A Combined Matheuristic for Service Network Design Problem

Naoto Katayama
Ryutsu Keizai University, Distribution and Logistics Systems,
email: katayama@rku.ac.jp

Abstract

Service network design problem is a model of a network design problem related to services in transportation and logistics planning. In this paper, we consider the service network design problem with a single-asset type. This problem is particularly relevant to firms that operate consolidation transportation systems and makes the determination of the transportation network configuration and the characteristics of the corresponding assets, and can be represented as a capacitated multicommodity network design problem with design balance constraints. This paper presents a combined matheuristic with capacity scaling, restricted branch-and-bound and local branching for the service network design problem. By combining capacity scaling and restricted branch-and-bound for a strong path-flow based formulation and local branching for a strong arc-flow based formulation, this combined matheuristic can offer one of the best current solutions compared to previous heuristics.

Keyword: Service Network Design, Capacity Scaling, Local Branching.

1. Introduction

Service network design problem (SNDB) is a model of a network design problem related to services in transportation and logistics planning. This problem is particularly relevant to firms that operate consolidation transportation systems and is typically related to the tactical/operational planning [9]. The service network design problem can be stated as selecting assets such that all commodities can be transported from their origin to their destination, and routing assets to ensure their availability between terminals to perform the selected assets.

In this paper, a fundamental service network design problem with a single-asset type is considered such that only one unit of asset is used for each service such as crews, trucks, or ships [15]. The service network design problem with a single-asset type is represented as the capacitated multicommodity network design problem with design balance constraints.

The capacitated multicommodity network design problem represents a generic network model for applications in designing the construction and improvement of logistics, transportation, distribution and production networks under arc capacities. The capacitated multicommodity network design problem is known as NP-Hard [5, 13]. As the problem removed design balance constraints from SNDB is reduced to the capacitated multicommodity network design problem, SNDB is also NP-Hard.

Pedersen et al. [15] presented a service network design models with a single-asset type and developed a multi-start matheuristic based on a tabu search heuristic. Andersen et al. [2] compared the node-arc based formulation, the path-based formulation and a cycle-based formulation for service network design problems. Bai et al. [3] proposed a guided local search, which is used to search based on a good network design vector solving the corresponding LP relaxation problem. Andersen et al. [1] proposed a branch and price method for a cycle-based formulation of the service network design problem. Chouman and Crainic [6] presented a MIP-based tabu search heuristic, and Chouman and Crainic [7] presented a matheuristic with cutting-plane of various valid inequalities by a MIP-based learning process. Bai et al. [4] developed a tabu assisted guided local search with a multistart and an efficient feasibility repairing heuristic. Recently, Vu et al. [16] developed a three-phase matheuristic combining tabu search, path relinking and intensification phases.

For the capacitated multicommodity network design problem, which is removed design balance constraints from SNDB, several surveys on the models and the associated solution methods can be found in Magnanti and Wong [13], Wong [17], Minoux [14], Balakrishnan et al. [5], Gendron et al. [11], Costa [8], and Yaghini and Rahbar [18]. Recently, Katayama and Yurimoto [12] proposed a combined capacity scaling and local branching. This matheuristic can offer highest quality results for the problem.

In many papers, an arc-flow based formulation or a path-flow based formulation without forcing constraints are used for SNDB. Since the formulation including forcing constraints is a large mixed integer programming problem, it takes significant amounts of computation time to solve the large size problem or its linear relaxation problem. However, for improving the solutions and the lower bound derived from the linear

Received:
Accepted:

relaxation problem, it is desirable to solve the formulation including forcing constraints. Column generation for path-flow variables and cutting plane for forcing constraints can reduce the problem size and increase solvability of the problem.

This paper presents a capacity scaling heuristic using column generation and cutting plane techniques for solving SNDB with forcing constraints. Capacity scaling is an approximate iterative solution approach based on changing arc capacities and it can be produced most convergence solutions. Capacity scaling produces good solutions within a reasonable computation time in general, but high-quality solutions may not be produced all the time. Restricted branch-and-bound is applied for the solutions derived from capacity scaling. Restricted branch-and-bound is the method to solve the problem which has restricted 0-1 variables. Consequently local branching is applied for the best solution of restricted branch-and-bound. Local branching is the method to solve a new problem with the addition of local branching constraints based on an exploration of solution neighborhoods. A combined capacity scaling, restricted branch-and-bound and local branching matheuristic can produce good solutions compared to previous approaches found in related literature for all benchmark problems.

2. Mathematical Formulation

A network $G=(N,A,K)$ is provided with a set of nodes N, a set of directed arcs A and a set of commodities K. d^k is defined as the demand of flow of commodity $k \in K$ from its single origin node to its single destination node. N_n^- is defined as the set of outward nodes from node n, and N_n^+ is defined as the set of inward nodes into node n. Let f_{ij} be the fixed cost of an asset on arc $(i,j) \in A$, c_{ij}^k the unit variable flow cost for commodity k flowing on arc (i,j), and C_{ij} the limited arc capacity for all commodities.

The formulation of SNDB has two type variables. The first type is a binary asset design variable. The asset design variable y_{ij} indicates whether an asset is used on arc (i,j), $y_{ij}=1$, or not, $y_{ij}=0$ in the network design. The second type is a continuous path flow variable. The path flow variable x_p^k represents the amount of path flow of commodity flowing on path $p \in P^k$. The constant δ_{ij}^p indicates whether arc (i,j) is included in path p, $\delta_{ij}^p=1$, or not, $\delta_{ij}^p=0$.

The path-flow based formulation SNDBP of SNDB can be expressed as follows:

(SNDBP)

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

subject to

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

$$\sum_{i\in N_n^+} y_{in} - \sum_{j\in N_n^-} y_{nj} = 0 \quad \forall n \in N, \quad (3)$$

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

$$\sum_{p\in P^k} \delta_{ij}^p x_p^k \le d^k y_{ij} \quad \forall (i,j) \in A, k \in K, \quad (5)$$

$$x_p^k \ge 0 \quad \forall p \in P^k, k \in K, \quad (6)$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A. \quad (7)$$

The objective function (1) minimizes the total cost, which is the sum of variable flow costs of commodities plus the sum of fixed costs for assets in a given network design. Equations (2) ensure flow conservations such that the sum of path flows of commodity k is equal to the demand. Equations (3) consist of the design balance constraints ensuring that the total number of assets on arcs entering node n is equal to that of assets on arcs leaving node n. Constraints (4) provide the capacity constraints for assets and state that the total flow on arcs cannot exceed the capacity, if the asset on arc (i,j) is used, $y_{ij}=1$, and must be 0 if the asset is not used, $y_{ij}=0$. Constraints (5) provide the forcing constraints for assets and commodities and state that the flow for commodity k cannot exceed the demand, if $y_{ij}=1$, and must be 0, if $y_{ij}=0$. Since forcing constraints are tight constraints for SNDB, a good lower bound can be obtained from its linear relaxation problem of the strong formulations with the forcing constraints. As the forcing constraints are additional valid constraints, the formulation removed the forcing constraints has the same optimal solution. Constraints (6) and (7) are non-negativity and integrality conditions for variables.

Let z_{ij}^k be the arc flow variable for commodity k flowing on arc (i,j). The arc-flow based formulation SNDBA of SNDB can be expressed as follows:

(SNDBA)

$$\min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}^k z_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \quad (8)$$

subject to

$$\sum_{i\in N_n^+} z_{in} - \sum_{j\in N_n^-} z_{nj} = \begin{cases} -d^k & \text{if } n = O^k \\ d^k & \text{if } n = D^k \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

$$\forall n \in N, k \in K,$$

$$\sum_{i\in N_n^+} y_{in} - \sum_{j\in N_n^-} y_{nj} = 0 \quad \forall n \in N, \quad (10)$$

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

$$(11)$$

$$z_{ij}^k \le d^k y_{ij} \quad \forall (i,j) \in A, k \in K, \quad (12)$$

$$z_{ij}^k \ge 0 \quad \forall (i,j) \in A, k \in K, \quad (13)$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A. \quad (14)$$

Equations (9) are flow conservations and 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 (11) are the capacity constraints for assets, and constraints (12) are the forcing constraints for assets and commodities.

Since the problem removed the design balance constraints from SNDB is reduced to the capacitated multicommodity network design problem, some heuristics for the capacitated multicommodity network design problem can be applied for solving SNDB without a major change. In this paper, capacity scaling and restricted branch-and-bound is applied for the path-flow based formulation SNDBP using column generation and cutting plane. Local branching is applied for the arc-flow based formulation SNDBA.

3. Capacity Scaling

Capacity scaling represented an approximate iterative solution approach for capacitated network problems based on changing arc capacities, which depend on arc flows. As the design variable value derived from the linear relaxation solution of SNDBP may be a decimal fraction, the solution may not be approximation for finding a good feasible binary solution of SNDBP. On the other hand, as almost the design variables of the solution derived from capacity scaling are binary, the solution may be a good initial solution for finding a good feasible binary solution of SNDB.

If the optimal path flow and the optimal arc flow of SNDBP are found, we consider the problem changed capacity to the optimal arc flow for each arc. When the linear relaxation problem with the capacities is solved, either 0 or 1 solutions for all design variables can be obtained. Since SNDBP associated with all fixed design variables is reduced to a multicommodity flow problem, the optimal objective value of SNDBP is obtained. As a matter of course, finding the optimal flow of SNDBP is our main purpose, and it is as hard as solving SNDB. Consequently, if a near-optimal flow can be found, the optimal arc flows can be estimated and a good approximate solution could possibly be derived from it. On the other hand, by changing capacities a little bit at a time, we may gradually identify the near-optimal flow.

Capacity scaling proceeds by solving the linear relaxation problem of SNDBP associated with C^l as substitution for C at the iteration l. The linear relaxation problem LR(C^l) with capacity C^l can be expressed as follows:

(LR(C^l))

$$\min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}^k \sum_{p\in P^k}\delta_{ij}^p x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \quad (15)$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k \quad \forall k \in K, \quad (16)$$

$$\sum_{k\in K}\sum_{p\in P^k}\delta_{ij}^p x_p^k \le C_{ij}^l y_{ij} \quad \forall (i,j)\in A, \quad (17)$$

$$\sum_{i\in N_n^+} y_{in} - \sum_{j\in N_n^-} y_{nj} = 0 \quad \forall n \in N, \quad (18)$$

$$\sum_{p\in P^k}\delta_{ij}^p x_p^k \le d^k y_{ij} \quad \forall (i,j)\in A, k \in K, \quad (19)$$

$$x_p^k \ge 0 \quad \forall p \in P^k, k \in K, \quad (20)$$

$$0 \le y_{ij} \le C_{ij} / C_{ij}^l \quad \forall (i,j)\in A. \quad (21)$$

C_{ij} is replaced for C_{ij}^l in the right-hand side of the constraint (17), and the upper bound 1 is replaced for C_{ij} / C_{ij}^l to enable arc flow up to its original capacity C_{ij} in the right-hand side of the constraint (21).

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 $\lambda\tilde{X}_{ij} + (1-\lambda)C_{ij}^l$, where $\lambda(0 \le \lambda \le 1)$ a smoothing parameter is designed to prevent rapid changes. If all design variables converge to zero or one in the solution of LR(C^l), then we solve the multicommodity network flow problem associated with all fixed design variables set to these convergent values, and both a feasible solution and an upper bound of SNDBP can be found. It may require numerous iterations for obtaining convergent solutions, otherwise few design variables may not converge. Consequently when most design variables converge to zero or one by a threshold value ε, and the number of non-converged design variable 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 solving SNDBP associated with non-convergent free variables, and the upper bound may be found. Let ΔB be subtracted from B. After the restricted 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 found, or the iteration number exceeds the maximum iteration number $ITE_{\max}$. An outline of capacity scaling and restricted branch-and-bound proceed as Algorithm 1.

Algorithm 1: Capacity Scaling and Restricted Branch-and-Bound

---

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;
  Get the arc flow solution $\tilde{X}$ and the design solution $\tilde{y}$ of LR(C^l);
  for $(i,j)\in A$ do
    if $\tilde{y}_{ij} < \varepsilon$ then $y_{ij} = 0$;
    else if $\tilde{y}_{ij} > 1-\varepsilon$ then $y_{ij} = 1$;
    else y_{ij} is free;

end

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

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

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

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

If $Z_{SNDBP} < UB$ then

$UB := Z_{SNDBP}$;

$l := l + 1$;

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

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

end

until $l > ITE_{\min}$ and $UB \neq \infty$, or $l > ITE_{\max}$

4. Column Generation and Cutting Plane

In capacity scaling, the linear programming problem LR(C^l) is solved iteratively. Since LR(C^l) has exponential number of path flow variables and the forcing constraints of the number of $O(|K||A|)$, all variables and constraint should not be included in the problem explicitly when solving large instances. In order to solve larger instances efficiently, column generation can be applied for path flow variables and cutting plane can be applied for forcing constraints.

At column generation procedure, we generate path flow variables, which have the negative reduced cost in the current problem. At cutting plane procedure, if there are forcing constraints which are violated by the current solution, these forcing constraints are generated and the new forcing constraints are added to the forcing constraint set.

For each commodity k, let $\tilde{P}^k \subset P^k$ be the initial set of paths. Let $\tilde{A}K \subset A \times K$, be the set of index of current forcing constraints.

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

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

$$\min \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^{k} \sum_{p\in \tilde{P}^k} \delta_{ij}^{p} x_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{22}$$

subject to

$$\sum_{p\in \tilde{P}^k} x_p^k = d^k \quad \forall k \in K, \tag{23}$$

$$\sum_{k\in K} \sum_{p\in \tilde{P}^k} \delta_{ij}^{p} x_p^k \le C_{ij}^{l} y_{ij} \quad \forall (i,j) \in A, \tag{24}$$

$$\sum_{i\in N_n^+} y_{in} - \sum_{j\in N_n^-} y_{nj} = 0 \quad \forall n \in N, \tag{25}$$

$$\sum_{p\in \tilde{P}^k} \delta_{ij}^{p} x_p^k \le d^k y_{ij} \quad \forall (i,j,k) \in \tilde{A}K, \tag{26}$$

$$x_p^k \ge 0 \quad \forall p \in \tilde{P}^k, k \in K, \tag{27}$$

$$0 \le y_{ij} \le C_{ij} / C_{ij}^{l} \quad \forall (i,j) \in A. \tag{28}$$

Let s be the dual variable for the constraint (23), $u(\ge 0)$ for the constraint (24), $w(\ge 0)$ for the constraint (26). If $(i,j,k) \notin \tilde{A}K$, we assume that $w_{ij}^k = 0$. By solving RLR($C^l, \tilde{P}, \tilde{A}K$) optimally, we get the optimal dual solution (s, u, w). The reduced cost of path flow variable x_p^k is represented by:

$$\sum_{(i,j)\in A} (c_{ij}^k + u_{ij} + w_{ij}^k) \delta_{ij}^{p} - s^k. \tag{29}$$

A pricing problem is used for generating new variables. The pricing problem of RLR($C^l, \tilde{P}, \tilde{A}K$) is reduced to shortest path problems for each commodity k. As s^k can be considered as a constant, the separated pricing problem PP^k for commodity k is expressed as follows:

(PP^k)

$$z^k = \min \sum_{p\in P^k} \sum_{(i,j)\in A} (c_{ij}^k + u_{ij} + w_{ij}^k) \delta_{ij}^{p} x_p^k \tag{30}$$

subject to

$$\sum_{p\in P^k} x_p^k = d^k, \tag{31}$$

$$x_p^k \ge 0 \quad \forall p \in P^k. \tag{32}$$

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 $\hat{p}$. If $z^k < s^k$, then the path flow variable $x_{\hat{p}}^k$ corresponding to $\hat{p}$ has 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.

By the solution of RLR($C^l, \tilde{P}, \tilde{A}K$), the violated forcing constraints are generated and the index of the forcing constraints is added to $\tilde{A}K$, and we repeat to solve RLR($C^l, \tilde{P}, \tilde{A}K$). When no new path and no path flow variable are generated, LR(C^l) is solved optimally and the optimal solution of LR(C^l) is the last solution of RLR($C^l, \tilde{P}, \tilde{A}K$). The algorithm of column generation and cutting plane is summarized in Algorithm 2.

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

comparatively easily.

Algorithm 2: Column Generation and Cutting Plane

Set $\tilde{P}$ and , $\tilde{A}K := \phi$;
repeat
  repeat
    Solve RLR($C^l, \tilde{P}, \tilde{A}K$) by an MIP solver;
    Get the optimal dual solution (s , u , w) of RLR($C^l, \tilde{P}, \tilde{A}K$);
    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 LR(C^l);
  for $(i, j) \in A$ do
    for $k \in K$ do
      if $\sum_{p \in P^k} \delta^p_{ij} \hat{x}^k_p \le d^k \hat{y}_{ij}$ then
        (i, j, k) is add to $\tilde{A}K$;
    end
  end
until *no violated forcing constraint is generated*

5. Local Branching

Although capacity scaling and restricted branch-and-bound may produce good solutions within a reasonable computation time in most cases, the method may not always yield high-quality solutions. Consequently, for improving the solutions, a kind of local branching [10] is applied to solutions derived from capacity scaling and restricted branch-and-bound. Local branching is the method to solve a new restricted problem based on an exploration of solution neighborhoods defined by local branching constraints.

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

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1}(1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} \le M, \quad (33)$$

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1}(1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} \ge 1. \quad (34)$$

The first branching constraint includes the domain of $M - OPT$ neighborhood of $\bar{y}$ and the second branching constraint excludes the current solution. Since the feasible design solution of SNDBP is also clearly that of SNDBA, the solution can be used for local branching of SNDBA. The reason for using SNDBA instead of SNDBP($\tilde{P}, \tilde{A}K$) is that there is a possibility that the optimal path set of SNDBA may not be included in paths of the restricted path set of SNDBP($\tilde{P}, \tilde{A}K$).

We add the branching constraint (33) and (34) for $\bar{y}$ to SNDBA, and solve the added problem by an MIP solver. Since solving SNDBA optimally is hard, we set a computation time limit T to solve SNDBA. If a better feasible solution $\bar{y}'$ than the current solution is found within T , it becomes the new incumbent as $\bar{y} := \bar{y}'$. Before this incumbent, we solve the problem such that the constraint (33) is replaced by the following constraint:

$$\sum_{(i,j)\in A|\bar{y}_{ij}=1}(1-y_{ij}) + \sum_{(i,j)\in A|\bar{y}_{ij}=0} y_{ij} \ge M+1. \quad (34)$$

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

Algorithm 3: Local Branching

Set $\bar{y}$ and UB by Capacity Scaling and Restricted Branch-and-Bound;
Add the equation (33) and (34) to SNDBA;
repeat
  Solve SNDBA within T ;
  if *a feasible solution of SNDBA is found* then
    Get the upper bound Z_{SNDBA} and the feasible design solution $\bar{y}'$ of SNDBA;
    Remove the equation (33) and add the equation (35) to SNDBA;
  if $Z_{SNDBA} < UB$ then
    $UB := Z_{SNDBA}$ and $\bar{y} := \bar{y}'$;
    Add the equation (33) and (34) to SNDBA;
  else $M := M / \Delta M$;
until $M < M_{\min}$

6. Computational Experiments

Experiments have been performed to evaluate the performance of the combined matheuristic proposed in this paper. The results of the combined matheuristic are compared to optimal values or lower bounds by an MIP solver, also compared to the results of parallel tabu search [15], tabu assisted guided local search [4], cutting-plane matheuristic [7] and three-phase matheuristic [16]. To ensure comparisons, the same two sets of problem instances as used for SNDB are employed. The detailed description of these problem instances is given in Pedersen et al. [15].

The first set of problem instances, denoted in category “C,” consists of 24 instances characterized by the

Table 1. Average Gap for C-category problems (%)

PTAB	TGLS	CPM	TFM	RBB	10-1000	10-2000	20-2000
4.55	2.55	2.30	1.18	0.92	0.69	0.66	0.65

Table 2. Results for C-category problems

N/A/K/FC	OPT/LB	PTAB	TGLS	CPM	TFM	RBB	10-1000	10-2000	20-2000
20/230/200/VL	97273.5^{O}	101345	98760.0	98699	**97274**	97597.0	**97273.5**	**97273.5**	**97273.5**
20/230/200/FL	139395.0^{O}	148384	142213.0	141744	**139395**	139831.0	**139395.0**	**139395.0**	**139395.0**
20/230/200/VT	100221.0^{O}	103371	102137.3	103103	100720	100530.0	**100221.0**	**100221.0**	**100221.0**
20/230/200/FT	138856.3^{L}	144766	141802.0	142638	138962	139253.0	139059.0	*138962.0*	*138994.0*
20/300/200/VL	77436.0^{O}	80269	78787.0	79953	77584	77584.0	77502.0	**77436.0**	77502.0
20/300/200/FL	118152.4^{L}	126258	121773.0	120979	119987	119324.3	*119259.0*	*119259.0*	*119259.0*
20/300/200/VT	76207.5^{O}	78444	77066.0	76545	76450	**76207.5**	**76207.5**	**76207.5**	**76207.5**
20/300/200/FT	110669.0^{L}	116338	114465.0	113412	111776	*111462.0*	*111462.0*	*111462.0*	*111462.0*
30/520/100/VL	54683.0^{O}	55786	55422.0	55733	54783	54733.0	**54683.0**	**54683.0**	**54683.0**
30/520/100/FL	96582.9^{L}	101612	100290.0	104235	100098	99193.0	98470.7	98519.0	*98170.8*
30/520/100/VT	53023.0^{O}	54092	53744.0	53224	53035	53246.0	**53023.0**	**53023.0**	**53023.0**
30/520/100/FT	99668.7^{L}	104702	103996.0	106251	101412	102043.0	101177.0	101229.0	*101131.0*
30/520/400/VL	114071.2^{L}	118071	116196.0	115220	115528	114688.2	*114565.1*	*114565.1*	*114565.1*
30/520/400/FL	150503.6^{L}	160979	154941.0	153737	153409	152893.4	*152776.3*	*152776.3*	*152776.3*
30/520/400/VT	116270.9^{L}	120421	118335.7	117056	117226	116670.9	116670.9	*116508.7*	116616.4
30/520/400/FT	153109.0^{L}	161987	157939.6	155942	155906	155121.0	155121.0	*154820.2*	*154820.2*
30/700/100/VL	48693.0^{O}	49429	49221.0	49268	48807	48708.0	**48693.0**	**48693.0**	**48693.0**
30/700/100/FL	60853.6^{L}	63292	62055.0	62267	61408	61528.0	61463.0	61331.0	*61298.0*
30/700/100/VT	46750.0^{O}	47487	47518.0	46928	46812	47137.0	**46750.0**	**46750.0**	**46750.0**
30/700/100/FT	56131.0^{O}	57187	57571.0	57701	56237	56609.0	56177.0	*56169.0*	*56169.0*
30/700/400/VL	98428.2^{L}	103932	101609.5	99458	100589	99332.5	99332.5	*99314.0*	*99314.0*
30/700/400/FL	134240.0^{L}	148114	142563.2	139607	141037	137889.0	137771.0	*137724.2*	*137724.2*
30/700/400/VT	96165.5^{L}	103085	98656.8	97737	97875	97425.5	97329.0	97329.0	*97278.0*
30/700/400/FT	130234.7^{L}	138609	135777.5	132855	133686	*132486.7*	*132486.7*	*132486.7*	*132486.7*

number of nodes, arcs, and commodities. 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 in category "R," consists of 54 instances divided into 6 groups ranging from r13 to r18. 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 costs. If the "C" value is small, the arc capacity is loose, and if the "C" value is large, the arc capacity is tight relative 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 access 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), (10, 2000 seconds) and (20, 2000 seconds). We set the parameters as $\varepsilon = 0.01$, $B = 150$, $\Delta B = B_{\min} = 10$, $\Delta M = 2$, $M_{\min} = 1$, $ITE_{\min} = 50$, 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 PTAB is the results by parallel tabu search, TGLS by tabu assisted guided local search, CPM by cutting-plane matheuristic, and TFM by three-phase matheuristic respectively. Column RBB is the result by the capacity scaling and restricted branch-and-bound excluding local branching. Column 10-1000, 10-2000 and 20-2000 are the results of the combined matheuristic, and the first number indicates M and the second number indicates T . The average gap of three-phase matheuristic, which is the best result among four previous heuristics, is 1.18%. The average gap of the capacity scaling and restricted branch-and-bound is 0.92%, and it is better than that of three-phase matheuristic. Meanwhile the average gaps of the three combined

Table 3. Average Computation Time for C-category problems (seconds)

PTAB	TGLS	CPM	TFM	RBB	10-1000	10-2000	20-2000
3600.0	2400.0	3581.4	7785.0	524.1	3780.2	6928.4	8637.0

Table 4. Computation Time for C-category problems (seconds)

N/A/K/FC	CPM	TFM	RBB	10-1000	10-2000	20-2000
20/230/200/VL	635	5460	91.5	2500.7	3876.1	5579.7
20/230/200/FL	1026	5520	219.3	3652.1	5122.4	8836.8
20/230/200/VT	544	4680	46.3	1474.1	1465.1	4223.3
20/230/200/FT	634	5040	556.0	3831.0	12808.4	9577.5
20/300/200/VL	432	9720	78.8	4254.2	9722.6	8596.3
20/300/200/FL	1027	6600	421.1	5205.2	6683.2	9056.2
20/300/200/VT	283	5400	36.5	1267.5	1261.1	1976.6
20/300/200/FT	958	8460	605.5	3492.9	5489.8	7863.3
30/520/100/VL	225	5100	14.9	496.8	496.9	1606.7
30/520/100/FL	1630	7080	412.7	5803.4	6435.8	10337.4
30/520/100/VT	82	6180	10.2	2791.5	2796.0	5332.9
30/520/100/FT	870	9900	422.3	6507.8	14119.4	21799.0
30/520/400/VL	6830	13260	193.7	3283.5	6986.8	9283.5
30/520/400/FL	10529	8760	806.3	3009.7	5159.1	5213.6
30/520/400/VT	5413	12600	82.3	3426.5	13336.9	12285.8
30/520/400/FT	10696	8160	1292.9	4724.4	9752.0	11929.2
30/700/100/VL	101	4440	8.1	228.4	226.1	381.2
30/700/100/FL	631	9180	68.7	3414.0	12898.8	9823.1
30/700/100/VT	125	8100	16.7	4180.3	5349.3	8587.2
30/700/100/FT	309	9120	51.4	5078.8	8975.1	8211.8
30/700/400/VL	7893	8160	497.5	4086.0	9616.6	11660.6
30/700/400/FL	15723	4800	1317.7	5377.6	5742.8	7745.6
30/700/400/VT	8405	13320	1825.4	4953.2	7064.8	14481.3
30/700/400/FT	10953	7800	3503.5	7686.1	10897.5	12900.6

Table 5. Average Gap for R-category problems (%)

PTAB	CPM	TFM	RBB	10-1000	10-2000	20-2000
4.56	2.75	0.76	0.75	0.20	0.18	0.15

matheuristic are 0.69%, 0.66% and 0.65%, and these are less than three-fifth of that of three-phase matheuristic.

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. Three-phase matheuristic find the optimal solutions for two instances out of 24 instances, meanwhile the three combined matheuristic find the optimal solutions for eight or nine instances. Additionally, the combined matheuristic 10-2000 find the new best upper bound for 11 instances and 20-2000 find the new best upper bound for 13 instances out of 16 instances to the exclusion of the instances of which the optimal values are found. Consequently, the combined matheuristic can find the best solutions for 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 and restricted branch-and-bound, and three combined matheuristic. The computational times for four previous heuristics are reported in their papers. Due to the fact that different CPUs are used, these computation times cannot be compared directly. The previous two heuristics were performed in a given computation time limit such as 2400 or 3600 seconds. The average computational time by the capacity scaling and restricted branch-and-bound is relatively short such as 524.1 seconds. The three combined matheuristic can be solved within a reasonable average computation time such as 3780.2, 6928.4 or 8637.0 seconds. Table 4 shows the detailed computation times for C-category problems.

Table 5 displays the average gaps of the results for

Table 6. Results for R-category problems

	C	F	OPT/LB	PTAB	CPM	TFM	RBB	10-1000	10-2000	20-2000
		F01	147349.0^{O}	**147349**	148494	**147349**	**147349.0**	**147349.0**	**147349.0**	**147349.0**
	C1	F05	277891.0^{O}	281668	298494	279389	**277891.0**	**277891.0**	**277891.0**	**277891.0**
		F10	385396.0^{O}	400656	417877	**385396**	388408.0	**385396.0**	**385396.0**	**385396.0**
		F01	155887.0^{O}	156585	**155887**	156616	156386.0	**155887.0**	**155887.0**	**155887.0**
r13	C2	F05	295180.0^{O}	307180	298582	**295180**	296556.0	**295180.0**	**295180.0**	**295180.0**
		F10	431140.0^{O}	437396	454625	434383	433117.0	**431140.0**	**431140.0**	**431140.0**
		F01	218787.0^{O}	223541	224632	**218787**	221091.0	**218787.0**	**218787.0**	**218787.0**
	C8	F05	491560.0^{O}	510887	497877	492959	495090.0	**491560.0**	**491560.0**	**491603.0**
		F10	782049.0^{O}	839174	798947	791213	796897.0	**782049.0**	**782049.0**	**782049.0**
		F01	422709.0^{O}	427872	423538	**422709**	**422709.0**	**422709.0**	**422709.0**	**422709.0**
	C1	F05	784626.0^{O}	811102	812423	**784626**	785754.0	**784626.0**	**784626.0**	**784626.0**
		F10	1119569.0^{O}	1157500	1156950	1137820	1126879.0	*1120185.0*	*1120185.0*	*1120185.0*
		F01	452591.0^{O}	458240	457421	453434	**452591.0**	**452591.0**	**452591.0**	**452591.0**
r14	C2	F05	883051.0^{O}	917832	890673	891138	887334.0	**883051.0**	**883051.0**	**883051.0**
		F10	1296477.0^{O}	1356910	1336490	1307770	1303528.0	**1296477.0**	**1296477.0**	**1296477.0**
		F01	702614.2^{O}	720494	708444	**702614**	704203.0	**702614.2**	**702614.2**	**702614.2**
	C8	F05	1685913.1^{L}	1795650	1706840	1693240	1692473.3	1689071.0	1689071.0	*1688981.0*
		F10	2755700.0^{O}	2997290	2772750	2769360	2786173.0	2756790.0	2757353.0	**2755700.0**
		F01	1017740.0^{O}	1032640	1019180	**1017740**	1018193.0	**1017740.0**	**1017740.0**	**1017740.0**
	C1	F05	2008205.5^{O}	2082990	2028140	2055803	2012100.0	2009112.0	**2008205.5**	**2008205.5**
		F10	2904651.8^{L}	3116770	3003990	2971500	2985328.0	2973425.5	2978364.0	*2966146.0*
		F01	1174517.5^{O}	1191440	1182020	1174520	1174959.0	**1174517.5**	**1174517.5**	**1174517.5**
r15	C2	F05	2535837.5^{L}	2698680	2574700	2561060	*2553657.0*	*2553657.0*	*2553657.0*	*2553657.0*
		F10	3947038.7^{L}	4310340	4176330	4045030	4041668.5	4032778.0	4034439.0	*4010444.0*
		F01	2401115.0^{O}	2465650	2403330	2408210	2402126.2	**2401115.0**	**2401115.0**	**2401115.0**
	C8	F05	5795320.0^{O}	5969370	5797170	5796510	5797171.0	5796508.0	5796508.0	**5795320.0**
		F10	9105014.0^{O}	9304650	9115830	9129360	**9105014.0**	**9105014.0**	**9105014.0**	**9105014.0**

R-category problems. The average gap of three-phase matheuristic is 0.76%. In contrast, the average gap of the capacity scaling and restricted branch-and-bound is 0.75% and that of the three combined matheuristic are only 0.20%, 0.18% and 0.15%. The gap of the capacity scaling and restricted branch-and-bound is as the same as that of three-phase matheuristic, and the gap of the combined matheuristic 20-2000 is only about one-fifth of that of three-phase matheuristic.

Table 6 and 7 displays the detailed results for R-category problems. The three combined matheuristic can find the optimal solutions for 42 instances out of 54 instances and find the new best upper bound for 14 instances to the exclusion of the instances of which the optimal values are found. Consequently, the combined matheuristic can find the best solutions for all instances of R-category problems.

Table 8 displays the average computation times in CPU seconds for three previous heuristics, these computational times of which are reported in their papers, the capacity scaling and restricted branch-and-bound, and the three combined matheuristic. When compared to previous heuristics, the computational times by capacity scaling are relatively short such as 251.3 seconds. The three combined matheuristic can be solved within a reasonable computation time such as 2283.5, 3691.4 or 5074.2 seconds.

According to the results, the capacity scaling and restricted branch-and-bound can offer good quality results and the computational time is relatively short. The combined capacity scaling, restricted branch-and-bound and local branching matheuristic can obtain good solutions within a reasonable computation time, and improve the current best solutions or find the optimal solutions for the all C and R-category problems.

7. Conclusion

This paper presents the combined capacity scaling, restricted branch-and-bound and local branching matheuristic, which is applied to column generation and cutting plane techniques for SNDB characterized by the strong formulation. The performance of the proposed matheuristic is evaluated by solving C and R-category problems. The numerical results are satisfactory, while the combined matheuristic can find the best new solutions or the optimal solutions for all instances of C-category and R-category problems.

The capacity scaling excluding local branching can offer good quality results and the computational effort can be reduced considerably. The combined capacity scaling, restricted branch-and-bound and local branching matheuristic can offer highest quality results and outperforms previous heuristics proposed in the literature.

Acknowledgments

This work was supported by Grant-in-Aid for Scientific Research 25350454.

References

Table 7. Results for R-category problems

	C	F	OPT/LB	PTAB	CPM	TFM	RBB	10-1000	10-2000	20-2000
r16	C1	F01	140082.0^{O}	140149	142797	**140082**	**140082.0**	**140082.0**	**140082.0**	**140082.0**
		F05	248703.0^{O}	261775	277712	**248703**	253681.0	**248703.0**	**248703.0**	**248703.0**
		F10	340641.0^{O}	360884	359648	350958	348522.0	**340641.0**	**340641.0**	**340641.0**
	C2	F01	142381.0^{O}	143921	159168	142605	142507.0	**142381.0**	**142381.0**	**142381.0**
		F05	259313.0^{O}	273024	285509	260822	**259313.0**	**259313.0**	**259313.0**	**259313.0**
		F10	361626.0^{O}	387601	376114	368572	367947.0	**361626.0**	**361626.0**	**361626.0**
	C8	F01	179639.0^{O}	185397	183475	180228	184624.0	**179639.0**	**179639.0**	**179639.0**
		F05	387360.0^{O}	419945	393541	388180	394679.0	**387360.0**	**387360.0**	**387360.0**
		F10	596660.0^{O}	647212	610267	598835	612353.0	597997.0	597997.0	**596660.0**
r17	C1	F01	364784.0^{O}	365913	368841	365788	**364784.0**	**364784.0**	**364784.0**	**364784.0**
		F05	675029.0^{O}	702957	717089	676528	678282.0	**675029.0**	**675029.0**	**675029.0**
		F10	947172.0^{O}	1026040	991205	966116	969046.0	**947172.0**	**947172.0**	**947172.0**
	C2	F01	382593.0^{O}	389249	388625	384579	383076.0	**382593.0**	**382593.0**	**382593.0**
		F05	734117.0^{O}	786198	744146	741744	738944.0	**734117.0**	**734117.0**	**734117.0**
		F10	1066292.0^{O}	1159440	1126380	1086640	**1066292.0**	**1066292.0**	**1066292.0**	**1066292.0**
	C8	F01	528923.0^{O}	539817	535474	529876	529395.0	**528923.0**	**528923.0**	**528923.0**
		F05	1213747.3^{L}	1323330	1241990	1230910	1228640.0	1223768.0	*1223698.0*	1224001.0
		F10	1974056.8^{L}	2207590	2064630	1999950	2007426.0	1997521.0	1997521.0	*1996006.3*
r18	C1	F01	844211.0^{O}	864425	848636	844260	846124.0	**844211.0**	**844211.0**	**844211.0**
		F05	1572707.0^{O}	1627700	1615730	1588890	1575173.0	**1572707.0**	**1572707.0**	**1572707.0**
		F10	2203024.0^{L}	2366280	2286290	2264470	2258224.0	2245541.0	*2229722.0*	*2229722.0*
	C2	F01	940627.8^{O}	962402	945562	944708	942127.0	**940627.8**	**940627.8**	**940627.8**
		F05	1842533.8^{L}	1958160	1904830	1883870	1879801.0	1873989.0	*1871741.0*	1873989.0
		F10	2734321.6^{L}	2986000	2809030	2806020	2801704.0	*2792872.0*	2793627.0	2793316.7
	C8	F01	1525156.8^{L}	1613790	1547430	1542500	1535767.9	1534743.0	*1533160.0*	1533402.9
		F05	3961276.9^{O}	4268580	3961280	4039410	3974511.6	3974511.6	**3961276.9**	**3961276.9**
		F10	6550762.5^{O}	7194120	6602450	6603500	6579400.3	6570234.2	*6554028.5*	*6554028.5*

Table 8. Average Computation Time for R-category problems (seconds)

PTAB	CPM	TFM	RBB	10-1000	10-2000	20-2000
3600.0	1004.1	5711.1	251.3	2283.5	3691.4	5074.2

[1] J. Andersen, M. Christiansen T. G. Crainic, and R. Grønhaug, “Branch and price for service network design with asset management constraints”, Transportation Science, Vol. 45, No. 1, pp. 33–49, 2011.

[2] J. Andersen, T. G. Crainic, and M. Christiansen, “Service network design with management and coordination of multiple fleets”, European Journal of Operational Research, Vol. 193, No. 2, pp. 377–389, 2009.

[3] R. Bai, G. Kendal, and J. Li, “An efficient guided local search approach for service network design problem with asset balancing”, ICLSIM, Vol. 1, pp. 110–115, 2010.

[4] R. Bai, G. Kendal, R. Qu, and J. Atkin, “Tabu assisted guided local search approaches for freight service network design”, Information Science, Vol. 189, No. 15, pp. 266–281, 2012.

[5] A. Balakrishnan, T. L. Magnanti, and P. Mirchandani, “Network design”, In M. Dell’Amico, F. Maffioli and S. Martello, editors, Annotated Bibliographies in Combinatorial Optimization, pp. 311–334. John Wiley & Sons, New York, 1997.

[6] M. Chouman and T. G. Crainic, “MIP-based tabu search for service network design with design-balanced requirements”, Technical Report CIRRELT-2011-68, Centre de recherche sur les transports, Universitè de Montrèal, 2011.

[7] M. Choumann and T.G. Crainic, “Cutting-plane matheuristic for service network design with design-balanced requirements”, Transportation Science, 2014.

[8] A. M. Costa, “A survey on benders decomposition applied to fixed-charge network design problems”, Computers and Operations Research, Vol. 32, pp. 1429 – 1450, 2005.

[9] T. G. Crainic, “Long-haul freight transportation”, In R. W. Hall, editor, Handbook of Transportation Science, pp. 451–516. Kluwer Academic Publishers, 2003.

[10] M. Matteo Fischetti and A. Lodi, “Local branching”, Mathematical Programming, Vol. 98, No.1-3, pp. 23–47, 2003.

[11] B. Gendron, T. G. Crainic, and A. Frangioni, “Multicommodity capacitated network design”, Technical Report CIRRELT-98-14, Centre de recherche sur les transports, Universitè de Montrèal, 1997.

[12] N. Katayama and S. Yurimoto, “Combining capacity scaling and local branch approaches for the logistics network design problem”, In Proceedings

of the 21th International Conference on Production Research, Stuttgart, Germany, 2011.

[13] T. L. Magnanti and R. T. Wong, “Network design and transportation planning: Models and algorithms”, Transportation Science, Vol. 18, pp. 1–55, 1984.

[14] M. Minoux, “Network synthesis and optimum network design problems: Models, solution methods and applications”, Networks, Vol. 19, pp. 313–360, 1989.

[15] M.B. Pedersen, T.G. Crainic, and O.B.G, “Madsen. Models and tabu search metaheuristics for service network design with asset-balance requirements”, Transportation Science, Vol. 43, pp. 158–177, 2009.

[16] D.M. Vu, T.G. Crainic, and M. Toulousein, “A three-phase matheuristic for capacitated multi-commodity fixed-cost network design with design-balance constraints”, Journal of Heuristic, Vol. 19, pp. 57–795, 2013.

[17] R. T. Wong, “Location and network design” , In M. O’heEigeartaigh, J. Lenstra, and A. RinnooyKan, editors, Combinatorial Optimization Annotated Bibliographies, pp. 129–147. John Wiley & Sons, New York, 1985.

[18] M. Yaghini and M. Rahbar, “Multicommodity network design problem in rail freight transportation planning”, Procardia Social and Behavioral Sciences, Vol. 43, pp. 728–739, 2012.

Naoto Katayama is a professor in the Faculty of Distribution and Logistics Systems, Ryutsu Keizai University. He received his Ph.D. degree in Ryutsu Keizai University in 2010. His research interests are in the area of network design problems. He is a member of the Japan Industrial Management Association (JIMA), the Operations Research Society of Japan (ORSJ), Japan Society of Civil Engineers (JSCE), and the Institute for Operations Research and Management Science (INFORMS)