A Multilevel Cooperative Tabu Search Algorithm for the Capacitated Multicommodity Network Design Problem

by

Ye Li

A thesis
presented to the University of Manitoba
in partial fulfilment of the
requirements for the degree of
Master of Science
in
Computer Science

Winnipeg, Manitoba, Canada, 2005

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

A Multilevel Cooperative Tabu Search Algorithm for the Capacitated Multicommodity Network Design Problem

BY

Ye Li

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

Of

Master of Science

Abstract

The objective of the capacitated multicommodity network design (CMND) problem is to identify the optimal design of a network, i.e., to select a subset of links such that a minimal total system cost (the sum of the fixed and routing costs) is found, while satisfying the demand for transportation on links with limited capacities. This problem is NP-hard. The focus of this thesis is to design an effective heuristic, called multilevel cooperative tabu search (MCTS) algorithm, for the CMND problem. It is a first attempt to combine the multilevel paradigm, parallel cooperative search heuristics and tabu search schemes to solve the CMND problem. The contribution is to build a multilevel structure of the search space, specify each search thread in cooperation and define the cooperation operators. Thereby obtaining a solution with less system cost than the ones obtained by the other methodologies is the major goal of this research.

Acknowledgements

I would like to express my gratitude to all who helped, accompanied, supported and encouraged me to accomplish this thesis.

I want to thank the Department of Computer Science, University of Manitoba for giving me the oppotunity to pursue my master's degree. I also want to acknowledge Centre de Recherche sur les Transports, Université de Montréal for allowing me to use the Centre resources and to do research work.

I am deeply indebted to my supervisors Dr. Michel Toulouse from the Department of Computer Science, University of Manitoba, and Dr. Teodor Crainic from Centre de Recherche sur les Transports, Université de Montréal. Their stimulating suggestions and guidances helped me during the research and writing of this thesis.

I am grateful to my friends, Zimpi Helen Komo and Manikandaprabhu Ganeshan, whose valuable comments, encouragements and confidence in me throughout my research supported me to complete this thesis.

Especially, I would like to give my appreciation to my parents and sister who persistently inspirate and support me not only for my research work but also in my life.

Finally, I would like to acknowledge the contribution of the Natural Sciences and Engineering Council of Canada which, through its Discovery Grant program, has funded this research project.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

The Capacitated Multicommodity Network Design (CMND) problem can be represented by a weighted directed graph $G = (N, A)$ where the set of nodes N is partitioned into three non-overlapping subsets: source nodes, sink nodes and transition nodes. Each sink node represents the demand for one commodity d. The demand for commodity d is satisfied by a single source node and must transit from the source node to the sink node using arcs and transition nodes in the graph G. Each arc $(i, j) \in A$ has a capacity u_{ij}, a fixed cost f_{ij} and a routing cost c_{ij}^d. The routing cost c_{ij}^d represents the cost to transit one unit of commodity d on arc (i, j). The capacity u_{ij} is an upper bound on the number of units from different commodities that can be transitted on arc (i, j). The fixed cost f_{ij} represents the cost for having this arc in the network design. The capacitated multicommodity network design problem consists of determining $\overline{y} \subseteq A$, a subset of arcs from G which minimizes the cost to transit commodities from source to sink nodes. The subset $\overline{y}$ is a "feasible solution" if all the demands are satisfied and the total number of commodity units that transit on each arc (i, j) is less or equal to the capacity u_{ij}. The solution $\overline{y}$ is an "optimal solution" if there exists no other subset $\overline{y}' \subseteq A$ for which the

sum of the fixed costs and routing costs is smaller than the one for the subset $\overline{y}$.

The capacitated multicommodity network design problem plays a significant role in modeling many network related applications. For example, it is used to design telephone networks, railway networks or computer networks in order to minimize the cost of communications, transportation of merchandises or the cost to upload and download data, images and videos. The CMND is a core problem in many areas of Operations Research, Computer Science, Applied Mathematics, Engineering and Management. Unfortunately, there is no known efficient approach to solve the CMND problem using computers. In fact, the CMND problem has been proved to be NP-hard [1, 24, 25], therefore it is unlikely that we will ever be able to solve it exactly for large problem instances.

As usually the case for NP-hard problems, in order to solve them, one has to settle for a feasible solution which is not necessary optimal. Different approaches for the CMND problem have been proposed in the literature: simplex-based cutting plane methods [3], Lagrangian relaxations [5], and neighborhood search heuristics [7, 12]. In this thesis, we propose an approach called Multilevel Cooperative Tabu Search (MCTS) which combines neighborhood search and parallel computing to obtain better approximations of optimal solutions for the CMND problem. More specifically, we have adapted the best known neighborhood search heuristic for the CMND problem [12] to be used as the search subroutine of a multilevel cooperative algorithm for the CMND problem. The main contributions of this thesis are the followings:

1. the adaptation of the cycle-based tabu search to multilevel search;

2. the definition and implementation of a strategy providing an initial set of search spaces of a CMND problem instance to be searched in parallel by independent cycle-based tabu search processes;

3. the design of operators which refine search spaces and reinitialize the exploration in each search space by a cycle-based tabu search procedure.

The thesis is organized as follows: Chapter 2 provides an intuitive description of the CMND problem followed by its mathematical formulation as well as general introduction to the application of neighborhood search methods to this problem. Chapter 3 describes the cycle-based tabu search algorithm. Chapter 4 introduces the multilevel cooperative method through an illustration of the application of this method to the hypergraph partitioning problem. Chapter 5 describes the multilevel cooperative algorithm for the CMND problem. Parameter calibrations are described in Chapter 6, which also reports and analyzes the results of our experimentations. The conclusion and potential research directions are provided in Chapter 7.

Chapter 2

Background

The capacitated multicommodity network design problem is a mixed-integer programming problem, with real and integer decision variables. The integer (0-1) decision variables represent the choice of including or not in the design a specific arc (i, j) from the graph G. The continuous decision variables represent an network flow optimization problem that needs to be solved for each network design. The CMND problem is a combinatorial optimization problem (in its design arcs) with an underlying network flow problem. These two main optimization structures are keys to understanding the mathematical formulation of the problem as well as in the solution strategies that are proposed to solve it. In this chapter, we first define the underlying problems associated with CMND. Next, we introduce the mathematical formulation of the problem that will be used throughout this thesis. Finally, we provide background on applications of neighborhood search methods to solve the CMND problem.

2.1 Definitions

Consider the following scenario: "suppose wheat is stored in warehouses in Winnipeg and Saskatoon which needs to be transported to customers in Montreal, Ottawa and Halifax. Each warehouse can ship up to 1500 units. Montreal, Ottawa and Halifax have customers which require 850, 700 and 600 units, respectively. Wheat would have to pass through other cities such as Thunderbay or Chicago before getting to their various destinations."

Applications involving transportation of goods from sources to destinations can be described using a mathematical model called the *network flow problem.* The network flow problem can be described using a directed graph. Arcs in the graph stand for the routes in the application that can be used to transport commodities such as wheat. Generally speaking, nodes in the graph are partitioned in three categories: *source* nodes, which indicate the origin points (cities such as Winnipeg and Saskatoon as shown in the above example); *sink* nodes, which indicate the destination points (cities such as Montreal, Ottawa and Halifax); and finally, *transition* nodes, which represent points (the cities of Thunderbay and Chicago) through which the routes have to pass.

Transporting wheat from its origin to the required destination incurs a cost, called a *routing cost*, which represents the cost per unit of commodity transported. In the model, we associate to each arc the cost of routing one unit of flow of the commodity. The network flow problem models some basic resource and associated costs which can then be used to express the overall cost of a specific transport application. The optimization version of network flow problem, i.e., finding routes that minimize the overall cost of a transportation application, is the *minimum cost flow problem.*

Wheat has to be conveyed using some transportation equipments (trucks, ships, trains, or airplanes) and each equipment has a limited carrying capacity. Hence, in the

corresponding model, each arc also has an associated capacity that denotes the maximum amount of flow traveling on the arc. This problem can be termed as the *capacitated minimum cost flow problem.*

Instead of transporting a single commodity such as wheat, one may envision a network flow problem where several kinds of commodities have to be transported simultaneously using the same underlying network. The problem of transporting more than one commodity at minimum cost is the *capacitated multicommodity minimum cost flow (CMCF) problem.* In the CMCF problem, each commodity has its own origins and destinations. Furthermore, to each arc and to each commodity is associated a different cost per unit of flow. Hence, it increases the difficulty of the problem. The objective of CMCF problem is to allocate the limited capacity of each arc to the different commodities in order to minimize the cost of transporting the goods from origins to destinations. The CMCF problem is a linear programming problem that can be solved using a commercial software called CPLEX. It is routinely used to solve this problem computationally. The CMCF problem is represented by the continuous decision variables in the CMND problem. This sub-problem needs to be solved for each network design.

In some transportation applications, one needs to take into account the cost of building/renting the transportation infrastructure associated to each arc in the network flow graph. The cost associated to the transportation infrastructure is termed the *fixed cost.* Hence, the whole system cost which aims to be minimal, must include the routing cost and the fixed cost. This type of transportation application is modeled as the *capacitated multicommodity network design* (CMND) problem and it consists of selecting a subset of arcs in the network that minimizes the whole system cost (the sum of the fixed cost of building system infrastructure and cost of routing commodities), while satisfying the transportation requirements for several commodities simultaneously.

2.2 Mathematical formulation

Let $G = (N, A)$ be a directed graph where N is a set of nodes and A a set of directed arcs. Without loss of generality, we assume that all arcs $(i, j) \in A$ are *design arcs*, i.e., arcs that can be opened (included in the network solution) or closed (excluded from the network solution) in order to form a design (solution) of the problem. We provide an arc-based mathematical formulation of CMND problem:

$$\min \xi(x, y) = \sum_{d \in D} \sum_{(i,j) \in A} c_{ij}^d x_{ij}^d + \sum_{(i,j) \in A} f_{ij} y_{ij}$$

$$\text{where} \quad \sum_{j \in N^+(i)} x_{ij}^d - \sum_{j \in N^-(i)} x_{ji}^d = \begin{cases} r^d & \text{if } i = o(d) \\ -r^d & \text{if } i = s(d) \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in N, \forall d \in D \qquad (1)$$

$$\sum_{d \in D} x_{ij}^d \leq u_{ij} y_{ij} \qquad \forall (i, j) \in A \qquad (2)$$

$$x_{ij}^d \geq 0 \qquad \forall (i, j) \in A, \forall d \in D \qquad (3)$$

$$y_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A \qquad (4)$$

The meaning of the variables is the following:

D :	the set of commodities to be moved over network G.
d :	one commodity $d \in D$.
$o(d)$:	each commodity d has a single origin $o(d)$.
$s(d)$:	each commodity d has a single destination $s(d)$.
r^d:	for each commodity d, there is a flow demand of r^d units

y_{ij}: the design variable that is equal to 1 if arc (i, j) is chosen in the design and equal to 0 if it is not chosen.

x_{ij}^d: the flow distribution decision variable that denotes the amount of flow of commodity $d \in D$ on arc (i, j).

f_{ij}: the fixed cost of arc (i, j).

c_{ij}^d: the routing cost of one unit flow of commodity d on arc (i, j).

u_{ij}: capacity of arc (i, j).

$N^+(i)$: the set of nodes j such that (i, j) is the arc from node i to j.

$N^-(i)$: the set of nodes j such that (j, i) is the arc from node j to i.

The objective function $\xi(x, y)$ which must be minimized, represents the system design cost that includes the fixed cost of the arcs that are chosen in a design and the routing cost which is proportional to the number of units of commodities. Constraint (1) shows the network flow conservation. That is, for each source node i, there is only the flow of r^d units for commodity d leaving this node i; for each sink node i, there is only the flow of r^d units for commodity d entering the node i; for each transition node i (other than source nodes or sink nodes) and commodity d, the amount of the flow entering node i is equal to the amount of the flow leaving node i. Constraint (2) denotes, for each arc (i, j), the total flow of all the commodities can not exceed its capacity. Constraint (3) illustrates that the flow distribution decision variable x_{ij}^d is not negative while constraint (4) shows that the design variable y_{ij} is either 0 or 1.

From this mathematical formulation, we notice that CMND is a capacitated multicommodity minimum cost flow problem (CMCF) once a design $\overline{y}$ is obtained. This CMCF can be written as

$$\min \xi(x(\overline{y})) = \sum_{d\in D}\sum_{(i,j)\in A(\overline{y})} c_{ij}^d x_{ij}^d$$

$$\text{where} \quad \sum_{j\in N^+(i)} x_{ij}^d - \sum_{j\in N^-(i)} x_{ji}^d = \begin{cases} r^d & \text{if } i = o(d) \\ -r^d & \text{if } i = s(d) \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in N, \forall d \in D$$

$$\sum_{d\in D} x_{ij}^d \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A(\overline{y})$$

$$x_{ij}^d \geq 0 \qquad \forall (i,j) \in A(\overline{y}), \forall d \in D$$

where $A(\overline{y})$ symbolizes the set of arcs corresponding to the design $\overline{y}$. Therefore, we said that CMND is the combinatorial problem of choosing of subgraph over the original network plus the optimization problem of the CMCF problem. That is, a solution of CMND problem can be seen as a design $\overline{y}$, assigning a value of 0 or 1 to each design variable, plus the optimal flow $x^*(\overline{y})$ of the corresponding CMCF problem. Hence, the objective function $\xi(x, y)$ can be also written as:

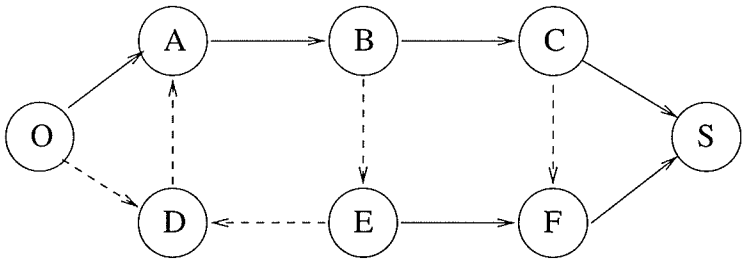
$$\xi(\overline{y}, x^*(\overline{y})) = \sum_{(i,j)\in A(\overline{y})} f_{ij}\overline{y}_{ij} + \xi(x^*(\overline{y}))$$

Even though considering only the fixed cost, the CMND is NP-hard because it has to consider all the possible subgraphs that satisfy the constraints of the transportation. In the CMND problem, this combinatorial problem of the subgraphs is combined with the CMCF problem. This makes the CMND problem even more difficult to solve. No exact method exists that can solve the CMND problem in a reasonable amount of time for application instances of a realistic size.

2.3 Neighborhood search techniques

The first search heuristics proposed for the capacitated network design problem were simple add/drop procedures. These procedures evaluate the impact on the cost function

of adding or removing an arc from the design [21, 27]. Given a current feasible solution, add/drop heuristics compute the cost of all feasible solutions that can be obtained by adding or removing a single arc from the current solution. The set of feasible solutions that can obtained in this way form the *neighborhood* of the current solution. In Figure 2.1, solid arcs represent a current feasible solution of an instance of the network design problem while dotted lines represent the arcs that are not chosen in the design. Two solutions in the neighborhood of the current solution can be obtained by adding arc (C, F) or dropping arc (C, S). Once the cost of all neighbors has been computed, add/drop procedures select the solution with the lowest cost (best improvement). If the cost of the chosen solution is better than the current solution, the chosen solution becomes the new current one and the neighborhood evaluation is repeated. Eventually, add/drop heuristics reach a point where there is no improving solution in the neighborhood of the current solution. Then, the add/drop heuristic stops and returns the current solution as the best solution to a network design problem instance.



(FixedCost, RoutingCost, Capacity, Flow)

Figure 2.1: Add/Drop techniques for network design

Add/drop procedures only make moves which improve the cost function. This is a serious drawback of this method, limiting their capacity to explore the solution space. For example, once an add/drop procedure reaches the first local optimum, the search will

end since, by the definition of a local optimum, no add or drop move can improve the cost function. To address this kind of limitations, Crainic, Gendreau and Farvolden [7] have applied the tabu search metaheuristic to the CMND problem. Tabu search is a neighborhood search heuristic which, unlike pure add/drop procedures, can explore beyond local optima by making moves which do not necessarily improve the cost of the current solution. During the period where the cost function is degraded, tabu search seeks to find a new plateau (or valley) in the solution space from which the sequence of improving moves can resume. This leads to a more extensive exploration of the solution space, yielding better solutions.

Neighborhoods based on adding (dropping) arcs are not ideal for a problem like CMND. For example, in Figure 2.1, adding arc (E, D) into the network will not produce any significant change to the cost function since adding arc (E, D) does not generate any new path between the origin and the destination (there is only one path $\{(O, A), (A, B), (B, C), (C, S)\}$ between the origin (node O) and the destination (node S). This situation will not be changed by adding arc (E, D)). In this context, Ghamlouche, Crainic and Gendreau [12] have proposed a new tabu search heuristic for the CMND problem with a completely different type of neighborhood, called cycle-based neighborhood. Cycle-based neighborhood is a key component of proposed algorithm. Details about this neighborhood structure will be discussed in Chapter 3.

Another heuristic for the CMND problem has been proposed recently which combines the cycle-based tabu search and the path relinking approach [13]. The cycle-based tabu search produces promising solutions, called *Elite solutions*, which are stored in a set called the *reference set*. The aim of the path relinking procedure is to find a path between two elite solutions in the reference set in order to intensify the exploration in the solution space of elite solutions. The solutions in the path are obtained by introducing attributes of elite solutions in the reference set to guide the search. Experimental results

indicate that the proposed path relinking method currently offers the best performance for the CMND problem.

Finally, the computation of neighborhood search heuristics can be speed up in different ways through parallel implementations of the heuristics [8]. One parallelization technique consists of the initiation of several concurrent tabu search processes from different regions of the solution space. During the computation, processes exchange information about their own exploration of the solution space. These parallelization techniques are called *cooperative searches*, and they have been applied in parallelization of the tabu search method [34]. A parallel cooperative tabu search method has been proposed and tested recently for the CMND problem [6]. As shown in [35], search parameters and cooperation interact with each other in a very complex manner. This makes it difficult to predict the search behavior of the individual processes. For example, it is difficult to prevent the search paths from overlapping with one another. In this thesis, I will use a new approach for cooperation, *parallel multilevel cooperative tabu search*, which provides a greater control on how search parameters and cooperation impact on each other.

Chapter 3

Cycle-based Tabu Search

In this chapter, we describe the cycle-based tabu search algorithm for the CMND problem [12]. A variation of this algorithm will be used as a search heuristic in our multilevel cooperative algorithm for the CMND problem. The key feature of the cycle-based tabu search algorithm is its neighborhood structure, called the cycle-based neighborhood. We first describe this neighborhood structure. Then, we elaborate on the tabu search algorithm with the cycle-based neighborhood structure for the CMND problem.

3.1 Neighborhood definition

Let $G = (N, A)$ be an directed graph representing a network design. A *path* in G is a sequence of closed or opened arcs such that the origin node of each arc is the destination node of the preceding arc in the sequence and no vertex appears more than once. That is, all arcs on a path follow the same direction. For example, in Figure 3.1 (a), the sequence $\langle (A, B), (B, C), (C, G) \rangle$ is a path. We define a *chain* like a path where the arcs do not necessary follow a single direction. A *cycle* is a closed chain. In Figure 3.1 (a), the

chain $\langle(E,B),(B,C),(C,F),(E,F)\rangle$ form a cycle.

A cycle in a cycle-based neighborhood structure is composed of two paths connecting the same pair of nodes. For example, cycle $\langle(E,B),\ (B,C),\ (C,F),\ (E,F)\rangle$ in Figure 3.1 (a) contains two paths $\langle(E,B),(B,C),(C,F)\rangle$ and $\langle(E,F)\rangle$ which connect to the same pair of nodes E and F. The purpose of the two paths forming a cycle is to allow for the flow of commodities on one path of the cycle to be deviated on the other path of this same cycle. Deviating flow in such cycles change the cost of the current solution $\overline{y}$, and provide a neighbor to $\overline{y}$. However, the number of neighbors based on flow deviation can be huge, which makes the computation time to evaluate the neighborhood excessive. To reduce the cost of computing neighbors, only those flow deviations that close at least one opened arc in the current solution are considered.

Let

$$\Gamma(\overline{y}) = \{\sum_{d\in D} x_{ij}^d > 0 : (i,j) \in A(\overline{y})\}.$$

be the set of the different volumes of flow on the open arcs in the current solution $A(\overline{y})$. The set $\Gamma(\overline{y})$ gives us the size of the flow deviations that can constitute neighbor solutions. Let the *residual capacity* of a cycle to be the maximum flow which can be deviated around a cycle. To deviate a flow of value γ on a given cycle, the residual capacity of this cycle should be at least γ. For each value $\gamma \in \Gamma(\overline{y})$, we compute the γ-*residual network*, which is a network that identifies all the cycles capable of deviating at least γ units of flow. The γ-residual network is obtained from the network corresponding to the current solution $\overline{y}$ by replacing each arc in the network $\overline{y}$ by at most two arcs $(i,j)^+$ and $(j,i)^-$. Arc $(i,j)^+$ represents γ units of flow that can be added to arc (i,j). Arc $(j,i)^-$ represents γ units of flow that can be eliminated from arc (i,j). An arc $(i,j)^+$ is included in the γ-residual network iff the residual capacity (i.e., the subtraction of the capacity and the flow amount) of arc (i,j) is greater or equal to γ. Arc $(j,i)^-$ is included in the γ-residual network iff the total flow on arc (i,j) is greater or equal to

γ, i.e. $\sum_{d \in D} x_{ij}^d \geq \gamma$. Figure 3.1 (b) shows an example of 2-residual network for the current network design $\overline{y}$ shown in Figure 3.1 (a).

Among all the cycles in a γ-residual network, there exists at least one arc $(i,j)^-$ corresponding to an opened arc (i,j) in the network $\overline{y}$ with total flow γ. The arc (i,j) can be closed, and its flow can be deviated in the network $\overline{y}$ on the alternate path of one of the cycles in which $(i,j)^+$ is involved in the γ-residual network.

From a γ-residual network, we can extract all the neighbors of the current solution $A(\overline{y})$ that can be obtained by deviating γ units of flow. Then, the cost of each of these neighbors is computed as follows: For each arc $(i,j)^+ \in \gamma$-residual network, we find an approximation c_{ij}^+ of the increase in cost from routing γ units of additional flow on this arc:

$$c_{ij}^+ = \begin{cases} \frac{\sum_{p \in P} c_{ij}^p}{|P|}\gamma + f_{ij} & \text{if } \sum_{p \in P} x_{ij}^p = 0; \\ \frac{\sum_{p \in P} c_{ij}^p}{|P|}\gamma & \text{if } \sum_{p \in P} x_{ij}^p > 0; \end{cases}$$

Similarly, for each arc $(j,i)^- \in \gamma$-residual network, we compute an approximation c_{ij}^- of the decrease in cost from the reduction of γ units of flow on this arc:

$$c_{ji}^- = \begin{cases} -\frac{\sum_{p \in P} c_{ij}^p x_{ij}^p}{\sum_{p \in P} x_{ij}^p}\gamma - f_{ij} & \text{if } \sum_{p \in P} x_{ij}^p = \gamma; \\ -\frac{\sum_{p \in P} c_{ij}^p x_{ij}^p}{\sum_{p \in P} x_{ij}^p}\gamma & \text{if } \sum_{p \in P} x_{ij}^p > \gamma; \end{cases}$$

The cost of a neighbor solution is computed by adding the cost of a flow deviation to the cost of the current solution. Figure 3.1 (c) is a 2-residual network with the changes on costs after deviating γ units of flow on each arc on the network $\overline{y}$ shown in Figure 3.1 (a).

Next, the move in the neighborhood is computed by identifying the cycle that maximize the cost reduction or minimize the cost increase. Let $\mathcal{C}(\gamma)$ be the set of arcs in the current solution $\overline{y}$ that can support the movement of γ units of flow. That means, for

open arcs, the flow in the current solution is at least γ, and for closed arcs, the residual capacity is at least γ. In Figure 3.1 (d), for $\gamma = 4$, if we assume that C is a set of all the arcs in the network, $C(4)$ includes arcs (E, D), (D, A), (A, B) and (B, C). The procedure to identify the best neighbor is the following one:

1. For each $\gamma \in \Gamma(\overline{y})$
   (a) For each arc $(i, j) \in \mathcal{C}(\gamma)$
      i. Find the cycle with the lowest cost that include arc (i, j).

This procedure considers each arc $(i, j) \in \mathcal{C}(\gamma)$ as the beginning of a cycle. For each arc (i, j), the procedure looks at all the cycles that contain arc (i, j) and identifies the cycle with the lowest cost.

3.2 Application of cycle-based tabu search to CMND

The cycle-based tabu algorithm that will be used in this thesis is illustrated in Algorithm 4. First, an initial solution is generated by opening all the arcs and solving the corresponding capacitated multicommodity minimum cost flow (CMCF) problem. This initial solution is then assigned to be the $CurrentSolution$ and $BestSolution$.

The tabu search procedure with cycle-based neighborhood is implemented through a while loop with a stopping criterion that is either a pre-defined number of iterations or a time limit. At each iteration in the loop, there are flow deviation, restoration and intensification phases.

Algorithm 1 Cycle-based Tabu Search Algorithm

// Initialization

Generate the initial solution and assign it to $BestSolution$ and $CurrentSolution$;

while (a stopping criterion is not satisfied)

1. Determine $\Gamma(\overline{y})$ and C for the current solution $\overline{y}$;
2. **for** each $\gamma \in \Gamma(\overline{y})$

   Install the γ_residual network;

   Determine $C(\gamma)$;

   for each arc $(i,j) \in C(\gamma)$

   Find the associated "low cost" cycle by using the network optimization procedure;

   Find the lowest cost cycle over all $(i,j) \in C(\gamma), \gamma \in \Gamma(\overline{y})$;
3. Move to the new solution by opening or closing corresponding arcs;
4. Evaluate the solution value of the new solution by solving the associated CMCF problem using CPLEX;
5. Trim, i.e., close opened arcs with no flow;
6. Set the tabu status to each complemented arc;
7. **if** the solution is not feasible

   Execute a $restoration$ phase;
8. Perform an $intensification$ phase if current solution satisfies:

   $\frac{CurrentSolution - BestSolution}{BestSolution} \leq IntensGap$;
9. **if** $currentSolution < BestSolution$

   Update $BestSolution$;