9} = z_{i,12,9} = z_{i,13,9} = z_{i,14,9} = z_{i,15,9} = 0.5$. Here, in fractional terms, the assignment of 8 and 9 terminals to *i* come into force. While this LP solution satisfies (6)-(9), the inequalities, *i*) $\sum_{j\in S} z_{ij8} \leq 3y_{i8}$, where $S = \{1,\ldots, 6\}$ and *ii*) $\sum_{j\in S} z_{ij9} \leq 2y_{i9}$, where $S = \{1,\ldots,4\}$, are both violated. The corresponding $(x, y)\in LP_a(x, y)$ is: $y_i = 1$, $x_{i2} = 0.125$, $x_{i3} = 0.6$, $x_{i4} = 1.0$, $x_{i5} = x_{i6} = 0.5$, $x_{i9} = 0.375$, $x_{i10} = 0.5$, $x_{i11} = 0.9$, $x_{i12} = x_{i13} = x_{i14} = x_{i15} = 1.0$. This solution however satisfies the (6, 4)-cover inequality $\sum_{j\in S} x_{ij} \leq 4y_i$, where $S = \{1,\ldots, 6\}$. □

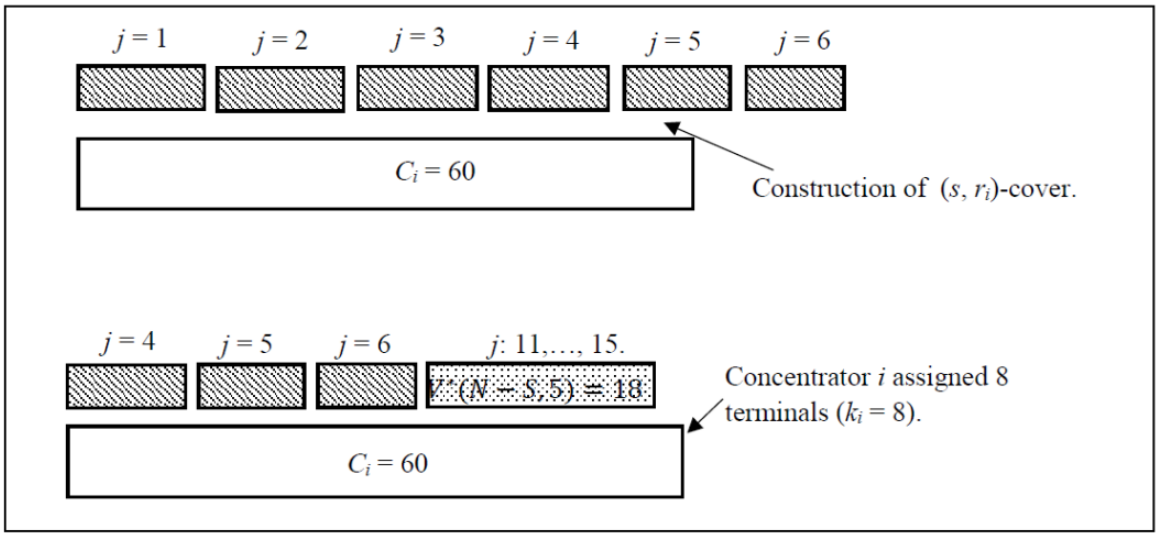

Figure 2. Illustration of Example 1.

An additional point is worth noting about the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequalities described above. First, a special case of the $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality is when $\hat{r}_{ik_i} = 0$. In Example 1 above, observe that for $S = \{j\in N: d_j \geq 23\}$, $\hat{r}_{i9} = 0$, implying that $z_{ij9} = 0$ for all $j\in S$. This is because, $V^*(N - S, 8) = 38$.

To identify a $(s, k_i, \hat{r}_{ik_i})$-*cover* inequality that a solution to the LP relaxation of **($\mathbf{P_{z\text{-}y}}$)** violates, the following separation heuristic is proposed. First, a *i*-k_i combination for which $y_{ik_i} > 0$, along with its support, $S_{ik_i} = \{j \in N | z_{ijk_i} > 0\}$ such that $|S_{ik_i}| = s_{ik_i}$ is identified. Next, items in S_{ik_i} are sorted in increasing order, in terms of the function value, $g_j(y_{ik_i}, z_{ijk_i}, d_j) = (y_{ik} - z_{ijk} + 0.1)/d_j$. This function captures the idea that larger the value of z_{ijk} and d_j, more preferred is it to include *j* in *S*. A linear search for $S\subseteq S_{ik_i}$ begins with $s = 2$. In the first iteration, the most 'preferred' items 1 to *s* are chosen from S_{ik_i}

and added to S. If (19) or (20) are not satisfied with $\hat{r}_{ik_i} = s$-1, then in the second iteration, items 2 to s+1 are chosen. Thus, in each subsequent iteration l, items l to s+l-1 are tested for (19) and (20) till s+l-1 equals s_{ik_i}. When this occurs, s increments by 1 and the search continues. The search terminates when $s = s_{ik_i}$. Whenever both (19) and (20) are satisfied for a given S, then $\hat{r}_{ik_i} = s$-1. At this point the cover inequality (21) is lifted by expanding S to selectively include items in $\{S_{ik_i}\text{-}S\}$ in the following way. First, all items $j \in \{S_{ik_i}\text{-}S\}$ whose $d_j \geq Max\ \{d_j | j \in S\}$ are included. Next, if for $j \in \{S_{ik_i}\text{-}S\}$ such that $d_j < Max\ \{d_j | j \in S\}$, $V^*(S, \hat{r}_{ik_i}) + d_j + V^*(S_{ik_i}\text{-}S, k_i\text{-}\hat{r}_{ik_i}+1) > C_i$, then j is included in S. After this lifting procedure, if the resulting cover inequality (21) is violated, it is added as a cut.

3.2 Cardinality Constrained (1, $\hat{p}_k$)-Configuration Inequalities

The (1, k)-Configuration inequality is a well-known inequality credited to Padberg [14] that is used to strengthen knapsack constraints. In that seminal paper, Padberg [14] also highlights conditions under which the (1, k)-Configuration inequality is a facet of the knapsack polytope. Since in this paper, k refers to cardinality, to avoid confusion, the same inequality is henceforth referred to as (1, p)-configuration inequality. The principal idea behind the (1, p)-configuration inequality is the following. For some 'knapsack' with a capacity of C, let $S \subset N$ denote a set of items that can all be accommodated in the knapsack. However, there exists a 'big' item q not in S, that certainly cannot be accommodated along with S. In fact, the item q is so big that only all subsets $P(p\text{-}1) \subset S$, each consisting of p-1 items can be accommodated along with q. That is, all subsets of S consisting of at least p items cannot be accommodated along with q, while they can be without q. This idea is captured in the (1, p)-configuration inequality. To present this formally, for some concentrator i, let $S \subset N$, with $|S| = s$ and $q \in N \backslash S$, be such that

(i) $\sum_{j \in S} d_j \leq C_i$, (22)

(ii) for all sets $P \subseteq S$, with $|P| = p$ and $2 \leq p \leq s$, $\sum_{j \in P \cup \{q\}} d_j > C_i$, but (23)

(iii) for all sets $P(p\text{-}1) \subset S$, with $|P(p\text{-}1)| = p$-1, $\sum_{j \in P(p-1) \cup \{q\}} d_j \leq C_i$. (24)

For some $R_{ik}(r_i) \subseteq S$, with $|R_{ik}(r_i)| = r_i$ satisfying $p \leq r_i \leq s$, the (1, p)-configuration inequality that is valid for $H(x, y)$ is,

$$(r_i - p + 1)x_{iq} + \sum_{j \in R_{ik}(r_i)} x_{ij} \leq r_i y_i. \quad (25)$$

The (1, p)-configuration inequality (25) essentially enforces the condition that in the absence of q, concentrator i can accommodate all terminals in $R_{ik}(r_i)$. However, in the presence of terminal q, at most (p-1) terminals in $R_{ik}(r_i)$ can be accommodated.

We now present a generalization of the (1, p)-configuration inequality by enforcing cardinality requirements in **($\mathbf{P}_{z\text{-}y}$)** as follows. For some concentrator-cardinality combination i-k, consider $S \subset N$ such that $|S| = s \leq k$ and a $q \in N\text{-}S$ that satisfies

i) $\sum_{j \in S} d_j + V^*(N - S, k - s) \leq C_i$, (26)

ii) for every $\hat{P}_k \subseteq S$, with $|\hat{P}_k| = \hat{p}_k$, $\sum_{j \in \hat{P}_k \cup \{q\}} d_j + V^*(N\text{-}\hat{P}_k\text{-}q, k\text{-}\hat{p}_k\text{-}1) > C_i$, (27)

iii) but for all $\hat{P}_k(\hat{p}_k - 1) \subset S$, $|\hat{P}_k(\hat{p}_k - 1)| = \hat{p}_k$-1,
$\sum_{j \in \hat{P}_k(\hat{p}_k - 1) \cup \{q\}} d_j + V^*(N\text{-}\hat{P}_k(\hat{p}_k - 1)\text{-}q, k\text{-}\hat{p}_k) \leq C_i$. (28)

Observe that (26), (27) and (28) are cardinality constrained equivalents of (22), (23) and 24), respectively. For some concentrator i, it is possible to accommodate all terminals in S, as well as k-s

additional terminals. However, there exists a terminal q, which if assigned to i, then it is not possible to assign all terminals in S, as well as (k-s-1) terminals from the remaining set $\{N\text{-}S\text{-}q\}$. In fact, only all subsets $\hat{P}_k(\hat{p}_k - 1) \subset S$ containing $\hat{p}_k$-1 elements along with q and (k-$\hat{p}_k$) from the remaining set $\{N\text{-}\hat{P}_k(\hat{p}_k - 1)\text{-}q\}$, can be accommodated.

Proposition 3. Consider a set $S \subset N$ and a terminal $q \in \{N\text{-}S\}$, that satisfies (26), (27) and (28) for every $R_{ik}(r_i) \subseteq S$ with $\hat{p}_k \leq r_i \leq s$. Then the following (1, $\hat{p}_k$)-configuration inequality is valid for $H(z, y)$:

$$(r_i - \hat{p}_k + 1)z_{iqk} + \sum_{j \in R_{ik}(r_i)} z_{ijk} \leq r_i y_{ik}. \quad (29)$$

Proof: The validity of (29) is established by showing that all feasible solutions in **($\mathbf{P_{z\text{-}y}}$)** satisfy it. Clearly, $y_{ik} = 0$ and $z_{ijk} = 0$ for all $j \in N$ satisfies (29). Next, consider the partial solution set, $y_{ik} = 1$, $z_{ijk} = 1$, for each $j \in R_{ik}(r_i)$, $z_{iqk} = 0$. This solution is feasible due to (26), and of course satisfies (29) exactly. Finally, consider the partial solution set: $y_{ik} = 1$, $z_{ijk} = 1$ for $j \in \hat{P}_k(\hat{p}_k - 1)$ for every $\hat{P}_k(\hat{p}_k - 1) \subset N_i$ and $z_{iqk} = 1$. Such a solution is feasible since it satisfies (28). It also satisfies (29). □

Proposition 4 below shows that the (1, $\hat{p}_k$)-configuration inequalities provide a tighter description of $H(z, y)$ than (1, p)-configuration inequalities do for $H(x, y)$.

Proposition 4. Let i) $LP_{1p}(x, y) = \{(x, y) \in LP(x, y) |$ (x, y) satisfies all (1, p)-configuration inequalities (25)$\}$, and ii) $LP_{1\hat{p}_k}(z, y) = \{z, y) \in LP(z, y) |$ (z, y) satisfies all (1, $\hat{p}_k$)-configuration inequalities (29)$\}$. Further, let $LP_{a,1\hat{p}_k}(x, y) = \{(x, y) \in R^{mn+m} |$ (x, y) satisfies (13) for each $(z, y) \in LP_{1\hat{p}_k}(z, y)\}$. Then, $LP_{a,1\hat{p}_k}(x, y) \subseteq LP_{1p}(x, y)$.

Proof: Observe that one (1, p)-configuration inequality (25) is constructed for each $R_{ik}(r_i) \subseteq S$. On the other hand, several (1, $\hat{p}_k$)-configuration inequalities (29) are constructed from the same set $R_{ik}(r_i)$, one for each $k \geq r_i$. If $R_{ik}(r_i) = S$ while $k = s = r_i$, then it is clear from (27) that $V^*(N\text{-}\hat{P}_k\text{-}q, k\text{-}\hat{p}_k\text{-}1) = 0$. In such a case, $\hat{p}_k$ in (29) is exactly equal to p in (25). However, for values of $k > r_i$, $V^*(N\text{-}\hat{P}_k\text{-}q, k\text{-}\hat{p}_k\text{-}1) > 0$, and therefore $\hat{p}_k \leq p$. For values of $k < r_i$, by setting $\hat{p}_k = p$ in (29), the resulting inequality will be valid. That is because, the maximum value the left-hand-side of (29) can take is $k < r_i$. By aggregating constraints (29) for a given set $R_{ik}(r_i)$ over $k = 1, \ldots, K_i$, we obtain

$$\sum_{k=1}^{r_i} (r_i - p + 1)z_{iqk} + \sum_{k=r_i+1}^{K_i} (r_i - \hat{p}_k + 1)z_{iqk} + \sum_{k=1}^{K_i} \sum_{j \in R_{ik}(r_i)} z_{ijk} \leq \sum_{k=1}^{K_i} r_i y_{ik}. \quad (30)$$

Let $LP_{1\hat{p}_k - agg}(z, y) = \{z, y) \in LP(z, y) |$ (z, y) satisfies all (1, $\hat{p}_k$)-configuration inequalities (30)$\}$ and $LP_{b,1\hat{p}_k}(x, y) = \{(x, y) \in R^{mn+m} |$ (x, y) satisfies (13) for each $(z, y) \in LP_{1\hat{p}_k - agg}(z, y)\}$. Since (30) is an aggregation of (29), it follows that $LP_{1\hat{p}_k}(z, y) \subseteq LP_{1\hat{p}_k - agg}(z, y)$ and therefore $LP_{a,1\hat{p}_k}(x, y) \subseteq LP_{b,1\hat{p}_k}(x, y)$. In (30), due to (13), the term $\sum_{k=1}^{K_i} r_i y_{ik}$ can be replaced by $r_i y_i$ and $\sum_{k=1}^{K_i} \sum_{j \in R_{ik}(r_i)} z_{ijk}$ can be replaced by $\sum_{j \in R_{ik}(r_i)} x_{ij}$. Further, since $\hat{p}_k \leq p$ for $k = r_i + 1, \ldots, K_i$, it follows from (13) that $\sum_{k=1}^{r_i} (r_i - p + 1)z_{iqk} + \sum_{k=r_i+1}^{K_i} (r_i - \hat{p}_k + 1)z_{iqk} \geq \sum_{k=1}^{K_i} (r_i - p + 1)x_{iq}$. Therefore, $LP_{b,1\hat{p}_k}(x, y) \subseteq LP_{1p}(x, y)$. The result follows. □

The following example best illustrates the strength of the (1, $\hat{p}_k$)-configuration inequality vis-à-vis the (1, p)-configuration inequality.

Example 2. Consider a concentrator i with $C_i = 60$ and terminal demands in sorted order as [22, 15, 15, 14, 14, 13, 12, 8, 8, 7, 5, 4, 4, 4, 3, 3]. For $S = \{2, \ldots, 7\}$ and $q = 1$, a (1, p)-configuration inequality is: $2x_{i1} + x_{i2} + x_{i3} + x_{i4} + x_{i5} + x_{i6} + x_{i7} \leq 4y_i$. That is because the total demand of the 4 smallest concentrators in S is 14+14+13+12 = 53, i.e. $r_i = 4$. However, in the presence of $q = 1$, at most p -1 = 2 concentrators from

S can be accommodated. Consider the case of $k = 6$. Since $V^*(N-S,2) = 6$ and $V^*(N-S,3) = 10$, the $(1, \hat{p}_6)$-configuration inequality is $2z_{i16}+z_{i26}+z_{i36}+z_{i4k}+z_{i56}+z_{i66}+z_{i76} \leq 4y_{i6}$, implying that in the absence of $q = 1$, there is space for 4 items in S, while in the presence of $q = 1$, there is space for at most 2 items in S. For $k = 7$, the $(1, \hat{p}_k)$-configuration inequality is $2z_{i1,7}+z_{i2,7}+z_{i3,7}+z_{i4,7}+z_{i5,7}+z_{i6,7}+z_{i7,7} \leq 3y_{i7}$. The partial LP solution (z, y): $y_{i6} = 0.5$, $z_{i2,6} = z_{i3,6} = 0.25$, $z_{i4,6} = 0.5$, $z_{i5,6} = 0.4$, $z_{i6,6} = 0.5$, $z_{i13,6} = 0.1$, $z_{i14,6} = z_{i15,6} = 0.5$, $y_{i7} = 0.5$, $z_{i1,7} = z_{i2,7} = 0.25$, $z_{i5,7} = z_{i6,7} = z_{i12,7} = z_{i13,7} = z_{i14,7} = z_{i15,7} = 0.5$, satisfies constraints (5)-(9). However, the $(1, \hat{p}_k)$-configuration inequalities listed above for $k = 6$ and for $k = 7$, are both violated. At the same time, the (x, y) solution obtained by aggregating as described in (13) is: $y_i = 1.0$, $x_{i1} = 0.25$, $x_{i2} = 0.5$, $x_{i3} = 0.25$, $x_{i4} = 0.5$, $x_{i5} = 0.9$, $x_{i6} = 1.0$, $x_{i12} = 0.5$, $x_{i13} = 0.6$, $x_{i14} = x_{i15} = 1.0$, which satisfies the $(1, p)$-configuration inequality listed above. □

The separation heuristic proposed to identify violated $(1, \hat{p}_k)$-configuration inequalities is briefly described next. An i-k combination for which $y_{ik} > 0$, along with its support, $S_{ik} = \{j \in N | z_{ijk} > 0\}$ such that $|S_{ik}| = s_{ik} > k$ is first identified. The terminal q is obtained as $d_q = Max\ \{d_j | j \in S_{ik}\}$. Let, $R_{ik} = S_{ik}$-q. The set S and $\hat{p}_k$ in (29) is determined from R_{ik} as follows. First, R_{ik} sorted in increasing order, in terms of $g_j(y_{ik_i}, z_{ijk_i}, d_j) = (y_{ik} - z_{ijk} + 0.1)/d_j$. To obtain $\hat{p}_k$, the smallest subset $R_{ik}(\hat{p}_k) \subset R_{ik}$ consisting of the first $\hat{p}_k$ elements in R_{ik} that satisfy $\sum_{j \in R_{ik}(\hat{p}_k)} d_j + V^*(N - R_{ik}(\hat{p}_k) - q, k - \hat{p}_k - 1) > C_i - d_q$ is obtained. Having identified $R_{ik}(\hat{p}_k)$, $S = R_{ik}(\hat{p}_k)$. Next, S is expanded to include those items $j^* \in \{R_{ik}\text{-}R_{ik}(\hat{p}_k)\}$ that satisfy $V^*(S, \hat{p}_k - 1) + d_{j^*} + V^*(N - R_{ik}(\hat{p}_k - 1) - j^* - q, k - \hat{p}_k - 1) > C_i - d_q$, as well as $\sum_{j \in S} d_j + d_{j^*} + V^*(N - S - j^*, k - s - 1) \leq C_i$. Whenever this happens, $S = S \cup j^*$ and $s = s+1$. This process of expanding S continues till all items in $\{R_{ik}\text{-}R_{ik}(\hat{p}_k)\}$ have been explored or that $s = k$. Having obtained S and $\hat{p}_k$, several $(1, \hat{p}_k)$-configuration inequalities (29) can be constructed, each associated with a unique set $R_{ik}(r_i) \subseteq S$, for all $\hat{p}_k \leq r_i \leq s$, where $|R_{ik}(r_i)| = r_i$. Amongst these, those that the LP solution violates are added as cuts.

3.3 2-Facility Cardinality Matching Inequality

The 2-Facility Cardinality Matching inequality presented in this section is derived for the un-capacitated version of **($\mathbf{P_{z\text{-}y}}$)**, i.e., one without knapsack constraints (7). Of course, by definition, this inequality is valid for **($\mathbf{P_{z\text{-}y}}$)** as well, albeit weaker. However, as will be shown later, it can be lifted to become a strong inequality for **($\mathbf{P_{z\text{-}y}}$),** and the lifting procedure is quick. Intuitively, this inequality accounts for how terminals are assigned to concentrators (or facilities) in a way that the cardinalities associated with terminals are matched to that of concentrators.

A 2-Facility Cardinality Matching inequality is constructed around a pair of concentrators $W = \{i_1, i_2\}$ and a designated set of 4 terminals $H_q = \{j_{q1}, j_{q2}, j_{q3}, j_{q4}\}$ selected from N. Further, for concentrators i_1 and i_2, respective cardinalities k_1 and k_2 are identified such that is when the LP relaxation of **($\mathbf{P_{z\text{-}y}}$)** is solved initially, $0 < y_{i_1 k_1} < 1$ and $0 < y_{i_2 k_2} < 1$. The construction of the 2-Facility Cardinality Matching inequality begins by aggregating constraints in (8) over i and k_i, resulting in

$$\sum_{i \in M} \sum_{j \in N} \sum_{k_i=1}^{K_i} z_{ijk_i} = \sum_{i \in M} \sum_{k_i=1}^{K_i} k_i y_{ik_i}. \tag{31}$$

First, a set of z variables associated with $j \in H_q$ are selectively removed from left-hand-side of (31). They are: a) $z_{ij_{q1}k_i}$ with $i=i_1$ and all $1 \le k_{i_1} \le k_1 - 1$, $i = i_2$ with $k_2 + 1 \le k_{i_2} \le K_{i_2}$, and with $i \in M\text{-}W$, $1 \le k_i \le K_i$, b) $z_{ij_{q2}k_i}$ with $i=i_2$ and all $1 \le k_{i_2} \le k_2$, $i = i_1$ with $k_1 \le k_{i_1} \le K_{i_1}$, and with $i \in M\text{-}W$, $1 \le k_i \le K_i$, c) $z_{ij_{q3}k_i}$ with $i=i_1$ and all $1 \le k_{i_1} \le k_1 - 1$, $i=i_2$ with $1 \le k_{i_2} \le k_2$, and with $i \in M\text{-}W$, $1 \le k_i \le K_i$, d) $z_{ij_{q4}k_i}$ with $i=i_1$ and all $k_1 \le k_{i_1} \le K_{i_1}$, $i=i_2$ with $k_2 + 1 \le k_{i_2} \le K_{i_2}$, and with $i \in M\text{-}W$, $1 \le k_i \le K_i$. We refer to these missing z variables as 'hidden' assignments. Second, the right-hand-side of (31) is modified wherein the coefficients of variables $y_{i_1k_{i_1}}$ and $y_{i_2k_{i_2}}$ are decreased by 1, for all $1 \le k_{i_1} \le K_{i_1}$ and $1 \le k_{i_2} \le K_{i_2}$. Finally, a constant 1 is added to the right-hand-side. This results in the following 2-Facility Cardinality Matching inequality:

$$\sum_{j \in N-\{j_{q1}, j_{q3}\}} \sum_{k_{i_1}=1}^{k_1-1} z_{i_1 j k_{i_1}} + \sum_{j \in N-\{j_{q2}, j_{q4}\}} \sum_{k_{i_1}=k_1}^{K_{i_1}} z_{i_1 j k_{i_1}} + \sum_{j \in N-\{j_{q2}, j_{q3}\}} \sum_{k_{i_2}=1}^{k_2} z_{i_2 j k_{i_2}} + \sum_{j \in N-\{j_{q2}, j_{q4}\}} \sum_{k_{i_2}=k_2+1}^{K_{i_2}} z_{i_2 j k_{i_2}} + \sum_{i \in M-W} \sum_{j \in \{N-H_q\}} \sum_{k_i=1}^{K_i} z_{ij k_i} \le \sum_{k_{i1}=1}^{K_{i_1}} (k_{i_1} - 1) y_{i_1 k_{i1}} + \sum_{k_{i2}=1}^{K_{i_2}} (k_{i_2} - 1) y_{i_2 k_{i2}} + \sum_{i \in M-W} \sum_{k_i=1}^{K_i} k_i y_{ik_i} + 1 \quad (32)$$

Figure 3 below illustrates some of those hidden assignments.

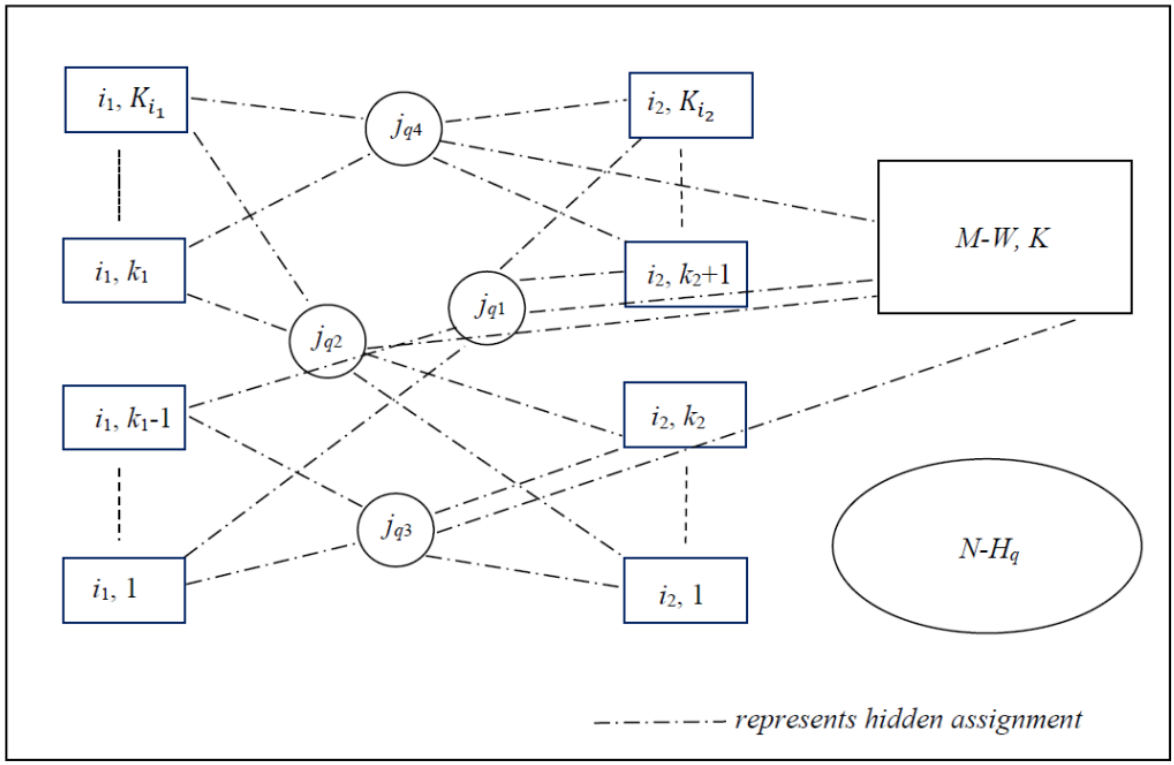

Figure 3. Illustration of hidden assignments in 2-Facility Cardinality Matching Inequality

Proposition 5. Let $HU(z, y) = Conv\{(z, y) \in R^w |$ (5), (6), (8)-(10), $w = \sum_{i=1}^{m}(n+1)K_i\}$, that is the convex hull of the un-capacitated version of **($P_{z\text{-}y}$)**. Given a pair of concentrators $W = \{i_1, i_2\}$ whose

respective, designated cardinalities $\{k_1, k_2\}$, a subset $H_q = \{j_{q1}, j_{q2}, j_{q3}, j_{q4}\}$ of terminals whose assignments are hidden as described above, the 2-Facility Cardinality Matching inequality (32) is valid for $HU(z, y)$.

Proof: We simply show that (32) is satisfied by all categories of feasible solutions to the un-capacitated version of $(\mathbf{P_{z\text{-}y}})$, i.e., one without the knapsack constraints (7).

Case I: Consider feasible solutions where no concentrator in W is used. Here, (32) reduces to $\sum_{i\in M-W}\sum_{j\in\{N-H_q\}}\sum_{k_i=1}^{K_i} z_{ijk_i} \leq \sum_{i\in M-W}\sum_{k_i=1}^{K_i} k_i y_{ik_i} + 1$. Due to (6), (8) and (9), all such feasible solutions satisfy it.

Case II: Consider feasible solutions in which $y_{i_1k_{i_1}} = y_{i_2k_{i_2}} = 1$, where $1 \leq k_{i_1} \leq k_1$-1 and k_2+1 $\leq k_{i_2} \leq K_{i_2}$. Observe that in this instance, all possible assignments of j_{q1} is hidden. Hence, the left-hand-side value of (32) is at most n-1, with n-k_{i_1}-k_{i_2} terminals assigned to concentrators in M-W. The right-hand-side is (k_{i_1}-1) + (k_{i_2}-1)+(n-k_{i_1}-k_{i_2})+1 = n-1. Hence, (32) is satisfied.

Case III: Consider instances where $y_{i_1k_{i_1}} = y_{i_2k_{i_2}} = 1$, with $k_1 \leq k_{i_1} \leq K_{i_1}$ and $1 \leq k_{i_2} \leq k_2$. Here, all z variables that describe the assignment of j_{q2} to any concentrator is missing. Therefore, here as well, the left-hand-side value of (32) is at most n-1, while the right-hand-side is (k_{i_1}-1)+(k_{i_2}-1)+(n-k_{i_1}-k_{i_2})+1 = n-1. Therefore, (32) is satisfied.

Case IV: Consider the case of $y_{i_1k_{i_1}} = y_{i_2k_{i_2}} = 1$, with $1 \leq k_{i_1} \leq k_1$-1 and $1 \leq k_{i_2} \leq k_2$. Here, j_{q3} is the terminal whose assignment is missing and therefore the left-hand-side of (32) is at most n-1. However, as with Case II and Case III, the right-hand-side of (32) will still be (k_{i_1}-1)+(k_{i_2}-1)+(n-k_{i_1}-k_{i_2})+1 = n-1 and therefore satisfy (32).

Case V: The next case is when $y_{i_1k_{i_1}} = y_{i_2k_{i_2}} = 1$, with $k_1 \leq k_{i_1} \leq K_{i_1}$ and k_2+1 $\leq k_{i_2} \leq K_{i_2}$. Here, j_{q4} is the terminal whose assignment is missing and therefore the left-hand-side of (32) is at most n-1, while the right-hand-side will be equal to n-1 and therefore (32) is satisfied.

Case VI: The last possible case is when exactly either $y_{i_1k_{i_1}} = 1$ or $y_{i_2k_{i_2}} = 1$, but not both. Observe that in this case, regardless of the value of k_{i_1} or k_{i_2}, the assignment of exactly two terminals in H_q are hidden. For instance, when $y_{i_1k_{i_1}} = 1$ with $1 \leq k_{i_1} \leq k_1$-1, then the assignment of j_{q1} and j_{q3} are hidden, while if $k_1 \leq k_{i_1} \leq K_{i_1}$ then the assignment of j_{q2} and j_{q4} are hidden. When $y_{i_2k_{i_2}} = 1$, then if $1 \leq k_{i_2} \leq k_2$, then the assignment of j_{q2} and j_{q3} are hidden, while if k_2+1 $\leq k_{i_2} \leq K_{i_2}$, then the assignment of j_{q1} and j_{q4} are hidden. Thus, in all such instances the left-hand-side is at most n-2, while the right-hand-side is n. Hence (32) is satisfied. □

Example 3. Consider the problem instance $W = \{i_1, i_2\}$, $k_1 = 3$, $k_2 = 3$, $K_{i_1} = 4$, $K_{i_2} = 5$, $H_q = \{j_{q1}, j_{q2}, j_{q3}, j_{q4}\} = \{2, 3, 4, 5\}$. The partial LP solution that satisfies the un-capacitated version of $(\mathbf{P_{z\text{-}y}})$ is: $y_{i_1,3} = y_{i_12} = y_{i_23} = y_{i_24} = 0.5$, $z_{i_1,2,3} = z_{i_1,4,3} = z_{i_1,3,2} = z_{i_1,5,2} = z_{i_11,3} = 0.5$, $z_{i_2,1,4} =$

$z_{i_2,3,4} = z_{i_2,4,4} = z_{i_2,6,4} = z_{i_2,2,3} = z_{i_2,5,3} = z_{i_2,6,3} = 0.5$. This partial solution is illustrated in Figure 3 below. Observe that this solution satisfies constraints (5), (6), (8) and (9). However, it violates the 2-Facility Cardinality Matching inequality where only the variables that are not zero are listed below:

$$z_{i_1 1,3} + z_{i_1,2,3} + z_{i_1,4,3} + z_{i_1,3,2} + z_{i_1,5,2} + z_{i_2,1,4} + z_{i_2,3,4} + z_{i_2,4,4} + z_{i_2,6,4} + z_{i_2,2,3} + z_{i_2,5,3} + z_{i_2,6,3}$$
$$+ \sum_{i \in M-W} \sum_{j \in \{N-H_q\}} \sum_{k_i=1}^{K_i} z_{ijk_i}$$
$$\leq 2y_{i_1,3} + y_{i_1 2} + 2y_{i_2 3} + 3y_{i_2 4} + \sum_{i \in M-W} \sum_{k_i=1}^{K_i} k_i y_{ik_i} + 1. \qquad (33)$$

It is clear from Figure 4 below that the fractional solution listed above violates (33) by 1.

In an accompanying paper [12], we describe in detail the strength of the 2-Facility Cardinality Matching inequality. In that paper we show that the 2-Facility Cardinality Matching inequality is a facet of the polytope $HU(z, y)$ defined in Proposition 5. In addition, we show that for the special case of $m = 2$, the 2-Facility Cardinality Matching inequalities, along with constraints (5), (7), (8) and (9), completely describes $HU(z, y)$. While the 2-Facility Cardinality Matching inequality is a non-trivial facet of $HU(z, y)$, it need not be so for $H(z, y)$. This is due to the presence of knapsack constraints (7). The inequality (32) can be strengthened by using a sequential lifting procedure on the missing z variables in it. The details on this procedure can be found in [12].

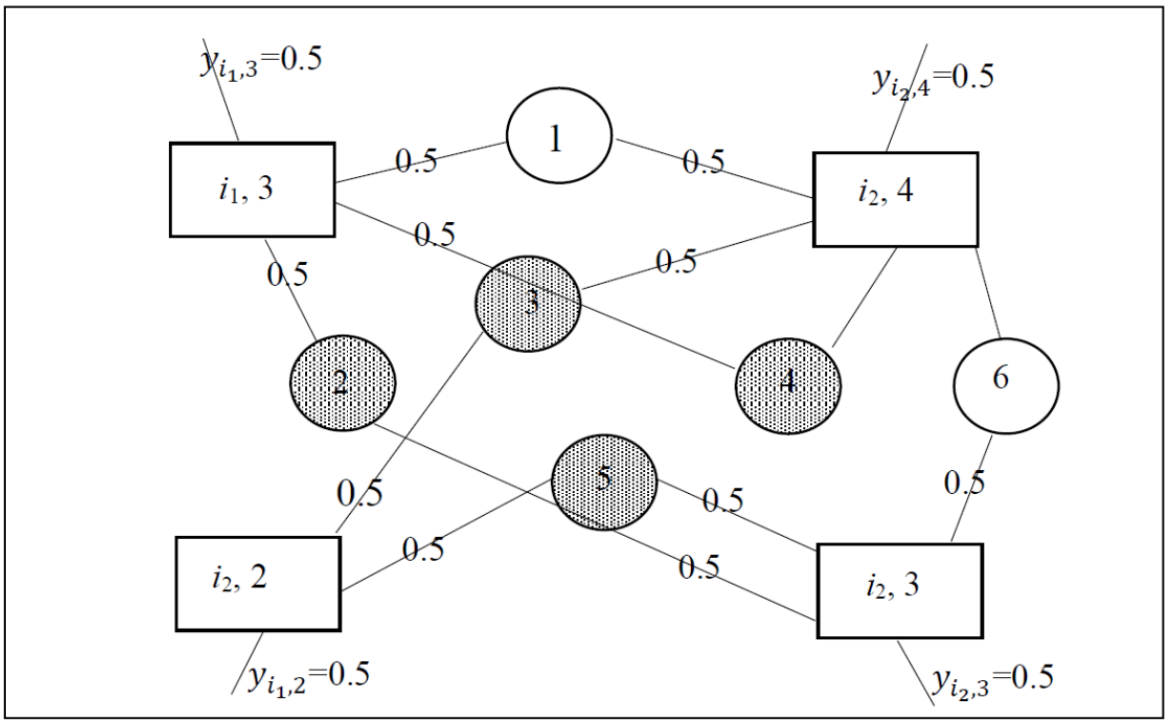

Figure 4. Illustration of Example 3.

The separation heuristic employed to identify a 2-Facility Cardinality Matching inequality (32) that the current LP solution violates is as follows. First, a pair $W = \{i_1, i_2\}$ with respective cardinalities k_1 and k_2 is identified such that from the LP solution, $0 < \bar{y}_{i_1 k_1} < 1$ and $0 < \bar{y}_{i_2 k_2} < 1$. In addition, $0 < \bar{y}_{i_1 k_1 - 1} < 1$ and $0 < \bar{y}_{i_2 k_2 + 1} < 1$. Given such an LP solution, for each $j \in N$, the sum $SUM_{q1}(j) = \sum_{k_{i_1}=1}^{k_1-1} \bar{z}_{i_1 j k_{i_1}} + \sum_{k_{i_2}=k_2+1}^{K_{i_2}} \bar{z}_{i_2 j k_{i_2}} + \sum_{i \in M-W} \sum_{k=1}^{K_i} \bar{z}_{ijk}$ is computed, followed by $MINSUM_{q1} = Min_{j \in N}\{SUM_{q1}(j)\}$.

Terminal j_{q1} is the designated terminal that corresponds to $MINSUM_{q1}$. Clearly, $MINSUM_{q1} = 0$ implies that the assignment of j_{q1} is hidden from (i_1, k_{i_1}) for $1 \le k_{i_1} \le k_1 - 1$, (i_2, k_{i_2}) for $k_2 + 1 \le k_{i_2} \le K_{i_2}$, and (i, k_i) for all $i \in M\text{-}W$ and all cardinalities. Similarly, $SUM_{q2}(j) = \sum_{k_{i_1}=k_1}^{K_{i_1}} \bar{z}_{i_1 j k_{i_1}} + \sum_{k_{i_2}=1}^{k_2} \bar{z}_{i_2 j k_{i_2}} + \sum_{i \in M-W} \sum_{k=1}^{K_i} \bar{z}_{ijk}$, followed by $MINSUM_{q2} = Min_{j \in N}\{SUM_{q2}(j)\}$ determines j_{q2}. To determine j_{q3}, the sum used is $SUM_{q3}(j) = \sum_{k_{i_1}=1}^{k_1-1} \bar{z}_{i_1 j k_{i_1}} + \sum_{k_{i_2}=1}^{k_2} \bar{z}_{i_2 j k_{i_2}} + \sum_{i \in M-W} \sum_{k=1}^{K_i} \bar{z}_{ijk}$, with j_{q3} corresponding to $MINSUM_{q3} = Min_{j \in N}\{SUM_{q3}(j)\}$. Finally, j_{q4} corresponds to $MINSUM_{q4} = Min_{j \in N}\{SUM_{q4}(j)\}$ where $SUM_{q4}(j) = \sum_{k_{i_1}=k_1}^{K_{i_1}} \bar{z}_{i_1 j k_{i_1}} + \sum_{k_{i_2}=k_2+1}^{K_{i_2}} \bar{z}_{i_2 j k_{i_2}} + \sum_{i \in M-W} \sum_{k=1}^{K_i} \bar{z}_{ijk}$.

If $MINSUM = MINSUM_{q1} + MINSUM_{q2} + MINSUM_{q3} + MINSUM_{q4} < 1$, then (32) is added as a cut.

4.0 Computational Results

In this section we present a detailed computational study that compares the performance of our proposed branch-and-cut algorithm that uses the disaggregated formulation **($\mathbf{P}_{z\text{-}y}$)** along with the valid inequalities described in this paper to a generic branch-and-cut algorithm applied to the traditional formulation **($\mathbf{P}_{x\text{-}y}$)**. The proposed branch & cut method has been developed in CPLEX 12.7, compiled with GNU g++ 4.4.5 (with -O3 optimization option) and ran single-threaded on a machine with 8 processors (4 cores, 2.2 GHz), each with 16 GB of RAM, under a i686 GNU/Linux operating system. The separation algorithms for the valid inequalities along with the disaggregated model have been implemented in CPLEX using the call-backs.

Computationally, a potential disadvantage of using **($\mathbf{P}_{z\text{-}y}$)** over **($\mathbf{P}_{x\text{-}y}$)** is its large size, even though the increase is polynomial in nature. This can be of significant concern for large problems. In fact, since each K_i in **($\mathbf{P}_{z\text{-}y}$)** can potentially be as large as n, for a problem with 200 terminals, the formulation in **($\mathbf{P}_{z\text{-}y}$)** would be about 200 times larger than that in **($\mathbf{P}_{x\text{-}y}$)**, both in terms of variables and constraints. This in turn has an impact on the time taken to solve LP relaxations in a branch-and-bound framework. To alleviate the computational burden of solving a large linear program, we address this issue 'locally' as follows. First, the LP relaxation of **($\mathbf{P}_{x\text{-}y}$)** is solved. Next, we measure $\chi_i = \sum_{j \in N} x_{ij}$, which in some sense, represents the 'number' of terminals assigned to concentrator i, in the LP solution. Therefore, χ_i for each $i \in M$ can be fractional. The y and x variable for each i are now split into a prespecified number of cardinalities or levels. In this paper, we tested our approach with these prespecified levels L_i being 3, 7 and 11. To illustrate, with $L_i = 3$, the three cardinalities used are: i) $k_{i1} = \lfloor \chi_i \rfloor - 1$, ii) $k_{i2} = \lfloor \chi_i \rfloor$ and iii) $k_{i3} = \lfloor \chi_i \rfloor + 1$ for each $i \in M$. Accordingly, the variable sets $\{y_{ik_{i1}}, y_{ik_{i2}}, y_{ik_{i3}}\}$ and $\{z_{ijk_{i1}}, z_{ijk_{i2}}, z_{ijk_{i3}}\}$ are defined for each $i \in M$. Using these variables, constraints (6) and (7) can be written as:

$$z_{ijk_l} \le y_{ik_l} \qquad \forall\, i \in M, j \in N, l = i1, i2, i3 \tag{35}$$

$$\sum_{j \in N} d_j z_{ijk_l} \le C_i y_{ik_l} \qquad \forall i \in M, l = i1, i2, i3. \tag{36}$$

Finally, the cardinality constraints (8) take the form:

$$\sum_{j \in N} z_{ijk_{i1}} \le k_{i1} y_{ik_{i1}}, \qquad \forall\, i \in M \tag{37}$$

$$\sum_{j \in N} z_{ijk_{i2}} = k_{i2} y_{ik_{i2}}, \qquad \forall\, i \in M \tag{38}$$

$$\sum_{j \in N} z_{ijk_{i3}} \geq k_{i3} y_{ik_{i3}}. \qquad \forall\, i \in M \qquad (39)$$

In the case of $L_i = 7$, a maximum of 3 levels below and above $\lfloor \chi_i \rfloor$ are defined, while ensuring that no level is below 1 and no level is above K_i. For instance, if for some $i \in M$, $\lfloor \chi_i \rfloor = 3$, then two levels below $\lfloor \chi_i \rfloor$ are specified. Similarly, if $K_i = 4$, then only one level above $\lfloor \chi_i \rfloor$ is specified. It is worth observing that the use of constraints (37), (38) and (39), in place of (8), makes the resulting LP relaxation, weaker than the relaxation of $\mathbf{(P_{z\text{-}y})}$. However, as our computational results show, the slight loss in the lower bound is more than made up by a dramatic reduction in the size of the formulation and consequently the time taken to solve the LP relaxations. It is also worth mentioning that the inequalities described in this paper apply to all levels except the lowest level.

The branch-and-cut approaches described above were tested on 36 medium to large size problem instances. Details on how to generate these instances can be found in [3]. The problem size instances can be found in Table 3. The 1st eight instances have sizes of $|M| = 16$ and $|N| = 50$, followed by the next eight instances having sizes of $|M| = 25$ and $|N| = 50$. The 3rd set of eight problems comprised of instances with sizes of $|M| = 50$ and $|N| = 50$. Finally, we tested a set of 12 problem instances with sizes of $|M| = 100$ and $|N| = 1000$.

Table 1. Comparison of bounds obtained at the root node between the Traditional formulation and Disaggregated Formulation with $L_i = 3$

Problems	*Traditional Formulation*					*Disaggregated Formulation - $L_i = 3$*					
	before cutting		*after cutting*			*before cutting*		*after cutting*			
# Name	Time (sec)	GAP	Time (sec)	#CPX	GAP	Time (sec)	GAP	Time (sec)	#user	#CPX	GAP
Pr.VI – cap61	0.00	7.7497%	0.01	50	0.0536%	0.01	0.0000%	0.01	0	0	0.0000%
Pr.VI – cap62	0.00	10.6211%	0.01	71	0.0852%	0.01	0.0000%	0.01	0	0	0.0000%
Pr.VI – cap63	0.00	12.4061%	0.01	85	0.0148%	0.01	0.1011%	0.05	1	5	0.0135%
Pr.VI – cap64	0.00	13.3786%	0.03	113	0.0799%	0.02	0.7154%	0.09	31	6	0.1060%
Pr.VII – cap71	0.00	10.3600%	0.00	60	0.0000%	0.01	0.0000%	0.01	0	0	0.0000%
Pr.VII – cap72	0.00	15.0625%	0.01	74	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.VII – cap73	0.00	18.2687%	0.01	92	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.VII – cap74	0.00	20.1189%	0.01	105	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.IX – cap91	0.00	17.0337%	0.01	80	0.0000%	0.01	0.0000%	0.01	0	0	0.0000%
Pr.IX – cap92	0.00	22.6502%	0.01	106	0.0000%	0.02	0.3534%	0.04	2	3	0.0000%
Pr.IX – cap93	0.00	25.3742%	0.02	136	0.2386%	0.02	0.6567%	0.06	2	6	0.2319%
Pr.IX – cap94	0.00	27.4029%	0.03	155	0.1439%	0.02	0.8873%	0.07	4	3	0.1619%
Pr.X – cap101	0.00	20.7767%	0.01	82	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.X – cap102	0.00	28.6275%	0.01	101	0.0000%	0.02	0.0000%	0.02	0	0	0.0000%
Pr.X – cap103	0.00	33.5287%	0.01	141	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%

Pr.X – cap104	0.00	37.2816%	0.01	166	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.XII – cap121	0.00	21.5199%	0.02	187	0.0000%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.XII – cap122	0.00	27.1854%	0.02	254	0.0000%	0.01	0.3547%	0.06	2	3	0.0000%
Pr.XII – cap123	0.00	29.9184%	0.05	300	0.0531%	0.01	0.4339%	0.14	2	6	0.0224%
Pr.XII – cap124	0.00	32.0601%	0.03	300	0.9703%	0.02	0.9007%	0.12	4	4	0.1927%
Pr.XIII – cap131	0.00	25.6435%	0.02	195	0.0004%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.XIII – cap132	0.00	33.7876%	0.03	259	0.0000%	0.02	0.0000%	0.03	0	0	0.0000%
Pr.XIII – cap133	0.00	39.2374%	0.04	300	0.0389%	0.01	0.0000%	0.02	0	0	0.0000%
Pr.XIII – cap134	0.00	43.1707%	0.03	300	1.3070%	0.02	0.0000%	0.02	0	0	0.0000%
Pr.A – capa1	0.21	39.2222%	85.87	3279	5.3554%	373.50	2.1672%	373.50	0	0	2.1672%
Pr.A – capa2	0.20	57.0841%	273.20	3276	6.0115%	218.02	3.1074%	218.02	0	0	3.1074%
Pr.A – capa3	0.16	71.6843%	506.51	3252	5.7933%	146.79	1.8431%	146.79	0	0	1.8431%
Pr.A – capa4	0.14	83.7912%	478.48	3273	2.1163%	36.98	0.0010%	41.05	0	2	0.0006%
Pr.B – capb1	0.16	35.8444%	27.56	3297	2.5636%	197.75	0.3750%	197.75	0	0	0.3750%
Pr.B – capb2	0.15	49.9976%	83.62	3295	4.8634%	157.77	0.4060%	157.77	0	0	0.4060%
Pr.B – capb3	0.14	63.3218%	161.31	3296	5.0810%	135.45	0.3331%	135.45	0	0	0.3331%
Pr.B – capb4	0.12	75.4272%	187.14	3260	5.1424%	82.49	0.0929%	82.49	0	0	0.0929%
Pr.C – capc1	0.13	45.7482%	35.19	3291	3.6659%	90.35	0.5361%	125.48	0	2	0.5350%
Pr.C – capc2	0.13	57.8666%	60.58	3274	4.1103%	64.03	0.4205%	78.59	0	2	0.4085%
Pr.C – capc3	0.12	68.9805%	100.27	3290	3.5686%	43.05	0.0920%	52.25	0	2	0.0797%
Pr.C – capc4	0.11	79.5210%	136.89	3283	3.3855%	26.49	0.0500%	33.47	0	2	0.0404%

In Table 1, the results present a comparison of the performance of the disaggregated formulation with $L_i = 3$ and that of the traditional formulation. Specifically, the gap between the best lower bound and upper bound in percentage terms, is measured at two points in time – once before any cuts are added to the respective LP relaxations, and once after all possible cuts are added at the root node of the branch-and-bound tree. Also measured are the total CPU time in seconds at both these points in time. In the case of the traditional formulation, CPLEX generates a variety of cuts using its separation heuristics. The number of such cuts added for each instance are reported under the heading #CPX. In the case of the disaggregated formulation, we first look for violated $(n_i, k_i, \hat{r}_{ik_i})$-*cover*, $(1, \hat{p}_k)$-configuration and 2-Facility Cardinality Matching inequalities. The number of such cuts added to the disaggregated formulation is reported under the heading #user.

A feature that stands out from the results in Table 1 is that the bounds obtained from the initial LP relaxation of the traditional formulation is poor. The percentage gap varies from 7.75% to 79.52%. In contrast, bounds obtained from the initial LP relaxation of the disaggregated formulation with $L_i = 3$ are very tight. In fact, in 16 of the 36 problem instances, the optimal integer solution was obtained from the initial LP relaxation. Even among the remaining 20 instances, in all but 3 of those instances, the percentage gap was found to be less than 1%. Consequently, for the traditional formulation, a large

number of cuts was generated by CPLEX. These numbers varied from 50 for the smallest instance to more than 3000 cuts for all the big problem instance. Computationally, this translates to that many re-optimisations of the resulting LPs. In contrast, very few cuts were generated for the disaggregated formulation. With $L_i = 3$, all the cuts generated were of the type, $(n_i, k_i, \hat{r}_{ik_i})$-*cover*, $(1, \hat{p}_k)$-configuration, besides the cuts generated by CPLEX. Because of the large number of CPLEX cuts generated for the traditional formulation, the overall time taken at the root node for the traditional formulation is of the same order of magnitude as that for the disaggregated formulation. This in spite of the disaggregated formulation being much larger than the traditional formulation. Finally, the percentage gap obtained after all the cuts were added at the root node was found to be smaller for the disaggregated formulation as compared to the traditional formulation in 33 of the 36 instances. More importantly, for the large problem instances, percentage gap was found to be significantly higher for the traditional formulation as compared to the disaggregated formulation.

In Table 2, the same set of measurements that were made for Table 1, are reported for the disaggregated formulation with $L_i = 7$ and $L_i = 11$. Clearly, the disaggregated model sizes are larger with $L_i = 7$ and with $L_i = 11$. As against $L_i = 3$, both for $L_i = 7$ and $L_i = 11$, a few more $(n_i, k_i, \hat{r}_{ik_i})$-*cover* and $(1, \hat{p}_k)$-configuration as well as a few 2-Facility Cardinality matching inequalities were generated. However, the overall improvement in the lower bound was found to be marginal, given that in most cases, the lower bounds were already very close to the optimal value. Of course, the time taken to solve the LP relaxations for $L_i = 7$ and $L_i = 11$ are much more.

Table 2: Bounds obtained at the root node for the Disaggregated Formulation with $L_i = 7$ and $L_i = 11$.

Problems	*Disaggregated Formulation – $L_i = 7$*						*Disaggregated Formulation – $L_i = 11$*					
	before cutting		*after cutting*				*before cutting*		*after cutting*			
# Name	Time (sec)	GAP	Time (sec)	#user	#CPX	GAP	Time (sec)	GAP	Time (sec)	#user	#CPX	GAP
Pr.VI – cap61	0.04	0.0000%	0.04	0	0	0.0000%	0.02	0.0000%	0.03	0	0	0.0000%
Pr.VI – cap62	0.04	0.0000%	0.05	0	0	0.0000%	0.03	0.0000%	0.03	0	0	0.0000%
Pr.VI – cap63	0.04	0.1011%	0.17	3	8	0.0135%	0.05	0.1011%	0.15	3	9	0.0117%
Pr.VI – cap64	0.05	0.7151%	0.23	67	2	0.1141%	0.06	0.7151%	0.64	40	2	0.1265%
Pr.VII – cap71	0.03	0.0000%	0.04	0	0	0.0000%	0.03	0.0000%	0.03	0	0	0.0000%
Pr.VII – cap72	0.03	0.0000%	0.03	0	0	0.0000%	0.03	0.0000%	0.04	0	0	0.0000%
Pr.VII – cap73	0.04	0.0000%	0.04	0	0	0.0000%	0.03	0.0000%	0.04	0	0	0.0000%
Pr.VII – cap74	0.03	0.0000%	0.04	0	0	0.0000%	0.03	0.0000%	0.04	0	0	0.0000%
Pr.IX – cap91	0.02	0.0000%	0.02	0	0	0.0000%	0.03	0.0000%	0.04	0	0	0.0000%
Pr.IX – cap92	0.02	0.3534%	0.09	18	4	0.0000%	0.03	0.3521%	0.17	6	3	0.0000%
Pr.IX – cap93	0.03	0.6567%	0.11	16	5	0.2395%	0.06	0.6555%	0.22	8	8	0.2386%
Pr.IX – cap94	0.06	0.8873%	0.22	20	3	0.0594%	0.11	0.8840%	0.40	10	5	0.1624%

Pr.X – cap101	0.02	0.0000%	0.03	0	0	0.0000%	0.03	0.0000%	0.04	0	0	0.0000%
Pr.X – cap102	0.02	0.0000%	0.03	0	0	0.0000%	0.03	0.0000%	0.05	0	0	0.0000%
Pr.X – cap103	0.02	0.0000%	0.03	0	0	0.0000%	0.03	0.0000%	0.05	0	0	0.0000%
Pr.X – cap104	0.02	0.0000%	0.04	0	0	0.0000%	0.04	0.0000%	0.06	0	0	0.0000%
Pr.XII – cap121	0.03	0.0000%	0.04	0	0	0.0000%	0.04	0.0000%	0.06	0	0	0.0000%
Pr.XII – cap122	0.04	0.3547%	0.21	18	3	0.0000%	0.06	0.3534%	0.33	6	3	0.0000%
Pr.XII – cap123	0.05	0.4339%	0.31	18	6	0.0224%	0.08	0.4327%	0.52	8	8	0.0224%
Pr.XII – cap124	0.07	0.9007%	0.42	20	2	0.0818%	0.16	0.8947%	0.70	10	5	0.1927%
Pr.XIII – cap131	0.03	0.0000%	0.04	0	0	0.0000%	0.04	0.0000%	0.06	0	0	0.0000%
Pr.XIII – cap132	0.03	0.0000%	0.05	0	0	0.0000%	0.06	0.0000%	0.08	0	0	0.0000%
Pr.XIII – cap133	0.04	0.0000%	0.05	0	0	0.0000%	0.06	0.0000%	0.08	0	0	0.0000%
Pr.XIII – cap134	0.05	0.0000%	0.06	0	0	0.0000%	0.07	0.0000%	0.10	0	0	0.0000%
Pr.A – capa1	1995.86	2.1672%	1988.94	0	0	2.1672%	-----	-----	-----	-----	-----	-----
Pr.A – capa2	1098.61	3.1074%	1148.92	0	0	3.1074%	-----	-----	-----	-----	-----	-----
Pr.A – capa3	753.91	1.8431%	789.24	0	0	1.8431%	1518.21	1.8431%	1518.21	0	0	1.8431%
Pr.A – capa4	205.82	0.0010%	210.67	0	2	0.0006%	158.46	0.0010%	375.30	0	2	0.0006%
Pr.B – capb1	1192.25	0.3750%	1191.88	0	0	0.3750%	2598.40	0.3750%	2598.40	0	0	0.3750%
Pr.B – capb2	798.36	0.4060%	855.19	0	0	0.4060%	2080.06	0.4060%	2080.06	0	0	0.4060%
Pr.B – capb3	575.48	0.3331%	634.25	0	0	0.3331%	1318.68	0.3331%	1318.68	0	0	0.3331%
Pr.B – capb4	432.83	0.0929%	467.15	0	0	0.0929%	973.98	0.0929%	973.98	0	0	0.0929%
Pr.C – capc1	410.37	0.5361%	603.31	0	3	0.5086%	750.57	0.5361%	1558.26	0	3	0.5204%
Pr.C – capc2	337.14	0.4205%	340.28	0	1	0.4138%	829.35	0.4205%	829.35	0	0	0.4205%
Pr.C – capc3	232.28	0.0920%	258.18	0	0	0.0920%	339.66	0.0920%	505.53	0	1	0.0863%
Pr.C – capc4	179.74	0.0500%	186.18	0	0	0.0500%	244.33	0.0500%	434.58	0	1	0.0311%

Table 3 presents results on the same set of 36 problem instances in which the branch-and-cut procedure using the traditional formulation is compared to the branch-and-cut procedure using the disaggregated formulation with $L_i = 3$. In both approaches, the tolerance limit on the gap between the lower bound and the incumbent solution was set at e^{-6}. In addition, both the procedures were aborted, as soon as the time taken reached 1 hour. In the table, the total number of branch-and-bound nodes visited is reported under column labelled *#nodes*. It is clear from the table that the performance of the branch-and-cut procedure using the disaggregated formulation is found to be superior to the branch-and-cut procedure using the traditional formulation, both in terms of time taken and the number of branch-and-bound nodes visited. What is indeed significant is that the dominance of our proposed procedure becomes even more pronounced for large problem instances. For instance, in problem labelled "Pr.A – capa4", our procedure solved the problem in 199.35 seconds using 101 branch-and-bound node visits. In contrast, in the traditional formulation, the total time taken was 2160.63 seconds with 746 branch-and-bound node visits. In the case of problem instance PrC – capc2, our branch-and-cut procedure took 298.24

seconds with 78 branch-and-bound node visits, while the branch-and-cut procedure using the traditional formulation took 3104.63 seconds with 19070 branch-and-bound node visits. In 6 of the 12 large problem instances, the branch-and-cut procedure using the traditional formulation had to be aborted before solving to optimality as time limit of 3600 seconds was reached. In contrast, our proposed procedure had to be aborted in just 2 of the 12 instances. In these two problem instances, the percentage gap obtained from our procedure was found to be smaller than that obtained using the traditional formulation. In summary, the computational results presented confirm the value of using the disaggregated formulation along with its associated valid inequalities for solving large size and difficult problem instances to optimality over the traditional formulation.

Table 3. Comparison in terms the total time taken and the # of branch-and-bound nodes visited for the Traditional formulation versus Disaggregated Formulation with $L_i = 3$

Problems			*Traditional Formulation*				*Disaggregated Formulation - $L_i = 3$*				
# Name	*M*	*N*	*Time* (sec)	*#CPX*	*#nodes*	*GAP*	*Time* (sec)	*#user*	*#CPX*	*#nodes*	*GAP*
Pr.VI – cap61	16	50	0.02	42	12	0.0000%	0.01	0	0	1	0.0000%
Pr.VI – cap62	16	50	0.02	48	1	0.0000%	0.01	0	0	1	0.0000%
Pr.VI – cap63	16	50	0.04	74	25	0.0000%	0.07	1	5	3	0.0000%
Pr.VI – cap64	16	50	0.05	110	54	0.0000%	0.32	6	9	6	0.0000%
Pr.VII – cap71	16	50	0.01	68	1	0.0000%	0.01	0	0	1	0.0000%
Pr.VII – cap72	16	50	0.01	75	1	0.0000%	0.01	0	0	1	0.0000%
Pr.VII – cap73	16	50	0.01	98	1	0.0000%	0.02	0	0	1	0.0000%
Pr.VII – cap74	16	50	0.01	98	1	0.0000%	0.01	0	0	1	0.0000%
Pr.IX – cap91	25	50	0.02	83	1	0.0000%	0.02	0	0	1	0.0000%
Pr.IX – cap92	25	50	0.04	73	6	0.0000%	0.04	2	3	1	0.0000%
Pr.IX – cap93	25	50	0.05	77	21	0.0000%	0.50	2	8	52	0.0000%
Pr.IX – cap94	25	50	0.07	149	59	0.0000%	0.49	2	6	40	0.0000%
Pr.X – cap101	25	50	0.01	98	1	0.0000%	0.01	0	0	1	0.0000%
Pr.X – cap102	25	50	0.02	102	1	0.0000%	0.02	0	0	1	0.0000%
Pr.X – cap103	25	50	0.01	154	1	0.0000%	0.02	0	0	1	0.0000%
Pr.X – cap104	25	50	0.02	159	1	0.0000%	0.02	0	0	1	0.0000%
Pr.XII – cap121	50	50	0.08	167	1	0.0000%	0.02	0	0	1	0.0000%
Pr.XII – cap122	50	50	0.08	232	1	0.0000%	0.06	2	3	1	0.0000%
Pr.XII – cap123	50	50	0.11	20	1	0.0000%	0.24	2	6	6	0.0000%
Pr.XII – cap124	50	50	0.78	300	1193	0.0000%	0.86	2	9	67	0.0000%
Pr.XIII – cap131	50	50	0.03	243	1	0.0000%	0.02	0	0	1	0.0000%
Pr.XIII – cap132	50	50	0.05	217	1	0.0000%	0.02	0	0	1	0.0000%
Pr.XIII – cap133	50	50	0.07	272	9	0.0000%	0.02	0	0	1	0.0000%

Pr.XIII – cap134	50	50	0.12	294	210	0.0000%	0.02	0	0	1	0.0000%
Pr.A – capa1	100	1000	3584.89*	5500	17703	2.0647%	3598.88*	0	2	353	1.8325%
Pr.A – capa2	100	1000	3589.94*	5500	6858	4.3752%	3590.82*	0	0	1278	3.0066%
Pr.A – capa3	100	1000	3585.66*	5500	7808	2.6201%	1882.17	0	2	568	0.0000%
Pr.A – capa4	100	1000	2160.63	5160	746	0.0001%	199.35	0	5	101	0.0001%
Pr.B – capb1	100	1000	2827.60	5495	25220	0.0001%	858.43	0	122	1054	0.0001%
Pr.B – capb2	100	1000	3599.00*	5480	14626	0.1236%	1313.42	0	31	587	0.0001%
Pr.B – capb3	100	1000	3585.30*	5500	9871	1.1452%	649.10	0	41	509	0.0001%
Pr.B – capb4	100	1000	3591.17*	5500	10299	1.0689%	261.35	0	89	846	0.0001%
Pr.C – capc1	100	1000	1165.49	5179	7724	0.0001%	485,17	0	100	716	0,0001%
Pr.C – capc2	100	1000	3104.63	5485	19070	0.0000%	298,24	0	0	78	0,0000%
Pr.C – capc3	100	1000	810.56	5497	2888	0.0001%	202,39	0	36	334	0,0001%
Pr.C – capc4	100	1000	707.75	5500	1734	0.0001%	118,08	0	1	14	0,0000%

***** Procedure aborted as time limit of 3600 seconds was reached.

For better assessing the behaviour of the proposed formulation, we have compared the results provided by our algorithm with those presented in [9], where a Lagrangian heuristic for the standard integer programming formulation P(x-y) is implemented. Constraints (1) are the relaxed ones. We have adopted the test problems described in Table 1 of [9]. They are a total of 71 problems, grouped in five subsets, characterized by different sizes and different quotients between the total capacity and the total demand.

In the following Table 4 we report, for each of the five groups, in column LH the average of the relative gap between the upper and lower bounds obtained by the Lagrangian heuristics [9] and in column DIS.FORM the average of the relative gap at the root node provided by our disaggregated formulation, corresponding to $L_i = 7$. What is indeed clear from the results in Table 4 is that the gap between the upper and lower bounds obtained using DIS.FORM is uniformly smaller than the corresponding gap obtained using the Lagrangian Heuristic reported in [9]. This in spite of the fact that the Lagrangian heuristic involves solving a series of sub-problems which are the potentially hard knapsack problems.

Table 4. Gap comparison with Lagrangian Heuristics [9].

Problems			LH	Dis.. Form.– Li = 7
Group	M	N	GAP	GAP
p1-p12	10	50	0,1192%	0,0667%
p13-p24	20	50	0,1608%	0,0989%
p25-p40	30	150	0,6956%	0,6824%
p41-p55	10-30	70-100	0,3753%	0,2788%
p56-p71	30	200	0,6063%	0,3813%

5. Concluding Remarks and Future Research Possibilities

In this paper, we present a new extended formulation of CCLP that uses the idea of cardinality associated with each concentrator. Even though the resulting formulation is bigger in terms of the number of variables and constraints, we show theoretically that it is indeed a stronger formulation. That is, its LP relaxation provides a tighter lower bound than the LP relaxation of the traditional formulation. In addition, we show that our proposed disaggregated formulation reveals generalizations of the Cover and (1, k)-Configuration inequalities which collectively are stronger than the original Cover and (1, k)-Configuration inequalities. Finally, we present another class of inequalities called: 2-Facility Cardinality Matching Inequalities, which are specific to the disaggregated formulation. We first present results of extensive computational tests on 36 benchmark problems which are medium to large sized whose characteristics can be found in [3]. These results confirm the value of using our proposed disaggregated formulation of CCLP. We were able to demonstrate that our approach was able to identify the optimal solution at the root node itself in most of the reasonable sized instances. For the much larger sized test problems, the proposed branch-and-cut procedure using the disaggregated formulation outperforms the branch-and-cut procedure applied to the traditional formulation by a significant order both in terms of CPU and the number of branches required to solve the problem to optimality. We also compared our approach to the Lagrangian heuristic approach reported by Holmberg et al. [9] on the same set of 71 test problems reported by them. Our computational results clearly show that much tighter lower bounds are obtained using the disaggregated formulation along with its associated cuts, than those obtained using the Lagrangian heuristic.

The results in this paper, both theoretical and computational, demonstrate an approach that seems very promising. However, much remains to be done to take the idea of cardinality-based disaggregation to its logical conclusion. From a theoretical standpoint, it is worthwhile extending the idea behind 2-Facility Cardinality Matching inequality to more than two concentrators. It is also worthwhile exploring the idea of generalizing the cardinality constrained Cover and (1, $\hat{p}_k$)-configuration inequalities to more than one facility or concentrator. Finally, the ideas presented in this paper can be applied to several other closely related NP-Hard problems such as the Capacitated Steiner Tree problem and the Capacitated Network Design Problem.

References

1. Ahuja R.K., J.B. Orlin, S. Pallottino, M.P. Scaparra, and M.G. Scutella, "A multi-exchange heuristic for the single source capacitated facility location. *Management Science*, (50), (2004), pp. 1749–760.
2. Barcelo, J., A. Hallefjord, E. Fernandez and K. Jornsten, "Lagrangean Relaxation and Constraint Generation Procedures for the Capacitated Plant Location Problems with Single Sourcing," *OR Spektrum* 12, (1990), pp. 79-88.
3. Beasley, J. E., "An Algorithm for Solving Large Capacitated Warehouse Location Problems," *European Journal of Operational Research* 33, (1988), pp. 314-325.
4. Boffey, T. B., "Location Problems Arising in Computer Networks," *Journal of Operational Research Society*, 40(4), (1989), pp. 347-354.

5. Celani, M., R. Cerulli, M. Gaudioso, and Ya.D. Sergeyev, "A Multiplier Adjustment Technique for the Capacitated Concentrator Location Problem," *Optimization Methods and Software*, 10(11), (1998), pp. 87-102.
6. Chen, C.H. and CJ. Ting, "Combining Lagrangian Heuristic and Ant Colony System to solve the Single Source Capacitated Facility Location Problem," *Transportation Research part E*, 44, (2008), pp. 1099–1122.
7. Cortinhal, M.J. and M. E. Captivo, "Upper and Lower Bounds for the Single Source Capacitated Location Problem," *European Journal of Operational Research* 151, (2003), pp. 333-351.
8. Gouveia, L. and F. Saldanha-da-Gama, "On the Capacitated Concentrator Location Problem: A Reformulation by Discretization". *Computers & Operations Research*, 33, (2006), pp. 1242-1258.
9. Holmberg, K., Rönnqvist, M., and Yaun Di, "An Exact Algorithm for the Capacitated Facility Location Problems with Single Sourcing," *European Journal of Operational Research* 113, (1999), pp. 544-559.
10. Labbè, M. and H. Yaman, "Polyhedral analysis for concentrator location problems", *Computational Optimization and Applications*, 34, (2006) pp. 377–407.
11. Lo, C. and A. Kershenbaum, "A Two-phased Algorithm and Performance Bounds for the Concentrator Location Problem", *IEEE Transactions on Communications*, 37(1989), pp. 1151-1163.
12. Murthy, I., "Facets of the Cardinality Induced Disaggregated Formulation of the Capacitated Concentrator Location Problem," Working Paper (2018), IIM Bangalore, Bangalore, India.
13. Martello, S, D. Pisinger and P. Toth, "New Trends in Exact Algorithms to Solve 0-1 Knapsack Problem," *European Journal of Operational Research*, 123, (2000), pp. 325–332.
14. Padberg, M., "(1, *k*)-Configurations and Facets of packing Problems," *Mathematical Programming* 18(1) 1980, pp. 94-99.
15. Pirkul, H., "Efficient Algorithms for the Capacitated Concentrator Location Problem", *Computers and Operations Research*, 14 (1987), pp. 197-208.
16. Pirkul, H. and V. Nagarajan, "Locating Concentrators in Centralized Computer Networks", *Annals of Operations Research*, 36(1992) 247-261.
17. Shima, T., "Integer Programming Model and Exact Solution for Concentrator Location Problem," *Journal of Operations Research Society of Japan*, 43, (2000), pp. 291-305.
18. Sridharan, R., "A Lagrangean Heuristic for the Capacitated Plant Problem with Single Source Constraints", *European Journal of Operational Research*, 66, (1993), pp. 305–312.
19. Yang, Z, Feng Chu and Haoxun Chen, "A Cut-and-Solve based Algorithm for the Single-source Capacitated Facility Location Problem," *European Journal of Operational Research*, 221 (2012), pp. 521-532.