A First Multilevel Cooperative Algorithm for Capacitated Multicommodity Network Design

Teodor Gabriel Crainic
Départment management et technologie
École des sciences de la gestion
Université du Québec à montréal
and
Centre de recherche sur les transports
Université de Montréal
theo@crt.umontreal.ca

Ye Li
Department of Computer Science
University of Manitoba
yeli@cs.umanitoba.ca

Michel Toulouse
Department of Computer Science
University of Manitoba
toulouse@cs.umanitoba.ca

May 2005

Abstract

We describe the first multilevel cooperative tabu search for the capacitated multicommodity network design problem. Main design challenges are associated to the specification of the problem instance addressed at each level in cooperation, as well as to the definition of the cooperation operators. The paper proposes a first approach to address these challenges and tests it on a set of well-known benchmark problems. The proposed method appears competitive, particularly when difficult problems with many commodities are considered. Directions and challenges for future research are identified and discussed.

Key words: Parallel search, multilevel cooperation, cycle-based tabu search, capacitated multicommodity network design

Résumé

Cet article décrit le premier algorithme de recherche tabou parallèle coopérative multi-niveaux pour le problème de synthèse de réseaux multi-produits avec coûts fixes et capacités. La spécification des problèmes traités par chaque programme dans la coopération, ainsi que la définition des opérateurs de coopération comptent parmi les défis les plus sérieux associés à la conception d'un tel algorithme. Nous proposons une première approche, que nous testons à l'aide d'un ensemble de problèmes présent dans la littérature. La méthode proposée offre des résultats intéressants, surtout pour les problèmes, plus difficiles, avec un nombre élevé de produits. Des directions et défis de recherche sont également identifiés et discutés.

Mots clés : Recherches parallèles, coopération multi-niveaux, recherche avec tabou par des cycles, synthèse de réseaux multi-produits avec coûts fixes et capacités.

1 Introduction

The capacitated multicommodity network design (CMND) is a well-known NP-hard problem (Magnanti and Wong 1984; Minoux 1989). Exact solution methods are therefore generally excluded when investigating practical approaches to address most CMND problem instances and heuristics are often the methodology of choice (Balakrishnan, Magnanti, and Mirchandani 1997; Crainic 2000; Gendron, Crainic, and Frangioni 1998; etc.).

Parallel computing offers the possibility to design procedures that search more efficiently the solution space of complex problems. This may be achieved through the acceleration of some tedious computational phases of the algorithm, the decomposition of the problem domain or search space, or the design of a multi-thread parallel meta-heuristic where several exact or heuristic methods are applied concurrently to a given problem instance with various degrees of synchronization and information exchanges. Cooperative multi-thread meta-heuristic strategies generally offer the best performance when compared to sequential methods and other parallelization strategies. They appear to consistently identify better solutions over diverse sets of problem instances without excessive calibration efforts. Details on parallel meta-heuristic strategies may be found in the reviews of Crainic (2005), Crainic and Toulouse (1998, 2003), Cung *et al.* (2002), Holmqvist, Migdalas, and Pardalos (1997), Pardalos *et al.* (1995), and Verhoeven and Aarts (1995).

The design of the information exchange mechanism represents a significant challenge for the development of efficient cooperative multi-thread strategies (Toulouse, Crainic, and Gendreau 1996; Toulouse, Crainic, and Sansó 2004; Toulouse *et al.* 1998; Toulouse, Crainic, and Thulasiraman 2000). The *multilevel cooperative search* mechanism is based on the principles of local interactions among cooperating searches and controlled diffusion of information within the framework of a multilevel algorithm. The mechanism has been recently introduced and proved very successful for graph and hypergraph partitioning problems (Toulouse, Thulasiraman, and Glover 1999; Toulouse, Glover, and Thulasiraman 1998; Ouyang *et al.* 2000, 2002).

The development of a multilevel cooperative search algorithm for the capacitated multicommodity network design problem must address several challenging issues, in particular those related to the specification of the problem instance solved at each level and the definition of the cooperation operators. The goal of this paper is to explore these issues and propose an approach to address them.

The paper makes several contributions. It proposes the first multilevel cooperative search algorithm for the capacitated multicommodity network design problem. Exper-

imental results on a set of well-known benchmark problems show that the method is competitive, particularly when difficult problems with many commodities are considered. The paper also presents a rather detailed analysis of the main design elements of the algorithm, providing guidelines for future developments. Finally, the paper presents for the first time a structured and comprehensive template for multilevel cooperative search methods.

The paper is structured in the following manner. In Section 2, we recall the mathematical formulation of the CMND problem and briefly survey the literature. Section 3 presents the multilevel cooperative search template. Section 4 is dedicated to the multilevel method for the CMND problem. We identify and discuss the main design issues and describe a specific implementation. Section 5 reports computational results and presents a first analysis of the behavior of the multilevel cooperative method applied to the CMND problem. A number of challenging research directions are discussed in Sectio 6. We conclude in Section 7.

2 Capacitated, Multicommodity Network Design

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with sets of nodes $\mathcal{N}$ and directed arcs $\mathcal{A}$. Without loss of generality, we assume that all $(i,j) \in \mathcal{A}$ are design arcs. Denote f_{ij} and u_{ij}, the fixed cost and the capacity of arc (i,j), respectively. Let $\mathcal{P}$ denote the set of commodities. For each commodity p, one must move w^p units of flow from its (unique) origin $o(p)$ to its (unique) destination $s(p)$. Denote c_{ij}^p the (variable) cost of moving one unit flow of commodity p on arc (i,j).

Two sets of decision variables are defined. *Design* variables y_{ij}, $(i,j) \in \mathcal{A}$, that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), and *distribution* variables x_{ij}^p indicating the amount of flow of commodity $p \in \mathcal{P}$ on arc (i,j). The arc-based formulation of the CMND can then be written as

$$\text{Minimize} \quad Z(x,y) = \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{p\in\mathcal{P}}\sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}^p \tag{1}$$

$$\text{subject to} \sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = \begin{cases} w^p & \text{if } i = o(p) \\ -w^p & \text{if } i = s(p) \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in \mathcal{N}, \forall p \in \mathcal{P}, \tag{2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij}y_{ij} \quad \forall (i,j) \in \mathcal{A}, \tag{3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}, \tag{5}$$

where $\mathcal{N}^+(i)$ and $\mathcal{N}^-(i)$ represent the sets of successor and predecessor nodes of node i, respectively. The objective function (1) accounts for the total system cost computed as sum of the fixed costs of the arcs included in the design plus the cost of routing the product demand on the resulting network. Constraints (2) represent the network flow conservation relations. The linking constraints (3) state that the total flow (all commodities) on an open arc ($y_{ij} = 1$) cannot exceed its capacity, while it must be 0 if the arc is closed ($y_{ij} = 0$). Relations (5) and (4) are the usual non-negativity and integrality constraints for decision variables.

The CMND problem is hard and, for now, exact methods cannot solve realistically-dimensioned instances (Crainic, Frangioni, and Gendron 2001; Gendron, Crainic, and Frangioni 1998; Holmberg and Yuan 2000), even though interesting developments are being proposed (Kim *et al.* 1999; Armacost, Barnhart, and Ware 2002; Chouman, Crainic, and Gendron 2003). For capacitated problems of any interesting size, only specially tailored heuristics have proved of any help. The simplest of these are drop and add procedures based on reduced cost calculations that determine the marginal value of including or excluding an arc from the network (Powell 1986, Koskosidis, Powell, and Solomon 1992), while more sophisticated approaches make use of the marginal values of paths (Crainic and Rousseau 1986, Farvolden and Powell 1994, Jarvis and Mejia de Martinez 1977). These heuristics were applied to particular problem classes, however, and did not attempt to explore past the first local optimum.

Meta-heuristics based on exploring very large neighborhoods have recently been proposed. Crainic, Gendreau, and Farvolden (2000) proposed a tabu search meta-heuristic for the path-based formulation of the problem that explores the space of the path-flow variables by using pivot-like moves and column generation. Very good results were reported. Crainic and Gendreau (2002) presented a cooperative multi-search method, where several threads of the path-based tabu search, using different parameter settings, exchanged solutions through a central memory mechanism. The cooperative multi-search displayed improved performance compared to the sequential search.

The cycle-based neighborhood structures for the CMND proposed by Ghamlouche, Crainic, and Gendreau (2003) explore the space of the arc design variables by re-directing flow around cycles and closing and opening design arcs accordingly. These neighborhoods are part of what appears to be the current best meta-heuristic for the CMND, the path relinking procedure proposed by Ghamlouche, Crainic, and Gendreau (2004). The Lagrangian relaxation-based slope scaling heuristic proposed by Crainic, Gendron, and Hernu (2004) opens interesting perspectives but does not alter the previous statement. Cycle-based neighborhoods are therefore used in the cooperative method we propose as described in Section 4.3.

3 A Multilevel Cooperative Search Template

The multilevel cooperative paradigm for the design of parallel meta-heuristics has been introduced quite recently. Consequently, only a limited number of contributions on this topic may be found in the literature. Moreover, none of these papers presents a comprehensive description of the multilevel cooperative mechanism. This section aims to fill this gap and describes the multilevel cooperative search template.

3.1 Cooperative Parallel Meta-Heuristics

Cooperative multi-thread search methods are parallel meta-heuristics built upon independent search threads that exchange information. There are essentially two basic approaches to sharing information among threads, through either *global* or *local interactions*.

Global interactions rely on explicit or implicit representations of the state of the global search to identify and coordinate the sharing of information. Thus, for example, threads may stop at a number of global synchronization points where best solutions are exchanged (generally, not a very successful strategy; Crainic and Toulouse 2003). Alternatively, some central depository is used where information, including but not restricted to best solutions, is stored and, possibly, processed, and from where it may be fetched at any time by one of the cooperating search threads. The *adaptive* (Rochat and Taillard 1995) and *central* (Crainic, Toulouse, and Gendreau 1997; Crainic and Gendreau 2002) memory strategies belong to this class of approaches and have been successful in addressing a number of difficult problems (e.g., the surveys of Crainic and Toulouse 2003 and Crainic 2005).

Cooperative search procedures based on local interactions are simpler to implement. No global point of reference is used by such methods, neither explicit ones such as synchronization schemes or global control by "higher" algorithms, nor implicit such as central repositories for exchanged information. In fact, usually, exchanges of information based on local interactions take place between two search threads: a sender and a receiver. In its simplest form, the exchange consists for the former to send a solution that the receiver uses to restart its search.

It is important to notice, however, that other than the formal exchange of information taking place between two locally interacting threads, a broader self-organizing (self-regulating) system of information exchanges is taking place in cooperative multi-thread searches based on local interactions. This system supports the diffusion of information among many threads through long chains of correlated local interactions

and feedback loops (Toulouse, Crainic, and Thulasiraman 2000; Toulouse, Crainic, and Sansó 2004). This diffusion process impacts in complex ways the exploration of the solution space performed by cooperating search threads and appropriate mechanisms must be devised to control its behavior.

The study of Toulouse, Crainic, and Thulasiraman (2000) showed, in particular, that the same values for subsets (blocks) of decision variables are propagated across several search threads through chains of correlated local interactions. To illustrate, consider a thread B that has already interacted with thread A and interacts now with thread C. Assuming threads A and B are senders, then the two local interactions overlap because they share a common thread: B. Two local interactions $A-B$ and $B-C$ are *correlated* if interaction $B-C$ occurs because of the occurrence of the local interaction $A-B$. In this case, *blocks* of information containing the state of some decision variables are sent unchanged from thread A to thread C using thread B. The Toulouse, Crainic, and Thulasiraman (2000) study showed that such duplication of information is a form of spontaneous global control structure that coordinates the exploration of the search space performed by cooperating search threads. Unfortunately, most cooperative search algorithms are not designed to monitor the blocks that get copied through correlated interactions. Consequently, when "low-quality" blocks get widely propagated, the overall search focuses on poor regions of the solution space and its performance declines.

The *multilevel cooperative search* method is a paradigm to design cooperative multi-thread algorithms based on local interactions (Toulouse, Thulasiraman, and Glover 1999; Ouyang *et al.* 2002). It aims to increase the control on the complex information diffusion process generated by chains of correlated local interactions and emulate the spontaneous behavior observed in cooperative search procedures (Toulouse, Crainic, and Thulasiraman 2000). The method addresses the previously-identified issues by making explicit the process by which blocks of information containing the state of decision variables are formed and exchanged by a given cooperative parallel algorithm. The same blocks are also used to define moves and to decompose the search space to allow parallel explorations. The next two subsections detail these ideas.

3.2 Moves, Neighborhoods, and Levels

Consider the set of solutions that may be searched by a given meta-heuristic for a particular problem. Define a *block* as a set of decision variables with the same *status* relative to the current solution (e.g., value in the solution vector) or search history (e.g., value in search memory vectors). A decision variable may belong to at most

one block. A block-building method, aggregation, variable fixing, etc., is applied to these variables fixing them for as long as the block is not modified or dismantled. The *coarsening factor* specifies the number of decision variables per block. We may then define a *neighborhood* as the set of solutions that may be reached from the current one by a move that modifies all variables in a block.

To illustrate, consider the graph partitioning problem. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, $\mathcal{P} = \{1, 2, \ldots, p\}$ the set of p partitions of $\mathcal{G}$. Define x_{ij} the decision variable that takes value 1 if vertex $i \in \mathcal{V}$ belongs to partition $j \in \mathcal{P}$, and 0 otherwise, and let $\mathcal{X}$ be the solution space. Consider a procedure that builds blocks with a coarsening factor of 2: "For a feasible solution $x \in \mathcal{X}$, aggregate pairs of vertices $u \neq v \in \mathcal{V}$ such that u, v are in the same partition $j \in \mathcal{P}$". We may then define a neighborhood as the set of solutions that may be reached from the current one by transferring all variables in a block to a different partition. A different neighborhood may be defined by exchanging (swapping) blocks between two different partitions.

More than two decision variables may make up a block and many aggregation rules are possible (e.g., Toulouse, Thulasiraman, and Glover 1999; Ouyang *et al.* 2000, 2002 for the graph and hypergraph partitioning problems). Each strategy used to define blocks induces a different neighborhood and, thus, a different search through the solution space of the given problem. These different neighborhood structures may then be viewed as a source of parallelism.

Multilevel cooperative meta-heuristics are parallel algorithms that use this source of parallelism. The name *multilevel* is derived from the technique used to differentiate neighborhoods, motivated by the goal of controlling the diffusion of information. The multilevel cooperation mechanism is thus defined by a hierarchical neighborhood structure for the global, cooperative search, and by a number of operators specifying the local interactions and information exchanges within the hierarchy. These operators are introduced in the next subsection.

The multilevel hierarchy is obtained by specifying a particular block size for the neighborhoods used at each level. No aggregation is used at level 0, the block size thus corresponding to a single decision variable. The number of variables in a block increases with each level, the highest level corresponding to blocks with the largest size. The number of levels and the coarsening factor are application-specific parameters.

3.3 Local Interactions and Cooperative Search

Local interactions in a multilevel cooperative search procedure take place only between adjacent levels in the hierarchical neighborhood structure. Interactions thus always consist in information being sent by a *sender* search process to a *receiver* search thread that is exactly one level up or down in the hierarchy. This limits the diffusion of information and simplifies to a great extend how information propagates among the cooperating search threads. Two main types of local interactions are defined and implemented through specific operators.

The first type of interaction does not modify the current blocks of the receiver thread, but it modifies (replaces in the extreme case) the solution from which the search will continue. Assuming the receiver thread is at level i, an *interpolation* operator is associated to information that arrives from a sender *above*, at level $i+1$, while a *reverse interpolation* operator is used when information arrives from the level *below* $(i-1)$. The solutions sent from level $i+1$ or $i-1$ are selected among a set of *elite solutions*, defined as the best solutions ever visited by the search thread at the corresponding level. Once a solution is received, the receiver search thread, at level i, modifies its current solution and thus continues its search in a region of the solution space selected strongly based on the solution received from its neighbor.

Notice that, because levels use different neighborhood structures, the search at the receiver level will proceed along a different trajectory from that followed by the sender even when the receiver completely replaces its current solution with the new one and restarts the search. In particular, the neighborhood structure at level i has more degrees of freedom compared to that of level $i+1$: blocks are smaller and solution neighborhoods are larger. Also notice that restarting using an elite solution from the level above is reminiscent of initiating an *intensification* search in the region identified by the elite solution. On the other hand, the neighborhood structure at level i has less degrees of freedom compared to level $i-1$: blocks are coarser and neighborhoods are smaller. A restart using an elite solution from the level below may then be viewed as a somewhat limited diversification of the search from the region identified by the received elite solution.

The second category of local interactions controls the generation of new blocks of decision variables. These local interactions use information from level $i-1$ to change the blocks at level i. Two local interaction operators were defined for changing blocks in the graph partitioning applications: a *partitioning* operator and a *re-coarsening* operator. The former destroys existing blocks, while the latter creates new ones. By changing blocks at a given level, some regions of the solution space become unreachable for the search thread at that level, while new ones become available. New blocks yield new neighborhoods and search threads may then explore new regions of the

solution space. Such interactions may therefore be considered major *diversification* moves.

Memories collecting information relative to the history of the search at each level may be used to guide the utilization of the interaction operators. Thus, in the graph partitioning applications, a *frequency* vector at level $i-1$ governed the application of the two block-modifying operators at level i. The frequency vectors registered the state of the decision variables among the best solutions visited (the set of elite solutions) by the search thread. Decision variables that appeared with the same values in several elite solutions were considered "settled". In terms of the search method at level i, settled variables at level $i-1$ can be handled as if they were a single variable. Therefore, decision variables at level $i-1$ with similar frequency values form new blocks at level i. Similarly, blocks at level i are dismantled if they are composed of two or more blocks from level $i-1$ with frequency values that are substantially different from one another.

Graph and hypergraph partitioning problems have been the object of intensive research for quite some time now (e.g., Kernighan and Lin 1970) and several multilevel algorithms have been proposed (e.g., Hendrickson and Leland 1993; Cong and Shinnert 2003). Yet, the application of the multilevel cooperative template, combined to a good local search heuristic (Fiduccia and Matteyses 1982) applied at each level, has produced superior results (Ouyang *et al.* 2002). We aim to reproduce this success for the CMND problem.

4 Multilevel Algorithm for the CMND Problem

This section is dedicated to the presentation of the multilevel cooperative algorithm proposed for the CMND problem. We start with the definition of levels and the specification of the initial values. The local interaction operators are defined next, followed by the description of the cycle-based tabu search used to explore each level. The algorithm is summed up in the last subsection.

4.1 Level Definition

The first issue one must address when contemplating the development of a multilevel cooperative search algorithm for the CMND problem is how to define levels. Recall that the main idea is to define levels by aggregating decision variables. On the other hand, one has to preserve the characteristics of the initial problem to ensure the

correctness of the diffused information.

It thus become clear quite rapidly that a straightforward aggregation of the nodes (and arcs) of the network, similar to that practiced for the graph partitioning problem, is not appropriate for the CMND. On the one hand, node aggregation makes the definition of the resulting link and, especially, node fixed costs and capacities awkward at best. On the other hand, while responding to the goal of the graph partitioning problem, graph aggregation does not relate to the objectives of a design process: deciding which arcs to select and which ones to discard.

We therefore define levels by the number of arcs that are *fixed*, each fixed arc being either *fixed-open* or *fixed-closed*. An arc that is fixed cannot have its status changed by the tabu search process that proceeds at the corresponding level. In fact, a fixed-closed arc "disappears" from the graph, while a fixed-open arc "looses" its fixed cost (appropriate accounting has to be enforced, of course) and cannot be closed. Only interactions with neighboring levels may yield a modification in the fixed status of arcs. Thus, the fixed status of an arc reduces the dimension of the design decision variable vectors and consequently the number of neighbors in the corresponding search space. The specification for each level of the fixed-open and fixed-closed arcs thus directly impacts the exploration of the solution space performed by the tabu search program.

To create the initial set of levels, one needs to fix the size of this set, the number of levels, as well as the coarsening factors for each level specifying the number of arcs that are fixed-open and fixed-closed at each level. One also needs a procedure to select which arcs become fixed-open and fixed-closed at each level. The number of levels and the coarsening factors are parameters with values obtained through calibration (Section 5).

The strategy to select the initial arcs in fixed-open and fixed-closed blocks should aim to select arcs that are likely to have the the status "open" or "closed" in good network designs. We considered two methods to estimate this likelihood. The first makes use of the FC ratio defined as $\frac{fixed\ cost}{capacity}$. A small FC ratio may be interpreted as a signal that the corresponding arc might be open in good designs since it brings capacity to the network with a relatively limited increase in fixed cost. Symmetrically, a large FC ratio is taken to indicate that the corresponding arc may not appear in good designs.

The second heuristic consists in selecting blocks of decision variables based on an initial exploration of the solution space. In this case, the multilevel cooperative search starts as a parallel independent search made up of l independent sequential cycle-based tabu search procedures, where l is the number of levels in the multilevel

procedure. Each independent search uses the same initial solution (all arcs are open). Because the cycle-based tabu search selects neighbors randomly, the independent searches eventually diverge in their exploration of the solution space.

To each level i is associated a set of *elite solutions* as well as a *frequency memory*. The elite set contains the best solutions found at level i. The size of this set is a pre-defined parameter. The frequency memory is an integer vector V of size equal to the number of decision variables. Each entry $V[a]$ records the number of times the corresponding decision variable $y_a,\ a = (i,j),\ i,j \in \mathcal{A}$, had the status open in solutions that either belonged to the elite set or had objective values close (less than a given threshold) to those of the worst solutions in the elite set.

The initial blocks of fixed-open and fixed-closed arcs for each level i are created once the cycle-based tabu search has failed to improve the best known solution for a pre-defined number of iterations. At that point, the coarsening factors pre-selected for level i are used to determine the size of each block. The decision variables with the highest frequency values in the frequency memory become fixed-open while those with the lowest frequency values become fixed-closed.

Once the blocks are generated at all levels, the search proceeds according to the multilevel cooperative idea: at each level the cycle-based tabu search attempts to improve the current solution given the blocks of fixed variables, while local interactions are executed by each level with its neighbor levels above and below in the multilevel structure. Search threads proceed asynchronously. At each level, the information required for cooperating with neighbor levels, the elite set and the frequency memory, is available at all time, therefore a level does not have to wait for the neighbor levels to initiate cooperation.

4.2 The Local Interaction Operators

Several local interaction operators have been considered to support cooperation among levels. Three have been retained: *interpolation*, *re-coarsening*, and *reverse interpolation*. Note that the two interpolation operators, the reverse interpolation one in particular, indirectly destroy blocks.

The *interpolation operator* is used to replace the current solution of the sequential cycle-based tabu search at level i based on an elite solution from level $i+1$ and start the search in the new region identified by the imported solution. The interpolation operation is initiated from level i by requesting one of the solutions in the elite set of level $i+1$ that has not already been sent to level i. The open (closed) status of

some of the decision variables in the imported solution from level $i+1$ may conflict with the status of some of the fixed decision variables at level i, however. Thus, for example, some arcs may be part of the network design at level $i+1$, while being fixed-closed at level i. In order to restart the search at level i using the imported elite solution from level $i+1$, the fixed-closed status is canceled for all the arcs that conflict with the imported solution. This makes sure that the search in level i will proceed freely in the region identified by the imported elite solution. Indeed, removing the fixed status of arcs at a given level has the effect of a block-destroying operator in the sense that it increases the number of neighbors of solutions at level i. Denote Y_i the vector $Y_i[a]$ that gives the status of arc a at level i. If $Y_i[a] = 1$, then arc a is part of the fixed-open variables. If $Y_i[a] = 0$, a is part of the fixed-closed variables. If $Y_i[a] = 2$, arc a is a design variable. The pseudo-code of the interpolation operator is given in Figure 1.

```
Interpolation(Y_i);
   Get an elite solution ȳ_{i+1} from level i + 1
   for  (a = 1; a ≤ n; a++)
      if arc ȳ_{i+1}[a] is opened then Y_i[a] = 2;
   return  Y_i;
   Restart search at level i with ȳ_{i+1} as initial solution;
```

Figure 1: Interpolation Operator

The *reverse interpolation* operator is similar to the interpolation operator, except that it propagates information up, from lower to higher levels. It uses the same elite set as the interpolation operator. At regular intervals, level i requests from level $i-1$ the best elite solution that has not yet been sent to level i. Upon receiving this elite solution, the fixed-closed status is removed at level i for all arcs that conflict with their status in the imported elite solution. Similar to the interpolation operator, reverse interpolation potentially reduces the size of the fixed-closed block, which increases the density of the neighborhood structure at level i. Figure 2 describes this operator.

The *re-coarsening operator* proceeds as follows. At regular intervals, i.e., after a predetermined number of iterations of the cycle-based tabu search method, level $i = 1, \ldots, l-1$ makes a request to obtain a copy of the frequency memory V_{i-1} from level $i-1$. The blocks of fixed-open and fixed-closed decision variables at level i are then modified based on the frequency memory V_{i-1}. Decision variables with a high frequency in V_{i-1} have their status changed to fixed-open while those with low frequency have their status changed to fixed-closed. The re-coarsening operator is also used to bring the number of arcs with fixed status in line with the figures associated to the coarsening factors for level i.

```
Reverse_Interpolation(Y_i);
  Get an elite solution ȳ_{i-1} from level i − 1
  for (a = 1; a ≤ n; a++)
    if arc ȳ_{i-1}[a] is opened then Y_i[a] = 2
  return Y_i;
  Restart search at level i with ȳ_{i-1} as initial solution;
```

Figure 2: Reverse Interpolation Operator

The re-coarsening operator is described in Figure 3. The input variables cfo_i and cfc_i represent the block sizes of fixed-open and fixed-closed variables, respectively, according to the coarsening factors for level i. The frequency memory V_{i-1} is scanned to sort decision variables in decreasing order of their frequency values (the **for** loop of line 1). $B[k] = p$ indicates that decision variable p has the k^{th} highest frequency. The vector Y_i is then updated according to the frequency memory V_{i-1} (**for** loops of lines 2 and 3).

4.3 Cycle-based Tabu Search

The search space of each level is explored by a cycle-based tabu search (Ghamlouche, Crainic, and Gendreau 2003). The basic neighborhood defines moves that may explicitly take into account the impact on the total design cost of potential modifications to the flow distribution of several arcs and commodities simultaneously. The fundamental idea is that one may move from one solution to another by 1) Identifying two points in the network together with two paths connecting these points, thus closing a cycle; 2) Deviating the flow from one path to another such that at least one currently open arc becomes empty; 3) Closing all previously open arcs in the cycle that are empty following the flow deviation and, symmetrically, opening all previously closed arcs that now have flow.

Cycle-based neighborhoods are huge and an optimization-based implicit exploration method is used. The goal is to identify moves that lead to a significant modification of the flow distribution, such as moves that close at least one arc and open new paths for a group of commodities. To close an arc (i, j), one must be able to deviate all its flow. Let the *residual capacity* of a cycle denote the maximum flow one can deviate around the cycle. The residual capacity of any cycle that includes (i, j) must then be at least equal to the total flow on arc (i, j). Consequently, cycles of interest are those that display a residual capacity equal to one of the values in the

```
Re-coarsening(Y_i, cfo_i, cfc_i);
  int  A, C, B[1..n];
  get the frequency vector V_{i-1} from level i − 1
1. for  (k = 1; k ≤ n; k++)
     A = 0;
     for  (p = 1; p ≤ n; p++)
       if  ((V_{i-1}[p] ≥ A)&(V_{i-1}[p] ≠ ∞)) then
         A = V_{i-1}[p];
         C = p;
     B[k] = C; V_{i-1}[C] = ∞;
2. for  (k = 1 to cfc_i)
     Y_{i+1}[B[k]] = 1;
3. for  (k = n − 1 down to cfo_i)
     Y_i[B[k]] = 0;
  return  Y_i;
```

Figure 3: The Re-coarsening Operator

set of the total (strictly positive) volumes on the open arcs.

Let $(\bar{y}, x^*(\bar{y})$ represent the current solution, where the design decision $\bar{y}$ assigns a value of 0 or 1 to each design variable, while $x^*(\bar{y})$ stands for the optimal flow of the corresponding minimum cost capacitated multicommodity network flow problem (CMCF). Cycles are then identified on *residual networks* of $\mathcal{A}(\bar{y})$ defined according to $\Gamma(\bar{y}) = \left\{\sum_{p\in\mathcal{P}} x_{ij}^{p*} > 0 \ : \ (i,j) \in \mathcal{A}(\bar{y})\right\}$, where $\mathcal{A}(\bar{y})$ stands for the set of arcs included in the design $\bar{y}$. The best move in the neighborhood of $\bar{y}$ is then identified by the cycle leading to the network modification that yields the largest improvement (smallest deterioration, eventually) in the objective function (1). The "best" cycle for each candidate link is identified by an optimization heuristic based on a modification of the shortest path label-correcting algorithm that avoids getting trapped in negative directed cycles.

To reduce the computational burden, cycles are identified and evaluated for a set of *candidate* links $\mathcal{C} \subseteq \mathcal{A}$. Let $\mathcal{C}(\gamma) \subseteq \mathcal{C}$ include all arcs $(i,j) \in \mathcal{C}$ that can support a movement of γ units of flow. A closed arc may belong to $\mathcal{C}(\gamma)$ only if its capacity is at least γ, while for an open arc, either its flow or its residual capacity must be at least γ units. Intensification moves aim to refine the search by implementing moves that result from the deviation of the flow of one commodity at a time. The corresponding neighborhoods are obtained by by instantiating the set Γ for each commodity, denoted

$\Gamma^p = \left\{ x_{ij}^p > 0 \ : \ (i,j) \in \mathcal{A}(\bar{y}) \right\}$.

A more comprehensive description and computational analysis of cycle-based neighborhoods and associated meta-heuristics may be found in Ghamlouche, Crainic and Gendreau (2003, 2004). Figure 4 illustrates the cycle-based tabu search used at each level i, while Figure 5 displays the sequence of tabu search and interaction operators (Section 4.4). The tabu search procedure thus accepts as input the best solution and its value *BestSolution*, as well as the current solution, and its value *CurrentSolution*, obtained by the application of an interaction operator (including any mending required to obtain a feasible solution - see Ghamlouche, Crainic and Gendreau 2003). The current *EliteSet*, the vector $Y_i[j]$, giving the fixed status of the design arcs, and the frequency memory V_i are also input. To simplify the representation, the level index i is generally not used in Figure 4.

The network is modified initially according to the fixed status of the design arcs. Then, at each iteration, the best non-tabu move in the neighborhood is determined and implemented, whether it improves the overall solution or not. A short-term tabu memory is used to record characteristics of visited solutions to avoid cycling. Moves may generate infeasible solutions detected through the use of artificial arcs. A mending phase is then undertaken, by restricting the sets $\mathcal{C}$ and Γ to the artificial arcs with positive flow. When a particularly good solution is encountered, the search may be intensified using the intensification neighborhood and accepting improving moves only. A solution is considered particularly good when it either improves the best overall solution or is close to it by at least a pre-defined percentage. Intensification is performed at levels 0 and 1 only. At higher levels, the network is too aggregated for intensification to be meaningful. The last steps are dedicated to the eventual update of the elite set (worst solution is replaced), the memory vector (solution is within threshold of the worst solution in elite set), and the best solution.

4.4 The Multilevel Algorithm for the CMND Problem

Each level of the multilevel cooperative algorithm explores the solution space according to the neighborhood structure imposed by the two blocks of fixed decision variables. The iterations of the cycle-based tabu search procedures are interrupted to communicate with adjacent levels using one the three interaction operators. Interactions are local and take place asynchronously. All the tabu procedures are identical for all the levels except for levels 0 and 1 where intensification may be performed to explore more thoroughly the regions of the solution space identified as "promising" by higher levels. Our stopping criterion is a pre-defined number of cycle-based tabu search iterations, including the iterations executed to obtain the initial blocks

Initialization.

If $Y_i[j] = FixedClose$, then delete arc (i, j) from the network;

If $Y_i[j] = FixedOpen$, then delete the fixed cost of arc (i, j).

Main local search loop. While a stopping criterion is not met

1. Determine sets $\mathcal{C}$, $\Gamma(\bar{y})$, and $\mathcal{C}(\gamma)$ for the current solution $\bar{y}$;
2. Determine the best cycle over all $(i, j) \in \mathcal{C}(\gamma)$, $\gamma \in \Gamma(\bar{y})$

   For each $\gamma \in \Gamma(\bar{y})$

   Build the γ-residual network;

   For each arc $(i, j) \in \mathcal{C}(\gamma)$

   Find the lowest cost cycle using the network optimization procedure;

   Select the best move;
3. Move to the new solution by opening and closing the appropriate arcs;
4. Evaluate the new *CurrentSolution* by solving exactly the associated CMCF;
5. Assign a tabu status to each complemented arc;
6. Trim (close arcs with no flow);
7. If the solution is infeasible, perform a *Mending* phase;
8. If level i is 0 or 1, perform an *Intensification* phase if:

$$\frac{CurrentSolution - BestSolution}{BestSolution} \leq IntensGap;$$

9. If $CurrentSolution < WorstEliteSolution$ update the *EliteSet*;
10. If

$$\frac{CurrentSolution - WorstEliteSolution}{WorstEliteSolution} \leq EliteGap$$

update the frequency memory V_i;
11. If $CurrentSolution < BestSolution$ update *BestSolution*.

Figure 4: Tabu Search with Cycle-based Neighborhoods at Level i

of fixed-open and fixed-close arcs. The number of tabu iterations is the same for all levels.

Generate the initial sets of fixed-open and fixed-closed blocks;

while (stopping criterion not satisfied) **do**

1. Perform cycle-based tabu search iterations;
2. All levels $i = 0, \ldots, l-1$, perform *Interpolation*;
3. Perform cycle-based tabu search iterations;
4. All level $i = 1, \ldots, l$, perform *Re-coarsening*;
5. Perform cycle-based tabu search iterations;
6. All level $i = 1, \ldots, l$, perform *Reverse interpolation*;

Figure 5: Core of the Multilevel Cooperative Algorithm for the CMND Problem

The three interaction operators, interpolation, reverse interpolation, and re-coarsening, support cooperation among the cycle-based tabu search programs. Each level executes the sequence of steps describes in Figure 5. Steps 2, 4, and 6 state the sequence of calls to specific interaction operators. This particular sequence has been set using the following logic: The first interaction operator is a restart of the cycle-based tabu search program at level i, $i = 0, \ldots, l-1$ using an elite solution from level $i+1$. This usually takes place once the cycle-based tabu search has exhausted its search at level i. Relative to level $i+1$, the search that follows an interpolation is a form of intensification performed with the help of the less coarsened neighborhood structure of level i. Once the tabu search has explored the region of the solution space identified by the interpolation operator, Step 4 calls the re-coarsening operator. The re-coarsening operator permits to explore again the region identified by interpolation, but using a new neighborhood provided by the frequency memory of level $i-1$. Step 6 is also a restart, but using an elite solution from the lower level. Chances are, the imported elite solution from level $i-1$ is a local optimum or comes from the basin of attraction from which the search at level $i-1$ failed to escape. The restart of this exploration in the more coarsened neighborhood of level i should help to cross barriers toward promising regions. With reference to level $i-1$, the search that follows a reverse interpolation is a form of diversification performed with the help of a more coarsened neighborhood structure.

5 Computational Results

This section is dedicated to the presentation and analysis of the results of the experiments performed with the simple multilevel cooperative design presented in the previous section. This is the first study of a multilevel cooperative strategy applied to the CMND problem. It is also one of the very few studies so far of the multilevel cooperative parallel strategy for meta-heuristics. Hence, in analyzing the results, we focus both on the capability of the multilevel design to yield high quality solutions for the CMND problem and on the understanding of the multilevel cooperation mechanism. This explains the somewhat lengthier than usual discussion on the calibration results.

To ensure meaningful comparisons, we experimented on the same problem instances used by Ghamlouche, Crainic, and Gendreau (2003, 2004). Problem instances are general transshipment networks with no parallel arcs. Each commodity corresponds to a single origin-destination pair. On each arc, routing costs are the same for all commodities. Problem instances have been generated to offer for each network size a variety of fixed-to-routing-cost and capacity-to-demand ratios. A detailed description of problem instances is given in Crainic, Frangioni, and Gendron (2001; see also Gendron and Crainic 1994, 1996). The problem generators as well as the problem instances can be obtained from the authors.

The multilevel cooperative meta-heuristic has been coded in C++. The exact evaluations of the capacitated multicommodity network flow problems were performed using the linear programming solver of CPLEX version 7.5. The parameters used for the cycle-based tabu search are those indicated in Ghamlouche, Crainic, and Gendreau (2003). Tests were conducted on a Sun Enterprise 10000 with 64 400 MHz clock processors and 64 Gb of RAM. The parallel programming model for the multilevel procedure was shared memory using Pthreads under Solaris 2.7.

5.1 Parameter Calibration

Initial experiments were performed to determine values for a number of key parameters and design features. Calibration tests were conducted using eight problem instances with 20 nodes and 300 design arcs with different number of commodities (40 and 200) and various combinations of fixed-versus-variable-cost ratio and capacity tightness measures. Our previous work on developing meta-heuristics for design problems indicates that these problems offer a reasonable sample of characteristics and difficulty. The following parameters were studied using these problem instances: 1) Number of levels; 2) Coarsening factors, that is, the size of the fixed-open and

fixed-closed blocks at each level; 3) Number of tabu iterations between activations of the interaction operators; 4) Dimension of the elite solutions set; 5) Updating of the frequency memory; 6) Selection of initial arcs to be fixed.

The number of levels and the coarsening factors are closely related. On the one hand, a large number of levels increases the level of parallelism and, potentially, the performance of the global search. On the other hand, when coarsening factors increase substantially from one level to the next, the corresponding dimension of the solution space tends to decrease rapidly. Consequently, a high number of levels comes at the cost of low coarsening factor values, which yield small blocks and conduct to weak interactions and low-impact parallelism, a behavior one aims to avoid. A compromise between large coarsening factor values and a high number of levels must therefore be established.

This compromise is particularly difficult to find for most CMND problems of interest, where the planned capacity is relatively tight with respect to the total demand (problem instances with almost no slack capacity or, at the other extreme, with almost unlimited capacity are significantly easier to solve). For such problems, increasing the number of fixed-closed variables rapidly yields infeasible instances. Moreover, this characteristic is independent of problem dimensions, which contrasts to other combinatorial problems, such as graph partitioning, where increasing the number of decision variables generally permits to increase the number of levels and the values of the coarsening factors. We decided therefore to fix the number of levels for all problem instances based on the number of design decision variables in the smallest problem instances tested (100 arcs). Experimentally, eight levels appeared as the maximum level of parallelism we could achieve for those instances and this number was used for all calibration and experimentation activities.

To calibrate the coarsening factors, one must consider that fixed-open and fixed-closed arcs do not have the same impact on the exploration of the solution space. During the neighborhood exploration performed by the cycle-based tabu search, cycles that contain fixed-closed arcs are automatically eliminated. On the other hand, a cycle with fixed-open arcs may generate a move if this move does not involve changes to the status of the fixed-open arcs in the cycle. In other words, a fixed-closed status impacts in a more stringent manner the exploration of the solution space than the fixed-open status. Consequently, we considered smaller block sizes for fixed-closed arcs. We tested increasing the total number of arcs per level by 3%, 6%, and 9%, respectively, and coarsening factors with a ratio 1/3 for the fixed-closed and 2/3 for the fixed-open blocks, respectively. A coarsening factor of 9% per level, with 3% of the arcs fixed closed and 6% of the arcs fixed open, offered the best results for the calibration tests.

For this first multilevel cooperative algorithm for the CMND problem, we adopted the simple strategy of activating the local interaction operators after a pre-defined number of cycle-based tabu iterations. We tested the values 5, 10, 20, and 50. A value of 20 offered the best performance on the restricted calibration problem set and was used throughout the experimentation phase.

The calibration tests indicated that, in general, differences in the cardinality of the elite solution sets did not impact significantly the solution performance. It also appeared, however, that this cardinality should not be "too high" and that using elite sets that include several solutions dominates elite sets of size one. The rational seems to be the following. When single-elite-solution sets are used, the elite set would be updated significantly less frequently than when larger sets are used. Consequently, the interpolation and reverse interpolation operators would tend to send the same solution to neighbor levels. On the other hand, large elite sets might contain low-quality solutions which, when transmitted to neighboring levels, would send the exploration at those level astray. A cardinality of 4 for the elite sets appeared a reasonable compromise.

The goal of the frequency vectors is to accumulate information about the status of decision variables in good solutions. Similarly to the elite set cardinality calibration, a compromise must also be reached in this case. Restricting "good" solutions to those that are very close (less than 5%, say) to the overall best solution enforces the elite characteristic of the frequency vector, but yields procedures where applications of the re-coarsening operator have little impact on coarsening. On the other hand, the update of the frequency vectors based on "poor"-quality solutions implies driving the search (through the re-coarsening operator) into possibly uninteresting regions of the solution space. A maximum value of 5% less than the cost of the worst solution in the elite set at the corresponding level gave the best performance.

We tested also the two approaches described in Section 4.1 for determining the initial variables to be fixed. The approach based on the frequency memory appeared superior to that based on the FC ratio. In the frequency-memory approach, the initial coarsening takes place once the cycle-based tabu search has failed to improve the best known solution for a number of iterations. We tested values of 20, 50, and 100 for this parameter and a value of 100 performed best. We impose, however, a limit (fixed at 200) for the maximum number of iterations the search at each level may run in independent mode. Consequently, a level switches to cooperation mode either when it has failed to improve its best solution during the last 100 iterations or when it has reached the 200-iteration limit.

5.2 Result Analysis

Table 1 displays the results of the computational experiments. Problem instances are identified, in the first column, by a number plus their characterization in terms of the number of nodes, arcs, and commodities, as well as the fixed cost and capacity type: a relatively high or low fixed cost relative to the routing cost is signaled by the letter **F** or **V**, respectively, while letters **T** and **L** indicate respectively if the problem is tightly or somewhat loosely capacitated compared to the total demand. Column MLEVEL reports the results obtained for one run of the multilevel cooperative algorithm for 600 iterations.

The four next columns display comparisons to the best and the average solution out of three repetitions for the cycle-based tabu search executed for 400 iterations, columns TC and AV.TC respectively (Ghamlouche, Crainic, and Gendreau 2003), and the path relinking approach, columns PR and AV.PR, respectively (Ghamlouche, Crainic, and Gendreau 2004). To standardize the presentation, gaps have been computed relative to the corresponding value achieved by the multilevel cooperative algorithm. Consequently, positive entries indicate better performance in terms of solution quality obtained by the multilevel algorithm. The last line of entries displays the corresponding averages.

The results are very encouraging. Compared to the sequential cycle-based tabu search, the multilevel cooperative method obtained consistently higher-quality solutions, for 93% of the problem instances tested, for an average improvement gap of 2.38%. Moreover, the method is very competitive when compared to the path relinking procedure, the best current meta-heuristic for the CMND problem. The objective function values of the solutions identified by the multilevel method are often equal or better than those obtained by the path relinking, for an overall average gap improvement of 1%. Moreover, all other problem characteristics being kept constant, the multilevel appears to perform better when the number of commodities is increased (which, normally, increases the difficulty of the problem). Gains are even more impressive when comparisons to average results of the tabu search

Comparing computing times is a very difficult task, because various results have been obtained on different architectures at different time periods. In the following, we give a rough estimation. The tabu search and path relinking methods were each run three times for 400 iterations. The multilevel cooperative algorithm run, that is, each level run once for 600 iterations. Thus, the differences in the iteration definition between methods notwithstanding, the wall-clock time of the parallel method is a little over half the total computing time of the three runs of the sequential method.

To gain insight into the behavior multilevel cooperation mechanism applied to the

PROB	MLEVEL	TC	AV. TC	PR	AV. PR
C1(25,100,10,V,L)	14712	0	0.39	0	0
C2(25,100,10,F,L)	14941	0	0	0	0.94
C3(25,100,10,F,T)	49937	1.19	1.37	-0.08	0.43
C4(25,100,30,V,T)	365385	0	0	0	0
C5(25,100,30,F,L)	37607	-0.25	2.83	0.13	0.26
C6(25,100,30,F,T)	86461.3	1.00	1.89	-0.04	0.04
C7(100,400,10,V,L)	28553	0.82	0.99	-0.24	-0.08
C8(100,400,10,F,L)	24022	0	0	0	0
C9(100,400,10,F,T)	66284	1.36	1.65	-1.52	-1.12
C10(100,400,30,V,T)	385282	0.06	0.06	-0.09	-0.03
C11(100,400,30,F,L)	50456	2.73	3.41	1.72	2.81
C12(100,400,30,F,T)	145721	1.01	1.21	-2.99	-1.59
C33(20,230,40,V,L)	426702	0.92	0.95	-0.54	-0.45
C35(20,230,40,F,T)	652894	-0.02	0.75	-1.13	-1.00
C36(20,230,40,V,T)	371475	0.28	0.38	0.09	0.19
C37(20,230,200,V,L)	98582	1.44	2.94	1.85	2.93
C38(20,230,200,F,L)	143150	3.43	4.07	3.38	5.73
C39(20,230,200,V,T)	102030	4.74	5.45	2.61	3.50
C40(20,230,200,F,T)	141188	4.27	4.73	4.51	4.86
C41(20,300,40,V,L)	429837	0.51	0.54	-0.10	-0.07
C42(20,300,40,F,L)	593544	1.46	1.77	-0.53	0.18
C4320,300,40,V,T)	466004	0.02	0.12	-0.32	-0.25
C44(20,300,40,F,T)	619203	-0.61	-0.38	-1.49	-1.49
C45(20,300,200,V,L)	78209.5	4.04	5.09	-0.03	1.13
C46(20,300,200,F,L)	121951	0.26	1.06	1.28	2.44
C47(20,300,200,V,T)	77251	4.00	4.50	2.09	2.54
C48(20,300,200,F,T)	111173	2.50	3.75	2.17	3.11
C49(30,520,100,V,L)	55754	1.52	1.74	-1.53	-1.27
C50(30,520,100,F,L)	99817	3.85	5.47	2.24	4.41
C51(30,520,100,V,T)	53512	1.76	1.92	-0.93	-0.56
C52(30,520,100,F,T)	102477	2.59	5.28	3.57	4.98
C53(30,520,400,V,L)	115671	6.05	6.58	3.24	3.42
C54(30,520,400,F,L)	156601	4.81	5.66	4.16	4.33
C55(30,520,400,V,T)	120980	1.39	1.84	-0.67	-0.18
C56(30,520,400,F,T)	160217	5.80	6.29	2.16	2.94
C57(30,700,100,V,L)	48869	2.40	2.52	-0.30	0.05
C58(30,700,100,F,L)	63756	1.29	1.64	-1.04	-0.50
C59(30,700,100,V,T)	47457	1.52	1.67	-0.52	-0.36
C60(30,700,100,F,T)	56910	1.26	2.11	-0.59	-0.18
C61(30,700,400,V,L)	102631	4.97	5.68	2.42	2.70
C62(30,700,400,F,L)	143988	4.35	4.81	0.72	2.71
C63(30,700,400,V,T)	99194.9	2.58	3.95	2.03	2.95
C64(30,700,400,F,T)	138266	4.76	6.10	1.99	2.30
Average		2.38	3.25	1.00	1.15

Table 1: Computational Results: Relative Gaps (%) Relative to MLEVEL

CMND problem, we analyzed the search patterns of several problem instances, and compared them to those of an independent parallel method with the same number of threads (8 - threads were differentiated through the random exploration of the neighborhoods) and number of iterations (600). We may sum up our observations as follows (see Crainic, Toulouse, and Li 2004 for additional details and illustrations of search trajectories). All methods, sequential, parallel independent, and multilevel cooperative, behaved in essentially the same way during the early stages of the search, since the multilevel search proceeds as an independent parallel search during initialization. As expected, levels 0 and 1 of the two parallel methods displayed steeper descents due to the intensification phases that help find better solutions. The behavior of the two methods diverged once the cooperation mechanism become active.

The trajectory of the multilevel cooperative method displayed a marked drop in the best solution value following the iteration when the initial blocks of fixed decision variables are generated and inter-level cooperations are activated. This drop was noticeable at all levels, though more accentuated at levels 0, 1, and 2 where the search has more degrees of freedom (smaller blocks). The marked improvement brought by the initiation of cooperation was followed by a more gradual improvement in the best-solution value, punctuated by more significant movements following the interaction phases (in the current implementation, these occur at fixed intervals). This behavior has been observed for all problem instances and is a clear indication of the impact of the cooperation mechanism.

A significant difference was observed between the multilevel cooperative parallel search and the other methods. Each thread participating to the parallel independent search was a cycle-based tabu search. Consequently, the trajectories of each independent thread was similar to that of the sequential method and, as expected, the best solutions identified by these threads were approximately of the same quality and were similar to those found by the sequential method. The situation was different for the multilevel cooperative method, where the best solutions were found almost exclusively at the lowest levels of the hierarchy, levels 0 to 2, while levels 3 and 4 sometimes obtained solutions close to the best ones, and levels 6 and 7 contributed few very good solutions due to the large number of fixed arcs. This observation does not mean that higher levels are not useful. On the contrary, they contribute to diversify the search towards solutions which, once worked on by lower-level moves, will provide the good solutions. This characteristic of the multilevel cooperative search cannot be reproduced by parallel independent search methods and follows from the combined effect of the local interactions, the hierarchical structure of neighborhoods, and the tabu search intensification phases used at levels 0 and 1.

A second difference between the independent and multilevel cooperative parallel methods appeared in the behavior of the evolution of the best solution values of the

participating searches: the graphic representations of these trajectories crossed each other significantly less for multilevel cooperative strategy than for the independent one. The frequency of trajectory crossings is generally taken to indicate the degree of overlapping of the corresponding search threads. One may thus infer that the multilevel cooperative neighborhood structure based on increasing sizes of blocks is relatively successful at preventing overlaps among parallel explorations of the solution space. Moreover, the gradual improvement observed once the cooperation is active supports the claim that cooperation by local interactions among levels is successful in driving levels 0 and 1 towards good regions of the solution space.

6 Perspectives

The development of this first multilevel cooperative parallel meta-heuristic for the CMND problem, together with the results of the numerical experiments that were performed, pointed to a number of research issues that should conduct to an enhanced methodology. While addressing these issues is beyond the scope of the present paper, we briefly discuss them as guidelines for future research.

A number of issues have been highlighted by the experiments. For example, a relatively high number of infeasible solutions were being generated. Also, searches proceeding at high levels (e.g., levels, 5, 6, and 7) were not very productive in terms of generating good solutions. We also noticed that the searches at the various levels did not always proceed at the same pace, and produced networks with dissimilar designs. We believe that one of the main reasons behind these behavior is the very (probably, too) simple and stringent definition of the cooperation mechanism, particularly the coarsening operation and factors.

Indeed, the cycle-based neighborhood definition is prone to yield moves that bring the search to infeasible solutions (see the example in Ghamlouche, Crainic, and Gendreau 2003). Fixing arcs, particularly fixing them closed, reduces the search space and increases the likelihood of such moves, especially for the problem instances tested, characterized by a tight total capacity compared to total demand. Then, when large coarsening factors are used for high levels, the search space becomes too constrained and many moves conduct to infeasible solutions. On the other hand, fixing arcs modifies the problem instance and thus the behavior of the tabu search. Thus, when computing cycles, the cost of moving flow on a fixed-open arc is equal to the variable cost, while this cost is weighted by a linearization of the fixed cost when the arc is undecided. This may bias the search towards using the fixed arcs and not allow a thorough exploration of the neighborhood.

One must also recall that, for CMND problems, the search proceeds in the space of the design variables only. The feasibility of the design resulting from a move as well as the associated value of the design and flow distribution must therefore be computed by solving a rather complex capacitated, multicommodity minimum-cost network flow problem. The impact of aggregating variables in blocks to define levels is therefore much more difficult to determine than, for example, in the case of the graph partitioning problem. Block building may thus set the search in a not-so-interesting region of the solution space of the original problem and many interactions may be required before the search moves toward a more interesting region.

A challenging research direction corresponds therefore to the study of coarsening operators. The ones presented in this paper are quite simple, as they are based on characteristics of individual arcs. More elaborated ones could exploit the characteristics of the multicommodity network design problem, e.g., the paths or trees used to move commodities. The relations between the problem characteristics, in terms of dimensions and capacity, on the one hand, and the number of levels and coarsening operators and factors, on the other hand, should also be part of this study.

A second, and complementary, research direction concerns the flexibility of the cooperation structure. Currently, the mechanism is quite strict. The operators are invoked at regular intervals, in strict order. The literature on parallel meta-heuristics indicates, however, that introducing flexibility in the algorithms is beneficial. Thus, while the strict mechanism used in the present paper yields very good results, we plan to study more advanced mechanisms that would allow the exploration to dynamically adapt to the problem instance and the state of the search.

Exploring in this direction brings up issues related to the utilization of memories in cooperation. Simple memories are already included in the algorithm and used by the interaction operators. More advanced memories may be built. Indeed, similar to solution values, other solution and exploration attributes may be put into memories and associated to interaction operators. Such data may even be part of the information diffusion process. The utilization of this diffusion to create or approximate at each level an image of the status of the entire search is a fascinating challenge.

7 Conclusions

We presented the first multilevel cooperative parallel tabu search algorithm for the capacitated multicommodity network design problem. Computational results on a set of benchmark problem instances show that this approach yields good quality solutions comparable to those obtained by the current best meta-heuristics for the problem.

These results are very encouraging but they also emphasize the need for more research to understand better the interplay between the CMND problem, the cycle-based tabu search, and the multilevel strategy for cooperative parallel search, and to develop the next generation of algorithms. We are currently undertaking this research and hope to report in the near future.

Acknowledgments

Funding for this project has been provided by the Natural Sciences and Engineering Council of Canada, through its Discovery Grant programs, and by the Fonds F.C.A.R. of the Province of Québec.

While working on this project, Dr. T.G. Crainic was Adjunct Professor at the Département d'informatique et de recherche opérationnelle of the Université de Montréal (Canada), the department of Computer Science of the University of Manitoba at Winnipeg (Canada), and at Molde University College (Norway).

References

Armacost, A.P., Barnhart, C., and Ware, K.A. (2002). Composite Variable Formulations for Express Shipment Service Network Design. *Transportation Science*, 36(1):1–20.

Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. (1997). Network Design. In Dell'Amico, M., Maffioli, F., and Martello, S., editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. John Wiley & Sons, New York, NY.

Chouman, M., Crainic, T.G., and Gendron, B. (2003). A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Network Design. Technical Report CRT-2003-16, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Cong, J. and Shinnert, J.R., editors (2003). *Multilevel Optimization in VLSICAD*, volume 14 of *Combinatorial Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Crainic, T.G. (2000). Network Design in Freight Transportation. *European Journal of Operational Research*, 122(2):272–288.

Crainic, T.G. (2005). Parallel Computation, Co-operation, Tabu Search. In Rego, C. and Alidaee, B., editors, *Metaheuristic Optimization Via Memory and Evolution: Tabu Search and Scatter Search*, pages 283–302. Kluwer Academic Publishers, Norwell, MA.

Crainic, T.G., Frangioni, A., and Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design. *Discrete Applied Mathematics*, 112(1–3):73–99.

Crainic, T.G. and Gendreau, M. (2002). Cooperative Parallel Tabu Search for Capacitated Network Design. *Journal of Heuristics*, 8(6):601–627.

Crainic, T.G., Gendreau, M., and Farvolden, J.M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12(3):223–236.

Crainic, T.G., Gendron, B., and Hernu, G. (2004a). A Slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design. *Journal of Heuristics*, 10(5):525–545.

Crainic, T.G. and Rousseau, J.-M. (1986). Multicommodity, Multimode Freight Transportation: A General Modeling and Algorithmic Framework for the Service Network Design Problem. *Transportation Research B: Methodological*, 20:225–242.

Crainic, T.G. and Toulouse, M. (1998). Parallel Metaheuristics. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 205–251. Kluwer Academic Publishers, Norwell, MA.

Crainic, T.G. and Toulouse, M. (2003). Parallel Strategies for Meta-heuristics. In F. Glover and G. Kochenberger, editors, *Handbook in Metaheuristics*, pages 475–513. Kluwer Academic Publishers, Norwell, MA.

Crainic, T.G., Toulouse, M., and Gendreau, M. (1997). Towards a Taxonomy of Parallel Tabu Search Algorithms. *INFORMS Journal on Computing*, 9(1):61–72.

Crainic, T.G., Toulouse, M., and Li, Y. (2004b). A Simple Cooperative Multilevel Algorithm for the Capacitated Multicommodity Network Design. Publication CRT-2004-, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Cung, V.-D., Martins, S.L., Ribeiro, C.C., and Roucairol, C. (2002). Strategies for the Parallel Implementations of Metaheuristics. In Ribeiro, C. and Hansen, P., editors, *Essays and Surveys in Metaheuristics*, pages 263–308. Kluwer Academic Publishers, Norwell, MA.

Farvolden, J.M. and Powell, W.B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28(3):256–272.

Fiduccia, C.M. and Matteyses, R.M. (1982). A Linear Time Heuristics for Improving Network Partitions. In *Procedings 19th ACM/IEEE Design Automation Conference*, pages 175–181. IEEE Computer Society Press.

Gendron, B. and Crainic, T.G. (1994). Relaxations for Multicommodity Network Design Problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B. and Crainic, T.G. (1996). Bounding Procedures for Multicommodity Capacitated Network Design Problems. Publication CRT-96-06, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Gendron, B., Crainic, T.G., and Frangioni, A. (1998). Multicommodity Capacitated Network Design. In Sansó, B. and Soriano, P., editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Norwell, MA.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2003). Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51(4):655–667.

Ghamlouche, I., Crainic, T.G., and Gendreau, M. (2004). Path Relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research*, 131:109–133.

Hendrickson, B. and Leland, R. (1993). A Multilevel Algorithm for Partitioning Graphs. Report SAND93-1301, Sandia National Laboratories.

Holmberg, K. and Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*, 48(3):461–481.

Holmqvist, K., Migdalas, A., and Pardalos, P.M. (1997). Parallelized Heuristics for Combinatorial Search. In Migdalas, A., Pardalos, P., and Storoy, S., editors, *Parallel Computing in Optimization*, pages 269–294. Kluwer Academic Publishers, Norwell, MA.

Jarvis, J.J. and Mejia de Martinez, O. (1977). A Sensitivity Analysis of Multicommodity Network Flows. *Transportation Science*, 11(4):299–306.

Kernighan, B. and Lin, S. (1970). An Efficient Heuristic Procedure for Partitioning Graphs. *Bell System Technical Journal*, 49:291–307.

Kim, D., Barnhart, C., Ware, K., and Reinhardt, G. (1999). Multimodal Express Package Delivery: A Service Network Design Application. *Transportation Science*, 33(4):391–407.

Koskosidis, Y.A., Powell, W.B., and Solomon, M.M. (1992). An Optimization–based Heuristic for Vehicle Routing and Scheduling with Soft Time Windows Constraints. *Transportation Science*, 26:69–85.

Magnanti, T.L. and Wong, R.T. (1984). Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, 18(1):1–55.

Minoux, M. (1989). Network Synthesis and Optimum Network Design Problems: Models, Solution Methods and Applications. *Networks*, 19:313–360.

Ouyang, M., Toulouse, M., Thulasiraman, K., Glover, F., and Deogun, J.S. (2000). Multi-Level Cooperative Search: Application to the Netlist/Hypergraph Partitioning Problem. In *Proceedings of International Symposium on Physical Design*, pages 192–198. ACM Press.

Ouyang, M., Toulouse, M., Thulasiraman, K., Glover, F., and Deogun, J.S. (2002). Multilevel Cooperative Search for the Circuit/Hypergraph Partitioning Problem. *IEEE Transactions on Computer-Aided Design*, 21(6):685–693.

Pardalos, P.M., Pitsoulis, L., Mavridou, T., and Resende, M.G.C. (1995). Parallel Search for Combinatorial Optimization: Genetic Algorithms, Simulated Annealing, Tabu Search and GRASP. In Ferreira, A. and Rolim, J., editors, *Proceedings of Workshop on Parallel Algorithms for Irregularly Structured Problems, Lecture Notes in Computer Science*, volume 980, pages 317–331. Springer-Verlag, Berlin.

Powell, W.B. (1986). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, 20(4):246–357.

Rochat, Y. and Taillard, É.D. (1995). Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. *Journal of Heuristics*, 1(1):147–167.

Toulouse, M., Crainic, T.G., and Gendreau, M. (1996). Communication Issues in Designing Cooperative Multi Thread Parallel Searches. In I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory & Applications*, pages 501–522. Kluwer Academic Publishers, Norwell, MA.

Toulouse, M., Crainic, T.G., and Sansó, B. (2004). Systemic Behavior of Cooperative Search Algorithms. *Parallel Computing*, 30(1):57–79.

Toulouse, M., Crainic, T.G., Sansó, B., and Thulasiraman, K. (1998a). Self-Organization in Cooperative Search Algorithms. In *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, pages 2379–2385. Omnipress, Madisson, Wisconsin.

Toulouse, M., Crainic, T.G., and Thulasiraman, K. (2000). Global Optimization Properties of Parallel Cooperative Search Algorithms: A Simulation Study. *Parallel Computing*, 26(1):91–112.

Toulouse, M., Glover, F., and Thulasiraman, K. (1998b). A Multi-Scale Cooperative Search with an Application to Graph Partitioning. Report, School of Computer Science, University of Oklahoma, Norman, OK.

Toulouse, M., Thulasiraman, K., and Glover, F. (1999). Multi-Level Cooperative Search: A New Paradigm for Combinatorial Optimization and an Application to Graph Partitioning. In Amestoy, P., Berger, P., Daydé, M., Duff, I., Frayssé, V., Giraud, L., and Ruiz, D., editors, *5th International Euro-Par Parallel Processing Conference*, volume 1685 of *Lecture Notes in Computer Science*, pages 533–542. Springer-Verlag, Heidelberg.

Verhoeven, M.G.A. and Aarts, E.H.L (1995). Parallel Local Search. *Journal of Heuristics*, 1(1):43–65.