A Combined Capacity Scaling and Local Branching Approach to Capacitated Multi-commodity Network Design Problem

Naoto Katayama
Ryutsu Keizai University, Distribution and Logistics Systems, katayama@rku.ac.jp, http://www.rku.ac.jp/∼katayama/

The capacitated multi-commodity network design problem (CMND) represents a generic network design model for applications used in designing construction and improvements to telecommunication, logistics, transportation, distribution, and production networks. This paper presents an approach combining capacity scaling and local branching for CMND. Capacity scaling is an approximate iterative solution approach for capacitated network problems based on the principle of changing arc capacities that depend on flow volumes on arcs. Local branching is a method of solving new restricted problems based on an exploration of solution neighborhoods defined as local branching constraints. Capacity scaling with a strong path flow based formulation including forcing constraints can produce high-quality solutions within a short period of time allotted for computation. By combined capacity scaling and local branching, the combined approach can offer one of the best current solutions compared to previous heuristics for CMND.

Key words: Capacitated Network Design; Capacity Scaling; Local Branching
History:

1. Introduction

The capacitated multi-commodity network design problem (CMND), also known as the capacitated fixed charge multi-commodity network flow problem, represents a generic network design model for applications used in designing construction and improvements to telecommunications, logistics, transportation, distribution, and production networks. For network design problems, many real-life applications can be found in Magnanti et al. (1986), as well as Powell and Sheffi (1989). The CMND solution provides the appropriate network design as well as routes of multi-commodity flows aimed at minimizing total cost, which is the sum of flow costs and fixed costs over the network with limited arc capacities. CMND is formulated as a mixed integer programming problem. Binary variables are used to model a network design selecting arcs from a candidate arc set appropriately, while continuous variables represent the volumes of arc flows on the network.

Several reviews on network design problems and associated methods of identifying solutions are described in Magnanti and Wong (1984), Wong (1984), Wong (1985), Minoux (1989), Balakrishnan et al. (1997), Gendron et al. (1997), Crainic (2003), Costa (2005), and Yaghini and Akhavan (2012).

CMND is known as an NP-hard problem (Magnanti and Wong 1984). Many techniques have been developed to accommodate these problems, such as valid inequalities, relaxation methods, heuristics, and meta-heuristics.

Polyhedral approaches to CMND have also been developed. Magnanti et al. (1993) proposed the integer rounding cut-set, as well as three partition and arc residual capacity inequalities, while Barahona (1996) proposed multi-cut inequalities. Also, Bienstock and Günlük (1996) developed cut-set, flow-cut-set, and three partition facets. Atamtürk and Rajan (2002) proposed the polyhedron of single arc-set along with related separation and lifting problems. Chouman et al. (2003) proposed cover and minimum cardinality inequalities, while Kliewer and Timajev (2005) developed cover inequalities and local cuts. Costa et al. (2009) developed Benders, metric and cut-set inequalities. Recently, Chouman et al. (2009) developed flow cover and flow pack inequalities, plus a method used to generate violated inequalities efficiently.

Relaxation and lower bound approaches have also been devised for solving CMND. Katayama and Kasugai (1993) presented a dual ascent approach based on integer rounding cut-set inequalities. Gendron and Crainic (1994, 1996) meanwhile presented linear relaxation and Lagrangian relaxation problems. Herrmann et al. (1996) proposed an extension method employing a dual ascent algorithm to address an uncapacitated multi-commodity network design problem. Holmberg and Yuan (2000) proposed a combination algorithm incorporating a Lagrangian relaxation method and a branch-and-bound algorithm. Crainic et al. (2001) developed a sub-gradient method employing a bundle type algorithm.

Heuristics and meta-heuristics designed to identify feasible approximate solutions within a reasonable computation time have also been developed. Gendron and Crainic (1994, 1996) proposed a resource decomposition heuristic based on a resource-directive decomposition algorithm. In the last decade, several tabu search heuristics have been created. Crainic et al. (2000) and Zaleta and Socarrás (2004) proposed a simplex-based tabu search heuristic. Ghamlouche et al. (2003) proposed a cycle-based tabu search heuristic including simplex-based search. Crainic and Gendron (2002) proposed a cooperative parallel tabu search heuristic, while Crainic et al. (2006) developed a multilevel cooperative search heuristic. Ghamlouche et al. (2004), Alvarez et al. (2005), and Crainic and Gendreau (2007) explored a scatter search algorithm and a path relinking algorithm to CMND. Scatter search and path relinking are related to a tabu search heuristic in the sense that they provide unifying principles for joining solutions based on generalized path constructions and by utilizing strategic designs. Meanwhile, Crainic et al. (2004) proposed slope scaling heuristics. Slope scaling is based on changing flow costs, which depend on arc flow volumes and dual variable information.

In the last few years, combined or hybrid approaches characterized by meta-heuristics and a mixed-integer programming (MIP) solver have been developed. Katayama et al. (2009) proposed capacity scaling using column generation and row generation techniques. Rodríguez-Martín and Salazar-Gonzáleza (2010) developed a local branching heuristic, which utilizes an MIP solver to explore neighborhoods. Hewitt et al. (2010) proposed a combined exact and heuristic approach to search very large neighborhoods by solving restricted problems. Ghamlouche et al. (2011) proposed learning mechanisms and local search heuristics, while Yaghini and Rahbar (2011) proposed a hybrid simulated annealing as well as a simplex method.

In many papers, an arc flow based formulation or a path flow based formulation excluding forcing constraints have been used to address CMND. Since the formulation including forcing constraints is a large mixed integer programming problem, significant amounts of computation time are required to solve large and complex problems and their relaxation problems. However, in order to improve solutions and the lower bound derived from linear relaxation problems, the optimal mean of solving formulations is to apply forcing constraints. Column generation for path flow variables and cutting planes for forcing constraints can reduce the problem size and increase solvability of the problem.

This paper presents capacity scaling using column generation and cutting plane techniques for solving CMND. Capacity scaling is an approximate iterative solution approach for capacitated network problems based on changing arc capacities, which in turn depend on flow volumes on arcs. Although capacity scaling produces good solutions within a reasonable computation time in general, this process may not always yield high-quality solutions. Consequently, local branching is applied for solutions derived from capacity scaling. Local branching consists of a method of solving a new problem with the same constraints and objective function as the original problem, but with the addition of local branching constraints, based on an exploration of solution neighborhoods. A combined capacity scaling and local branching approach can produce the best solutions compared to previous approaches found in related literature, for nearly all benchmark problems.

2. Mathematical Formulation

Let $G = (N, A)$ be a directed network with the set of nodes N and the set of directed arcs A. Let K be the set of commodities on the network. For each commodity $k \in K$, let P^k be the set of paths of commodity k, and d^k the demand of flow of commodity k from its single origin node to its single destination node.

The following measures characterize arc $(i, j) \in A$: f_{ij}, the fixed cost of including arc (i, j) in the network design, c_{ij}^k, the unit variable flow cost for commodity k flowing on arc (i, j), and C_{ij}, the limited arc capacity which is shared by all the commodities flowing on the arc.

The formulation of CMND has two types of variables. The first type is a binary design variable, which is defined as $y_{ij} = 1$ if arc (i, j) is included in the network design, $y_{ij} = 0$ otherwise. The

second type is a continuous path flow variable, which is defined by x_p^k, representing the amount of the path flow of commodity k flowing on the path $p \in P^k$. Let δ_{ij}^p be the constant, $\delta_{ij}^p = 1$ if arc (i,j) is included in path p, $\delta_{ij}^p = 0$ otherwise.

The path flow based formulation CMNDP of CMND can be expressed as follows:

(CMNDP)

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k \sum_{p\in P^k} \delta_{ij}^p x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{1}$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k \quad \forall k \in K, \tag{2}$$

$$\sum_{k\in K} \sum_{p\in P^k} \delta_{ij}^p x_p^k \le C_{ij} y_{ij} \quad \forall (i,j) \in A, \tag{3}$$

$$\sum_{p\in P^k} \delta_{ij}^p x_p^k \le d^k y_{ij} \quad \forall (i,j) \in A, k \in K, \tag{4}$$

$$x_p^k \ge 0 \quad \forall p \in P^k, k \in K, \tag{5}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A. \tag{6}$$

The objective function (1) is the total cost, that is the sum of variable flow costs of commodities plus the sum of fixed costs in a given network design, and should be minimized. Constraints (2) consist of flow conservation equations, which represent the fact that the sum of path flows of commodity k is equal to the demand. Constraints (3) provide the capacity constraints, which prohibit flowing if the arc is excluded, $y_{ij} = 0$, and allow for flow up to the arc capacity if the arc is included, $y_{ij} = 1$. Constraints (4) provide the forcing constraints, which prohibit flowing of commodity k if the arc is excluded, and allow for flow up to the demand if the arc is included.

Let x_{ij}^k be the arc flow variable for commodity k flowing on arc (i,j). Let N_n^+ be the set of outward nodes from node n, and N_n^- be the set of inward nodes into node n. The arc flow based formulation CMNDA of CMND can be expressed as follows:

(CMNDA)

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{7}$$

subject to

$$\sum_{i\in N_n^+} x_{in}^k - \sum_{j\in N_n^-} x_{nj}^k = \begin{cases} -d^k & if\ n = O^k \\ d^k & if\ n = D^k \\ 0 & otherwise \end{cases} \quad \forall n \in N,\ k \in K, \tag{8}$$

$$\sum_{k\in K} x_{ij}^k \le C_{ij} y_{ij} \quad \forall (i,j) \in A, \tag{9}$$

$$x_{ij}^k \le d^k y_{ij} \quad \forall (i,j) \in A, k \in K, \tag{10}$$

Figure 1 **X-y Correlations in a Linear Relaxation Problem**

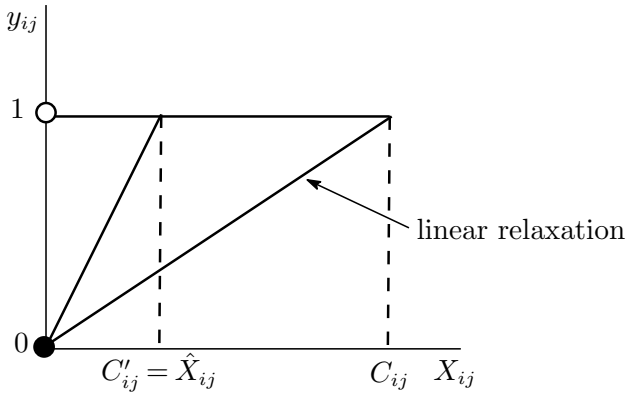
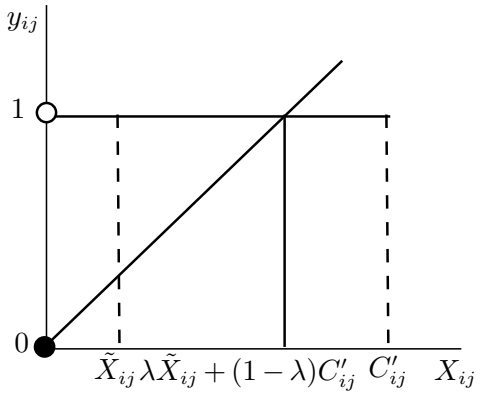


Figure 2 **Changing Capacity**



$$x_{ij}^k \geq 0 \quad \forall (i,j) \in A, k \in K, \tag{11}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A. \tag{12}$$

Constraints (8) are the flow conservation equations, which represent relations between the sum of arc flows into node n and the sum of arc flows out from node n for commodity k. Constraints (9) are the capacity constraints, and constraints (10) are the forcing constraints.

In this paper, capacity scaling using column generation is applied for path flow based formulation CMNDP, and local branching is applied to arc flow based formulation CMNDA.

3. Capacity Scaling

Capacity scaling represents an approximate iterative solution approach for capacitated network problems based on changing arc capacities, which depend on flow volumes on arcs (Pochet and Vyve 2004). When solving the linear relaxation problem of CMNDP, the capacity constraints (3), which define a variable upper bound on design variable y_{ij} for arc flow $X_{ij}(=\sum_{k\in K}\sum_{p\in P^k}\delta_{ij}^p x_p^k)$, is approximated by a linear function (Figure 1). At the linear relaxation problem, the optimal design variables may not necessarily comprise a binary, but a decimal fraction in a solution and arc capacities of the right hand of equation (3) are underestimated all over the domain. As a consequence, a relaxation solution may not be an approximation for finding a good feasible binary solution for CMND.

If the optimal arc flow $\hat{X}$ of CMNDP is found, we should change capacity C to C', which is equal to $\hat{X}$ for each arc. Then we solve the linear relaxation problem with capacity C' again. As the result, either 0 or 1 solutions for all design variables can be obtained. CMNDP associated with all fixed design variables is reduced to a multi-commodity flow problem. By solving the multi-commodity flow problem, we obtain the optimal objective value of CMNDP. It is a given that finding the optimal flow of CMNDP is extremely difficult, yet identifying the optimal flow is our main purpose. Consequently, if a near-optimal flow can be identified, C' may be estimated and

a good approximate solution could possibly be derived from it. On the other hand, by changing capacity C' a little bit at a time, we can gradually identify near-optimal flow.

Capacity scaling starts by solving the linear relaxation problem of CMNDP associated with $C^l(=C')$ as a substitution for C at the iteration l. Initially we set $C^1 := C$. The linear relaxation problem LR(C^l) with capacity C^l at iteration l can be expressed as follows:

(LR(C^l))

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k \sum_{p\in P^k} \delta_{ij}^p x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{13}$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k \quad \forall k \in K, \tag{14}$$

$$\sum_{k\in K} \sum_{p\in P^k} \delta_{ij}^p x_p^k \le C_{ij}^l y_{ij} \quad \forall (i,j) \in A, \tag{15}$$

$$\sum_{p\in P^k} \delta_{ij}^p x_p^k \le d^k y_{ij} \quad \forall (i,j) \in A, k \in K, \tag{16}$$

$$x_p^k \ge 0 \quad \forall p \in P^k, k \in K, \tag{17}$$

$$0 \le y_{ij} \le C_{ij}/C_{ij}^l \quad \forall (i,j) \in A. \tag{18}$$

In the right-hand side of the constraint (15), C_{ij} is replaced for C_{ij}^l, and in the right-hand side of the constraint (18), the upper bound 1 is replaced for C_{ij}/C_{ij}^l to enable arc flow up to its original capacity C_{ij}.

Let $\tilde{X}_{ij}$ be the optimal arc flow on arc (i,j) of LR(C^l). At the next iteration, we substitute C_{ij}^l with $\lambda\tilde{X}_{ij} + (1-\lambda)C_{ij}^{l-1}$ (Figure 2), where $\lambda(0 \le \lambda \le 1)$ is a smoothing parameter designed to prevent rapid changes. If all design variables converge to zero or one in the solution of LR(C^l) at some point of iteration, then we solve the multi-commodity network flow problem associated with all fixed design variables set to these convergent values, and both a feasible solution and an upper bound can be identified for the CMNDP. Obtaining convergent solutions may require numerous iterations; otherwise some design variables may not converge. Consequently, when most design variables converge to zero or to one by a threshold value ϵ, and the number of free design variables, which are not converged variables, is less than or equal to a branch-and-bound execution parameter B, the branch-and-bound algorithm of an MIP solver is applied for CMNDP associated with non-convergent free variables. Here, the upper bound may be identified. Let ΔB be subtracted from B. After the branch-and-bound algorithm, B is reduced by ΔB, until $B < B_{min}$. The capacity scaling stops when the iteration number exceeds the minimum iteration number ITE_{min} and an upper bound UB has been identified. Otherwise, if the iteration number exceeds the maximum iteration number ITE_{max}, capacity scaling also stops. An outline of capacity scaling proceeds as Algorithm 1.

Algorithm 1: Capacity Scaling

Set λ, ϵ, ITE_{min}, ITE_{max}, B, B_{min}, and ΔB;

$C^1 := C$; $UB := \infty$; $l := 1$;

repeat

Solve LR(C^l) by Column Generation and Cutting Plane;

Obtain the arc flow solution $\tilde{X}$ and the design solution $\tilde{y}$ of LR(C^l);

for $(i,j) \in A$ **do**

if $\tilde{y}_{ij} < \epsilon$ **then** $y_{ij} = 0$;

else if $\tilde{y}_{ij} > 1 - \epsilon$ **then** $y_{ij} = 1$;

else free;

end

if *the number of free variables of* y *is less than* B **then**

Solve CMNDP associated with the restricted design variable y by an MIP solver;

Get the objective function value Z_{CMNDP} and the design solution $\bar{y}$ of CMNDP;

$B := \max\{B_{min}, B - \Delta B\}$;

if $Z_{CMNDP} < UB$ **then** $UB := Z_{CMNDP}$;

$l := l + 1$;

for $(i,j) \in A$ **do**

$C_{ij}^l := \lambda \tilde{X}_{ij} + (1 - \lambda) C_{ij}^{l-1}$;

end

until *If* $l \geq ITE_{min}$ *and* $UB \neq \infty$, *or* $l \geq ITE_{max}$

4. Column Generation and Cutting Plane

In capacity scaling, the linear programming problem LR(C^l) is solved iteratively. Since LR(C^l) has an exponential number of path flow variables and the forcing constraints of the number of $O(|K||A|)$, all variables and constraints should not be considered at large problems explicitly. In order to solve large problems efficiently, column generation is applied for path flow variables, while cutting plane is applied for forcing constraints.

For each commodity k, let $\tilde{P}^k \subset P^k$ be the initial set of paths. $\tilde{AK} \subset A \times K$ is the set of index of generated forcing constraints. Initially, $\tilde{AK} := \phi$ and violated forcing constraints by solutions are generated by cutting plane.

We reformulate the restricted problem, which has restricted path sets $\tilde{P}$ and restricted forcing constraint set $\tilde{AK}$ for LR(C^l), as follows in RLR($C^l, \tilde{P}, \tilde{AK}$):

(RLR($C^l, \tilde{P}, \tilde{AK}$))

$$\min \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k \sum_{p\in \tilde{P}^k} \delta_{ij}^p x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{19}$$

subject to

$$\sum_{p\in\tilde{P}^k} x_p^k = d^k \quad \forall k \in K, \tag{20}$$

$$\sum_{k\in K}\sum_{p\in\tilde{P}^k} \delta_{ij}^p x_p^k \le C_{ij}^l y_{ij} \quad \forall (i,j) \in A, \tag{21}$$

$$\sum_{p\in\tilde{P}^k} \delta_{ij}^p x_p^k \le d^k y_{ij} \quad \forall (i,j,k) \in \tilde{AK}, \tag{22}$$

$$x_p^k \ge 0 \quad \forall p \in \tilde{P}^k, k \in K, \tag{23}$$

$$0 \le y_{ij} \le C_{ij}/C_{ij}^l \quad \forall (i,j) \in A. \tag{24}$$

Let s be the dual variable for the constraint (20), $u(\ge 0)$ for the constraint (21), and $w(\ge 0)$ for the constraint (22). If $(i,j,k) \notin \tilde{AK}$, we assume that $w_{ij}^k = 0$. By solving RLR($C^l, \tilde{P}, \tilde{AK}$) optimally, we get the optimal dual solution (s,u,w). The reduced cost of path flow variable x_p^k is represented by:

$$\sum_{(i,j)\in A} (c_{ij}^k + u_{ij} + w_{ij}^k)\delta_{ij}^p - s^k. \tag{25}$$

A pricing problem is used for generating new path flow variables. The pricing problem of RLR($C^l, \tilde{P}, \tilde{AK}$) is disjointed for each commodity k, from which point it can be solved separately. The separated pricing problem PP^k for commodity k is expressed as follows:

(PP^k)

$$z^k = \min \sum_{p\in P^k}\sum_{(i,j)\in A} (c_{ij}^k + u_{ij} + w_{ij}^k)\delta_{ij}^p x_p^k \tag{26}$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k, \tag{27}$$

$$x_p^k \ge 0 \quad \forall p \in P^k. \tag{28}$$

Since PP^k is the shortest path problem associated with nonnegative arc length $c_{ij}^k + u_{ij} + w_{ij}^k$, it can be solved efficiently by Dijkstra's algorithm. Let $\hat{p}$ be the optimal path of PP^k and z^k be the length of $\hat{p}$ for commodity k. If $z^k < s^k$, then the path flow variable $x_{\hat{p}}^k$ corresponding to the optimal path $\hat{p}$ exhibits negative reduced cost. Then path $\hat{p}$ is added to $\tilde{P}^k$, and the new variable $x_{\hat{p}}^k$ is generated as a new column.

After RLR($C^l, \tilde{P}, \tilde{AK}$) is solved optimally, violated forcing constraints by the solution of RLR($C^l, \tilde{P}, \tilde{AK}$) are generated. Let $\hat{x}$ be the optimal path flow and $\hat{y}$ the optimal design solution of RLR($C^l, \tilde{P}, \tilde{AK}$). If $\sum_{p\in\tilde{P}^k} \delta_{ij}^p \hat{x}_p^k > d^k \hat{y}_{ij}$, this forcing constraint is violated and then (i,j,k) is added to $\tilde{AK}$. When a cutting plane is applied for RLR($C^l, \tilde{P}, \tilde{AK}$), new violated forcing constraints may be generated. In such cases, we repeat to solve RLR($C^l, \tilde{P}, \tilde{AK}$) by column generation, until no

Algorithm 2: Column Generation and Cutting Plane

Set $\tilde{P}$ and $\tilde{AK} := \phi$;
repeat
  repeat
    Solve RLR($C^l, \tilde{P}, \tilde{AK}$) by an MIP solver;
    Get the optimal dual solution (s, u, w) of RLR($C^l, \tilde{P}, \tilde{AK}$);
    for $k \in K$ **do**
      Solve PP^k;
      Get the shortest path $\hat{p}$ and the shortest path length z^k;
      if $z^k < s^k$ **then** $\hat{p}$ is added to $\tilde{P}^k$ and generate path variable $x^k_{\hat{p}}$;
    end
  until *no path variable is generated*
  Get the optimal path flow $\hat{x}$ and the optimal design solution $\hat{y}$ of RLR($C^l, \tilde{P}, \tilde{AK}$);
  for $k \in K$ **do**
    for $(i, j) \in A$ **do**
      if $\sum_{p \in \tilde{P}^k} \delta^p_{ij} \hat{x}^k_p > d^k \hat{y}_{ij}$ **then** (i, j, k) is added to $\tilde{AK}$;
    end
  end
until *no violated forcing constraint is generated*

violated forcing constraint is generated. When no new path, no path flow variable and no forcing constraint are generated, LR(C^l) is solved optimally, and the optimal solution of LR(C^l) is the last solution of RLR($C^l, \tilde{P}, \tilde{AK}$). The algorithm of column generation and cutting plane is summarized in Algorithm 2.

In algorithm 1, CMNDP must be solved when the number of free design variables is less than or equal to B. Since solving this problem optimally is difficult, even if it has restricted design variables, we solve CMNDP associated with $\tilde{P}$ instead of P as the path set, and $\tilde{AK}$ instead of A and K as the forcing constraint set. Furthermore, if some upper bound UB of CMNDP has been identified, we add $Z_{CMNDP} < UB$ to CMNDP as a constraint, where Z_{CMNDP} is the objective function value of CMNDP. Let this restricted problem be CMNDP($\tilde{P}, \tilde{AK}$). CMNDP($\tilde{P}, \tilde{AK}$) can be solved comparatively easily.

5. Local Branching

Although capacity scaling may produce good solutions within a reasonable computation time in general, the process does not always yield high-quality solutions. Consequently, local branching (Fischetti and Lodi 2003) is applied to solutions derived from capacity scaling. Local branching is a method used to solve a new restricted problem based on an exploration of solution neighborhoods defined by local branching constraints.

Algorithm 3: Local Branching

Set $\bar{y}$ and UB by Capacity Scaling;

Add the equation (29) and (30) to CMNDA;

repeat
  Solve CMNDA within T;

  if *a feasible solution of CMNDA is found* **then**
    Obtain the upper bound Z_{CMNDA} and the feasible design solution $\bar{y}'$ of CMNDA;

    Remove the equation (29) and add the equation (31) to CMNDA;

    if $Z_{CMNDA} < UB$ **then**
      $UB := Z_{CMNDA}$ and $\bar{y} := \bar{y}'$;

      Add the equation (29) and (30) to CMNDA;
  else $M := M/\Delta M$;
until $M < M_{min}$

Given the feasible design solution $\bar{y}$ derived from capacity scaling and a neighborhood size parameter $M(>0)$, additional local branching constraints are as follows:

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1} (1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} \leq M, \tag{29}$$

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1} (1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} > 1. \tag{30}$$

The first branching constraint includes the domain of $M - OPT$ neighborhood of $\bar{y}$, while the second branching constraint excludes the current solution.

Local branching is applied for CMNDA. The feasible design solution $\bar{y}$ of CMNDP is also clearly that of CMNDA. The reason for using CMNDA instead of CMNDP($\tilde{P}, \tilde{AK}$) is that there is a possibility that the optimal path set of CMNDP is not included in paths of the restricted path set of CMNDP($\tilde{P}, \tilde{AK}$).

The branching constraint (29) and (30) for $\bar{y}$ is added to CMNDA, and we solve the added problem by an MIP solver. Since identifying an optimal solution for CMNDA is difficult, we set a computation time limit T to solve CMNDA. If a better feasible solution $\bar{y}'$ is found within T, then it becomes the new incumbent as $\bar{y} := \bar{y}'$. At this incumbent, we solve the problem such that the constraint (29) is replaced by the following constraint:

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1} (1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} > M+1. \tag{31}$$

If a better solution cannot be found, the neighborhood size parameter M is reduced to $M/\Delta M$ ($\Delta M > 1$) until $M < M_{min}$.

6. Computational Experiments

Experiments have been performed to evaluate the performance of the combined capacity scaling and local branching approach proposed in this paper. The results of the combined approach are compared to optimal values or lower bounds by an MIP solver, also compared to the results of local branching (Rodríguez-Martín and Salazar-Gonzáleza 2010), IP search (Hewitt et al. 2010), MIP tabu search (Chouman and Crainic 2010), cooperative parallel tabu search (Crainic and Gendron 2002) and cycle-based tabu search (Ghamlouche et al. 2003). To ensure comparisons, the same two sets of problem instances as used in Gendron and Crainic (1996) are employed. The detailed description of these problem instances is given in Crainic et al. (2001).

The first set of problem instances, denoted as "C," consists of 43 instances characterized by the number of nodes, arcs, and commodities. In the experiments, 37 instances in C-category problems are used, because the first six instances are trivial problems. Two letters are used to characterize the fixed cost level "F" for high, and "V" for low relative to flow cost, and the capacity level "T" for tight and "L" for loose relative to total demand.

The second set of problem instances, denoted as "R," consists of 153 instances divided into 18 groups ranging from r1 to r18. In the experiments, 81 instances, of which the number of nodes is 20, are used, because the instances in the first 9 groups from r1 to r9 are trivial problems. They are characterized by three fixed cost levels, "F01," "F05," and "F10," and three capacity levels, "C1," "C2," and "C8." If the "F" value is small, the fixed cost is low, and if the "F" value is large, the fixed cost is high relative to flow cost. If the "C" value is small, the arc capacity is loose; if the "C" value is large, the arc capacity is tight relatively to total demand.

The experiments were performed on PCs with Pentium i7 CPU 3.4GHz with four cores and 16GBytes RAM. The computer code was written in C++, compiled on Ubuntu 11 and CPLEX 12. An MIP solver by ILOG was used for solving linear programming problems and mixed integer programming problems. In order to assess solution quality relative to optimal values or lower bounds, all instances were solved by the branch-and-bound algorithm of CPLEX, and a limit of 30 hours of computation time was imposed for each instance. If the problem could not be solved optimally within the computation time limit, the lower bound found by the branch-and-bound algorithm was used instead of the optimal value. The smoothing parameter λ was calibrated from 0.025 to 0.250, and the neighborhood size parameter M and the time limit T were tested combinations 10-1000 seconds and 20-2000 seconds. We set the parameters as ϵ=0.01, $B = 150$, $\Delta B = B_{min} = 10$, $\Delta M = 2$, $M_{min} = 1$, $ITE_{min} = 50$, and $ITE_{max} = 1000$.

Table 1 displays the average gaps of the results for C-category problems. The average gaps are relative to the optimal value or the lower bound by CPLEX for the upper bound by each heuristic. Column LBR is the results by local branching, IPS by IP search, and MIP by MIP tabu search

Table 1 Average Gap for C-Category Problems(%)

LBR	IPS	MIP	CAP	10-1000	20-2000
3.28	1.86	1.04	0.95	0.51	0.43

respectively. Column CAP is the result by the capacity scaling heuristic excluding local branching. Columns 10-1000 and 20-2000 are the results of the combined approaches, and the first number indicates M and the second number indicates T. The average gap of MIP tabu search, which is the best result among four previous heuristics, is 1.04%. The average gap of the capacity scaling heuristic is 0.95%, and it is less than that of MIP tabu search. The average gaps of the combined approaches are 0.51% and 0.43%, and these are only about half of that of MIP tabu search.

Table 2 displays the detailed results for C-category problems. Column N/A/K/FC indicates the number of nodes, arcs, commodities, the fixed cost level, and the capacity level. Column OPT/LB corresponds to the optimal value or the lower bound by CPLEX. "O" indicates that the optimal value is found, while "L" indicates that the numerical value is the lower bound. A numerical value in bold type is the optimal value, while a numerical value in italic type is the best upper bound except the optimal value. Local branching is identified optimal solutions for nine instances out of 37 instances, conduct IP search for six instances, and utilize MIP tabu search for seven instances respectively. Meanwhile, the capacity scaling heuristic find the optimal solutions for six instances and both of the two combined approaches find for 21 instances. Additionally, the combined approach 20-2000 is used to identify the new best upper bounds for all of the 16 instances to the exclusion of the instances of which the optimal values are found. This means that the combined approach can help to identify the best solutions for the all instances of C-category problems.

Table 3 displays the average computation times in CPU seconds for four previous heuristics, as well as capacity scaling heuristic and combined approaches. These computation times for four previous heuristics are reported in the papers written about them. Due to the fact that different CPUs are used, these computation times cannot be compared directly. However, compared to previous heuristics, the average time required for computation by the capacity scaling heuristic is relatively short; e.g. 456.8 seconds, which is almost the same as the time level by local branching or IP search. The combined approaches can be solved within a reasonable average time for computation, such as 3180.7 or 7011.4 seconds, and the latter is the same as the time level by MIP tabu search. Table 4 shows the detailed computation times for instances of C-category problems.

Table 5 displays the average gaps of the results for R-category problems. Column CPT is the results by cooperative parallel tabu search and CYC by cycle-based tabu search. Although R-category problems are solved by path relinking and MIP tabu search, these average gaps cannot be listed here, because no detailed results are reported in the papers on these topics, and the gaps between them are a comparison between their upper bounds and upper bounds by CPLEX,

Table 2 Results for C-Category Problems

N/A/K/FC	OPT/LB	LBR	IPS	MIP	CAP	10-1000	20-2000
100/400/010/VL	28423.0^O	**28423**	**28423**	**28423**	28426.0	**28423.0**	**28423.0**
100/400/010/FL	23949.0^O	24690	**23949**	24161	24459.0	**23949.0**	**23949.0**
100/400/010/FT	63066.2^L	67357	65885	67233	68410.0	64207.0	*63753.0*
100/400/030/VT	384802.0^O	384809	384836	384940	384809.0	**384802.0**	**384802.0**
100/400/030/FL	49018.0^O	49872	49694	49682	49588.0	**49018.0**	**49018.0**
100/400/030/FT	132128.9^L	141633	141365	144349	142191.0	138152.0	*136250.0*
20/230/040/VL	423848.0^O	**423848**	424385	**423848**	**423848.0**	**423848.0**	**423848.0**
20/230/040/VT	371475.0^O	**371475**	371779	**371475**	371906.0	**371475.0**	**371475.0**
20/230/040/FT	643036.0^O	**643036**	643187	643538	643649.0	**643036.0**	**643036.0**
20/230/200/VL	94213.0^O	95295	95097	94218	94218.0	**94213.0**	**94213.0**
20/230/200/FL	137642.3^O	143446	141253	138491	137702.0	**137642.3**	**137642.3**
20/230/200/VT	97914.0^O	98039	99410	98612	97968.0	**97914.0**	**97914.0**
20/230/200/FT	135380.1^L	141128	140273	136309	136265.0	136031.0	*135866.5*
20/300/040/VL	429398.0^O	**429398**	**429398**	**429398**	**429398.0**	**429398.0**	**429398.0**
20/300/040/FL	586077.0^O	**586077**	**586077**	588464	587512.0	**586077.0**	**586077.0**
20/300/040/VT	464509.0^O	**464509**	**464509**	**464509**	**464509.0**	**464509.0**	**464509.0**
20/300/040/FT	604198.0^O	**604198**	**604198**	**604198**	**604198.0**	**604198.0**	**604198.0**
20/300/200/VL	74753.2^L	76375	75319	75045	74840.0	*74811.0*	*74811.0*
20/300/200/FL	113862.3^L	119142.8	117543	116259	115801.0	115748.0	*115539.0*
20/300/200/VT	74991.0^O	76167.5	76198	74995	74995.0	**74991.0**	**74991.0**
20/300/200/FT	106671.6^L	109808	110344	109164	107315.0	107315.0	*107102.0*
30/520/100/VL	53958.0^O	54026	54113	54008	53976.0	**53958.0**	**53958.0**
30/520/100/FL	93569.5^L	96255	94388	93967	94201.0	*93967.0*	*93967.0*
30/520/100/VT	52046.0^O	52129	52174	52156	52248.0	**52046.0**	**52046.0**
30/520/100/FT	96260.3^L	101102	98883	97490	97833.0	97385.0	*97107.0*
30/520/400/VL	112646.8^L	114367.4	114042	112927	112786.7	*112774.4*	*112774.4*
30/520/400/FL	147790.1^L	157725.5	154218	149920	149486.0	149423.0	*149335.4*
30/520/400/VT	114640.5^O	115240	114922	114664	**114640.5**	**114640.5**	**114640.5**
30/520/400/FT	150685.0^L	168561	154606	152929	152629.8	152575.8	*152510.6*
30/700/100/VL	47603.0^O	**47603**	47612	**47603**	**47603.0**	**47603.0**	**47603.0**
30/700/100/FL	59612.3^L	60272	60700	60184	60067.0	*59958.0*	*59958.0*
30/700/100/VT	45871.5^O	45905	46046	45880	46070.0	**45871.5**	**45871.5**
30/700/100/FT	54904.0^O	55104	55609	54926	55164.0	**54904.0**	**54904.0**
30/700/400/VL	97188.5^L	103787	98718	97982	97900.7	*97875.0*	*97875.0*
30/700/400/FL	131689.6^L	169759.7	152576	135109	134723.3	134619.5	*134589.8*
30/700/400/VT	94508.1^L	96680	96168	95781	95267.4	*95249.6*	*95249.6*
30/700/400/FT	128242.8^L	144925.5	131629	130856	*129909.6*	*129909.6*	*129909.6*

Table 3 Average Computation Time for C-Category Problems(Seconds)

LBR	IPS	MIP	CAP	10-1000	20-2000
574.6	408.1	6958.6	456.8	3180.7	7011.4

instead of lower bounds. The average gap of cooperative parallel tabu search is 9.46%, while the average gap of cycle-based tabu search is 5.74%. In contrast, the average gap of the capacity scaling heuristic excluding the local branch is 0.59% and those of the combined approaches stand at just

Table 4 Computation Time for C-Category Problems(Seconds)

N/A/K/FC	LBR	IPS	MIP	CAP	10-1000	20-2000
100/400/010/VL	547	35	49	0.4	1.9	2.3
100/400/010/FL	600	9	109	23.2	106.3	109.8
100/400/010/FT	600	813	918	11.8	2735.5	12978.9
100/400/030/VT	600	330	85	1.5	1503.0	2106.0
100/400/030/FL	600	886	2068	167.0	863.5	2643.2
100/400/030/FT	600	888	145	11.8	11027.8	24434.2
20/230/040/VL	328	4	116	0.4	1.2	1.1
20/230/040/VT	440	41	37	0.6	3.1	3.8
20/230/040/FT	600	45	63	0.6	14.6	10.8
20/230/200/VL	600	822	8328	104.2	3060.3	6538.8
20/230/200/FL	600	691	15006	253.7	4408.9	8899.1
20/230/200/VT	600	821	3965	67.7	2160.8	2894.8
20/230/200/FT	600	156	835	263.1	4537.9	11394.2
20/300/040/VL	146	19	83	0.4	1.2	0.9
20/300/040/FL	600	29	46	1.2	8.3	10.3
20/300/040/VT	600	24	161	0.6	3.9	3.7
20/300/040/FT	600	68	35	0.5	4.4	3.8
20/300/200/VL	600	802	4476	100.7	5047.8	9157.1
20/300/200/FL	600	686	9109	349.8	7768.8	8582.9
20/300/200/VT	600	388	1913	65.2	2157.9	2560.2
20/300/200/FT	600	396	542	257.6	3798.2	8239.3
30/520/100/VL	600	218	2415	7.1	672.5	1669.1
30/520/100/FL	600	226	2925	150.0	3266.4	6582.8
30/520/100/VT	600	455	2521	6.6	2004.0	4064.4
30/520/100/FT	600	815	4161	4992.4	4801.5	18076.9
30/520/400/VL	600	394	22797	95.4	5916.8	10474.8
30/520/400/FL	600	750	5769	1183.7	3386.7	18046.1
30/520/400/VT	600	621	38793	53.6	3725.8	8930.4
30/520/400/FT	600	466	8556	357.7	5148.7	15049.4
30/700/100/VL	600	32	3938	4.1	64.1	77.1
30/700/100/FL	600	741	5650	228.4	2888.3	6960.5
30/700/100/VT	600	371	4263	9.0	5559.1	8103.7
30/700/100/FT	600	387	3018	25.9	4455.9	7911.4
30/700/400/VL	600	222	35241	466.0	4726.9	10045.9
30/700/400/FL	600	860	21429	2115.0	7311.7	15460.7
30/700/400/VT	600	365	15372	1569.6	6376.3	11604.0
30/700/400/FT	600	225	32531	3955.4	8164.6	15789.0

Table 5 Average Gap for R-Category Problems(%)

CPT	CYC	CAP	10-1000	20-2000
9.46	5.74	0.59	0.19	0.17

0.19% and 0.17%. The gaps of the proposed approaches are very small relative to those reported in the two previous heuristics.

Table 6 shows the average gaps for the results of proposed approaches and two previous heuristics, according to the fixed cost level and the capacity level of R-category problems. The gaps by cycle-

Table 6 **Gap Distribution According to Fixed Cost and Capacity Level, R-Category Problems(%)**

Fixed Cost	Capacity	CPT	CYC	CAP	10-1000	20-2000
F01	C1	3.25	2.79	0.03	0.00	0.00
	C2	2.34	2.53	0.04	0.00	0.00
	C8	2.86	2.98	0.39	0.06	0.06
F05	C1	12.52	6.25	0.34	0.00	0.00
	C2	10.87	6.72	0.58	0.30	0.29
	C8	5.87	6.77	0.85	0.16	0.13
F10	C1	23.59	7.46	1.00	0.34	0.31
	C2	16.26	8.49	0.95	0.45	0.43
	C8	7.59	7.66	1.13	0.38	0.31

Table 7 **Gap Distribution According to Dimensions, R-Category Problems(%)**

N/A/K	CPT	CYC	CAP	10-1000	20-2000
20/100/40	3.36	2.78	0.75	0.00	0.00
20/100/100	4.97	2.73	0.13	0.00	0.00
20/100/200	4.26	5.26	0.19	0.01	0.00
20/200/40	5.92	3.57	1.07	0.00	0.00
20/200/100	10.48	7.10	0.28	0.00	0.00
20/200/200	16.54	8.28	0.64	0.51	0.47
20/300/40	6.63	3.44	0.71	0.15	0.10
20/300/100	15.63	6.69	0.58	0.29	0.29
20/300/200	17.38	11.79	0.96	0.74	0.69

based tabu search range from 2.53% to 8.49%. In contrast, the gaps indicated in the capacity scaling heuristic excluding the local branch range from 0.03% to 1.13%, those by the combined approach for 10-1000 range from 0.00% to 0.45%, and those for 20-2000 range from 0.00% to 0.43% respectively. Table 7 displays the same information, but in accordance with problem dimensions. The gaps by cycle-based tabu search range from 2.73% to 11.79%. In contrast, the gaps indicated by the combined approach for 10-1000 range from 0.00% to 0.74%, and those for 20-2000 range from 0.00% to 0.69%.

Table 8 indicates detailed results for instances of R-category problems from r10 to r14, while Table 9 shows the same for the range of r15 to r18. Numerical values shown in bold type are the optimal values, while those shown in italic are the best upper bound except the optimal value. The two combined approaches identify optimal solutions for the 43 instances out of 45 instances, ranging from r10 to r14, and for the 22 or 23 instances out of 36 instances, ranging from r15 to r18.

Table 10 displays the average computation times in CPU seconds for two previous heuristics, as well as the capacity scaling heuristic and the combined approaches. These computation times are reported in the papers written about them. Compared to previous heuristics, the computation times by capacity scaling are very short, such as 103.1 seconds. The combined approaches can be solved within a reasonable computation time, such as 1630.6 or 3374.9 seconds.

Table 8 Results for R-Category Problems

Group	C	F	OPT/LB	CPT	CYC	CAP	10-1000	20-2000
r10	C1	F01	200087.0^O	201350	200613	**200087.0**	**200087.0**	**200087.0**
		F05	346813.5^O	356255	350573	349099.0	**346813.5**	**346813.5**
		F10	488015.0^O	556586	507118	495407.0	**488015.0**	**488015.0**
	C2	F01	229196.0^O	231669	232473	229549.0	**229196.0**	**229196.0**
		F05	411664.0^O	428702	432913	416899.0	**411664.0**	**411664.0**
		F10	609104.0^O	632706	640621	620823.0	**609104.0**	**609104.0**
	C8	F01	486895.0^O	492159	488737	487014.0	**486895.0**	**486895.0**
		F05	951056.0^O	966670	980010	958575.0	**951056.0**	**951056.0**
		F10	1421740.0^O	1435900	1487270	1427375.0	*1421746.0*	*1421746.0*
r11	C1	F01	714431.0^O	725324	725416	**714431.0**	**714431.0**	**714431.0**
		F05	1263713.0^O	1362380	1306090	1265185.0	**1263713.0**	**1263713.0**
		F10	1843611.0^O	2181770	1914040	1854683.0	**1843611.0**	**1843611.0**
	C2	F01	870451.0^O	875021	876894	870906.0	**870451.0**	**870451.0**
		F05	1623640.0^O	1764460	1694860	1625443.0	**1623640.0**	**1623640.0**
		F10	2414060.0^O	2589570	2607690	2419709.0	**2414060.0**	**2414060.0**
	C8	F01	2294912.0^O	2294980	2295790	2295439.0	**2294912.0**	**2294912.0**
		F05	3507100.0^O	3510450	3568430	3508336.0	**3507100.0**	**3507100.0**
		F10	4579353.0^O	4601830	4621900	**4579353.0**	**4579353.0**	**4579353.0**
r12	C1	F01	1639443.0^O	1693860	1713670	1639565.0	**1639443.0**	**1639443.0**
		F05	3396050.0^O	3837460	3746250	3408018.0	**3396050.0**	**3396050.0**
		F10	5228711.0^O	6035740	6070200	5297992.0	**5228711.0**	**5228711.0**
	C2	F01	2303557.0^O	2338020	2326230	**2303557.0**	**2303557.0**	**2303557.0**
		F05	4669799.0^O	4756140	4967940	**4669799.0**	**4669799.0**	**4669799.0**
		F10	7100019.0^O	7311290	7638050	**7100019.0**	**7100019.0**	**7100019.0**
	C8	F01	7635270.0^O	7636930	7637250	**7635270.0**	**7635270.0**	**7635270.0**
		F05	10067742.0^O	10079200	10121700	**10067742.0**	**10067742.0**	**10067742.0**
		F10	11967768.0^O	11980100	12079300	**11967768.0**	**11967768.0**	**11967768.0**
r13	C1	F01	142947.0^O	143388	144138	143036.0	**142947.0**	**142947.0**
		F05	263800.0^O	276760	270316	265049.0	**263800.0**	**263800.0**
		F10	365836.0^O	416917	374999	369084.0	**365836.0**	**365836.0**
	C2	F01	150977.0^O	151998	151513	**150977.0**	**150977.0**	**150977.0**
		F05	282682.0^O	302360	291510	284213.0	**282682.0**	**282682.0**
		F10	406790.0^O	439292	420028	411899.0	**406790.0**	**406790.0**
	C8	F01	208088.0^O	213287	212451	209306.5	**208088.0**	**208088.0**
		F05	444826.0^O	480773	484112	455397.0	**444826.0**	**444826.0**
		F10	697967.0^O	753007	758715	721716.0	**697967.0**	**697967.0**
r14	C1	F01	403414.0^O	411327	415119	403703.0	**403414.0**	**403414.0**
		F05	749503.0^O	873856	803356	752488.0	**749503.0**	**749503.0**
		F10	1063098.0^O	1292350	1155840	**1064986.0**	**1063098.0**	**1063098.0**
	C2	F01	437607.0^O	443289	453204	**437607.0**	**437607.0**	**437607.0**
		F05	849163.0^O	911975	912456	850303.0	**849163.0**	**849163.0**
		F10	1214609.0^O	1467430	1333440	1215904.0	**1214609.0**	**1214609.0**
	C8	F01	668216.3^O	702321	702226	**668216.3**	**668216.3**	**668216.3**
		F05	1613428.8^O	1760950	1748930	1616844.0	*1613813.0*	*1613813.0*
		F10	2602690.0^O	2873660	2882710	2640197.0	**2602690.0**	**2602690.0**

According to our results, capacity scaling heuristic excluding local branching can offer good quality results, and time required for computation is relatively short. The combined capacity scaling

Table 9 Results for R-Category Problems

Group	C	F	OPT/LB	CPT	CYC	CAP	10-1000	20-2000
r15	C1	F01	1000787.0^O	1041800	1049360	**1000787.0**	**1000787.0**	**1000787.0**
		F05	1966206.0^O	2344340	2158720	1972352.0	1966584.0	**1966206.0**
		F10	2826200.7^L	4249580	3135760	2887346.0	2881867.0	*2876628.0*
	C2	F01	1148604.0^O	1196910	1215130	1148933.0	**1148604.0**	**1148604.0**
		F05	2452534.5^L	3134060	2756680	*2476246.0*	*2476246.0*	*2476246.0*
		F10	3771050.7^L	5287330	4384640	3845665.0	3831483.7	*3826956.0*
	C8	F01	2297919.0^O	2323700	2355730	2301016.8	**2297919.0**	**2297919.0**
		F05	5573412.8^O	5633870	5926330	5581719.3	**5573412.8**	**5573412.8**
		F10	8696932.0^O	8760190	9180920	**8696932.0**	**8696932.0**	**8696932.0**
r16	C1	F01	136161.0^O	138606	136538	**136161.0**	**136161.0**	**136161.0**
		F05	239500.0^O	256813	247682	**239500.0**	**239500.0**	**239500.0**
		F10	325671.0^O	369862	338807	325839.0	**325671.0**	**325671.0**
	C2	F01	138532.0^O	138946	139973	**138532.0**	**138532.0**	**138532.0**
		F05	241801.0^O	257604	246014	**241801.0**	**241801.0**	**241801.0**
		F10	337762.0^O	370432	355610	338663.0	**337762.0**	**337762.0**
	C8	F01	169233.0^O	173164	172268	172135.0	**169233.0**	**169233.0**
		F05	348167.0^O	370873	365214	355756.0	**348167.0**	**348167.0**
		F10	525454.3^L	586912	569874	536929.0	532580.0	*529988.0*
r17	C1	F01	354138.0^O	368966	370090	**354138.0**	**354138.0**	**354138.0**
		F05	645488.0^O	731107	688554	648354.0	**645488.0**	**645488.0**
		F10	910518.0^O	1143550	971151	915958.0	**910518.0**	**910518.0**
	C2	F01	370590.0^O	383225	380850	370622.0	**370590.0**	**370590.0**
		F05	706746.5^O	831600	753188	709562.0	**706746.5**	**706746.5**
		F10	1019646.0^O	1273240	1108180	1023343.0	*1020007.0*	*1020007.0*
	C8	F01	501634.5^O	522318	524038	502802.0	**501634.5**	**501634.5**
		F05	1097312.6^L	1292310	1195140	1108348.0	1106828.5	*1105083.0*
		F10	1752512.3^L	2274520	1945080	1791149.5	1785356.5	*1784793.0*
r18	C1	F01	828117.0^O	922368	872888	829097.0	**828117.0**	**828117.0**
		F05	1533675.0^O	1962020	1716680	1537887.0	**1533675.0**	**1533675.0**
		F10	2152710.3^L	3001340	2377560	2188346.0	*2174031.0*	2174276.0
	C2	F01	919325.0^O	993825	975396	920693.0	**919325.0**	**919325.0**
		F05	1799590.4^L	2102750	2037950	1831762.6	1830356.0	*1829968.0*
		F10	2640190.2^L	3396310	2966370	2703852.0	2703852.0	*2701286.0*
	C8	F01	1469035.8^L	1608720	1622520	1481064.8	1477522.2	*1477395.0*
		F05	3870930.2^L	4193520	4576750	3904541.0	3896893.2	*3888572.3*
		F10	6361906.0^O	6759150	7504310	6395749.7	6375803.0	*6369890.3*

Table 10 Average Computation Time for R-Category Problems(Seconds)

CPT	CYC	CAP	10-1000	20-2000
1654.7	2650.2	103.1	1630.6	3374.9

and local branching approach can obtain good solutions within a reasonable computation time, and can also help to improve current best solutions or to identify optimal solutions for all C-category problems. This method also yielded optimal solutions for 66 out of the 81 instances of R-category problems.

7. Conclusion

This paper presents the combined capacity scaling and local branching approach, which is applied to column generation and cutting plane techniques for CMND characterized by strong formulation. The performance of the proposed approaches is evaluated by solving C and R-category problems. The numerical results are satisfactory, while the combined approach helps to identify the best new solutions or the optimal solutions for all instances of C-category problems, as well as optimal solutions for 66 out of 81 instances of R-category problems.

The capacity scaling approach excluding local branching can offer good quality results while also helping to reduce computation efforts. The combined capacity scaling and local branching approach can offer the highest quality results, and outperforms previous heuristics proposed in the literature. We believe that the combined approach proposed in this paper is one of the best current heuristics for solving CMND.

Acknowledgments

This work was supported by Grant-in-Aid for Scientific Research of Japan, No.21510155.

References

Alvarez, A. M., J. L. González-Velarde, K. De-Alba. 2005. Scatter search for network design problem. *Ann. Oper. Res.* **138**(1) 159–178.

Atamtürk, A., D. Rajan. 2002. On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Math. Programming* **92**(2) 315–333.

Balakrishnan, A., T. L. Magnanti, P. Mirchandani. 1997. Network design. M. Dell'Amico, F. Maffioli, S. Martello, eds., *Annotated Bibliographies in Combinatorial Optimization*. John Wiley & Sons, New York, 311–334.

Barahona, F. 1996. Network design using cut inequalities. *SIAM J. Comput.* **6** 823–837.

Bienstock, D., O. Günlük. 1996. Capacitated network design – polyhedral structure and computation. *ORSA J. Comput.* **8** 243–259.

Chouman, M., T. G. Crainic. 2010. A MIP-Tabu search hybrid framework for multicommodity capacitated fixed-charge network design. Technical report CIRRELT-2010-31, Centre de recherche sur les transports,Université de Montréal, Canada.

Chouman, M., T. G. Crainic, B. Gendron. 2003. A cutting-plane algorithm based on cutset inequalities for multicommodity capacitated fixed charge network design. Technical report CIRRELT-2003-16, Centre de recherche sur les transports,Université de Montréal, Canada.

Chouman, M., T. G. Crainic, B. Gendron. 2009. A cutting-plane algorithm for multicommodity capacitated fixed-charge network design. Technical report CIRRELT-2009-20, Centre de recherche sur les transports,Université de Montréal, Canada.

Costa, A., J. F. Cordeau, B. Gendron. 2009. Benders, metric and cutset inequalities for multicommodity capacitated network design. *Comput. Optimization and Applications* **42** 371–392.

Costa, A. M. 2005. A survey on benders decomposition applied to fixed-charge network design problems. *Comput. Oper. Res.* **32** 1429 – 1450.

Crainic, T. G. 2003. Long-haul freight transportation. R. W. Hall, ed., *Handbook of Transportation Science*. Kluwer Academic Publishers, 451–516.

Crainic, T. G., A. Frangioni, B. Gendron. 2001. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design problems. *Discrete Applied Math.* **112** 73–99.

Crainic, T. G., M. Gendreau. 2007. A scatter search heuristic for the fixed-charge capacitated network design problem. K. F. Doerner, M. Gendreau, P. Greistorfer, W. Gutjahr, R. F. Hartl, M. Reimann, eds., *Metaheuristics*. Springer, 25–40.

Crainic, T. G., M. Gendreau, J. M. Farvolden. 2000. A simplex-based tabu search for capacitated network design. *ORSA J. Comput.* **12** 223–236.

Crainic, T. G., B. Gendron. 2002. Cooperative parallel tabu search for capacitated network design. *J. Heuristics* **8** 601–627.

Crainic, T. G., B. Gendron, G. Hernu. 2004. A slope scaling/Lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. *J. Heuristics* **10** 525–545.

Crainic, T. G., Y. Li, M. Toulouse. 2006. A first multilevel cooperative algorithm for capacitated multicommodity network design. *Comput. Oper. Res.* **33** 2602–2622.

Fischetti, M. Matteo, A. Lodi. 2003. Local branching. *Math. Programming* **98**(1-3) 23–47.

Gendron, B., T. G. Crainic. 1994. Relaxations for multicommodity capacitated network design problems. Technical report CIRRELT-965, Centre de recherche sur les transports,Université de Montréal, Canada.

Gendron, B., T. G. Crainic. 1996. Bounding procedures for multicommodity capacitated fixed charge network design problems. Technical report CIRRELT-96-06, Centre de recherche sur les transports,Université de Montréal, Canada.

Gendron, B., T. G. Crainic, A. Frangioni. 1997. Multicommodity capacitated network design. Technical report CIRRELT-98-14, Centre de recherche sur les transports,Université de Montréal, Canada.

Ghamlouche, I., T. G. Cainic, M. Gendreau. 2011. Learning mechanisms and local search heuristics for the fixed charge capacitated multicommodity network design. *Internat. J. Comput. Sci. Issues* **8**(6) 21–32.

Ghamlouche, I., T. G. Crainic, M. Gendreau. 2003. Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Oper. Res.* **51** 655–667.

Ghamlouche, I., T. G. Crainic, M. Gendreau. 2004. Path relinking, cycle-based neighborhoods and capacitated multicommodity network design. *Ann. Oper. Res.* **131** 109–134.

Herrmann, J. W., G. Ioannou, I. Minis, J. M. Proth. 1996. A dual ascent approach to the fixed-charge capacitated network design problem. *Eur. J. Oper. Res.* **95** 476–490.

Hewitt, M., G. L. Nemhauser, M. Savelsbergh. 2010. Combining exact and heuristics approaches for the capacitated fixed charge network flow problem. *ORSA J. Comput.* **22** 314–325.

Holmberg, K., D. Yuan. 2000. A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Oper. Res.* **48** 461–481.

Katayama, N., M. Chen, M. Kubo. 2009. A capacity scaling heuristics for the multicommodity capacitated network design problem. *J. Comput. Applied Math.* **232** 90–101.

Katayama, N., H. Kasugai. 1993. A capacitated multi-commodity network design problem - a solution method for finding a lower bound using valid inequalities. *J. Japan Industrial Management Association* **44** 164–175.

Kliewer, G., L. Timajev. 2005. Relax-and-cut for capacitated network design. G. S. Brodal, S. Leonardi, eds., *Algorithms - ESA 2005: Lecture Notes in Computer Science*. Springer, 47–58.

Magnanti, T. L., P. Mirchandani, R. Vachani. 1993. The convex hull of two core capacitated network design problems. *Math. Programming* **60** 233–250.

Magnanti, T. L., P. Mireault, R. T. Wong. 1986. Tailoring benders decomposition for uncapacitated network design. *Math. Programming Study* **26** 112–155.

Magnanti, T. L., R. T. Wong. 1984. Network design and transportation planning : Models and algorithms. *Transportation Sci.* **18** 1–55.

Minoux, M. 1989. Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks* **19** 313–360.

Pochet, Y., M. V. Vyve. 2004. A general heuristic for production planning problems. *ORSA J. Comput.* **16** 316 – 327.

Powell, W. B., Y. Sheffi. 1989. Design and implementation of an interactive optimization system for network design in the motor carrier industry. *Oper. Res.* **37** 12–29.

Rodríguez-Martín, I., J. J. Salazar-González. 2010. A local branching heuristics for the capacitated fixed-charge network design problem. *Comput. Oper. Res.* **37** 575–581.

Wong, R. T. 1984. Introduction and recent advances in network design: Models and algorithms. M. Florian, ed., *Transportation Planning Models*. Elsevier Science, North Holland, Amsterdam, 187–225.

Wong, R. T. 1985. Location and network design. M. O'heEigeartaigh, J. Lenstra, A. RinnooyKan, eds., *Combinatorial Optimization Annotated Bibliographies*. John Wiley & Sons, New York, 129–147.

Yaghini, M., R. Akhavan. 2012. Multicommodity network design problem in rail freight transportation planning. *Procedia - Social and Behavioral Sci.* (43) 728–739.

Yaghini, M., M. Karimiand M. Rahbar. 2011. A hybrid simulated annealing and simplex method for fixed-cost capacitated multicommodity network design. *Internat. J. Applied Metaheuristic Comput.* **2**(4) 13–28.

Zaleta, N. C., A. M. A. Socarrás. 2004. Tabu search-based algorithm for capacitated multicommodity network design problems. *Proc. 14th International Conference on Electronics, Communications and Computers*, IEEE, Veracruz. 144 – 148.