CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

# Parallel Solution Methods for Vehicle Routing Problems

**Teodor Gabriel Crainic**

**August 2007**

**CIRRELT-2007-28**

**Bureaux de Montréal :**
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

**Bureaux de Québec :**
Université Laval
Pavillon Palasis-Prince, local 2642
Québec (Québec)
Canada G1K 7P4
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · UQÀM · HEC Montréal · École Polytechnique Montréal · Université de Montréal

# Parallel Solution Methods for Vehicle Routing Problems

**Teodor Gabriel Crainic**[1,*]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 and Chaire de recherche industrielle du CRSNG en management logistique, ESG, Université du Québec à Montréal, C.P. 8888, succursale Centre-ville, Montréal, Canada H3C 3P8

**Abstract.** Parallel solution methods contribute to efficiently address large and complex combinatorial optimization problems, vehicle routing problems in particular. Parallel exact and heuristic methods for VRP variants are increasingly being proposed and the pace seem to increase in recent years. “New” strategies have been proposed and many of the best known solutions to classical formulations have been obtained. The paper describes and discusses the main strategies used to parallelize exact and meta-heuristic solution methods for vehicle routing problems. It also provides an up-to-date survey of contributions to this rapidly evolving field and points to a number of promising research directions.

**Keywords**. Parallel computation, parallelization strategies, branch-and-bound, meta-heuristic, vehicle routing problems.

**Acknowledgements.** Funding for this project has been provided by the Natural Sciences and Engineering Research Council of Canada (NSERC). The authors wish to thank three anonymous referees. Their comments have contributed to make a better paper.



---

* Corresponding author: Teodor-Gabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque nationale du Québec,
Bibliothèque nationale du Canada, 2007

# 1 Introduction

The *Vehicle Routing Problem* (*VRP)* is one of the core operations research and combinatorial optimization problem classes with numerous applications in transportation, telecommunications, production planning, and so on. The basic VRP may be briefly described as follows. Given one or more depots, a fleet of vehicles, homogeneous or not, and a set of customers with known or forecast demands, find a set of closed routes, originating and, generally, ending at one of the depots, to service all customers at minimum cost, while satisfying vehicle and depot capacity constraints. Other characteristics and requirements may be considered, such as service and travel time restrictions, multiple commodities with different transportation requirements, time-dependent and uncertain demands or travel times, etc., yielding a rich set of VRP variants.

Vehicle routing problems have been the object of numerous studies and a very large number of papers propose solution methods. Because most VRP variants are NP-Hard, exact solution methods are confined to limited-size problem instances. Heuristics, meta-heuristics principally, are thus proposed in most cases. Contributions are continuously being made to VRP methodology and practice alike. Yet, despite the progress the field has seen in recent years, many challenges still stand and new ones are emerging.

Problem instances of interest are becoming larger. More importantly, problems are becoming more complex in terms of "new" constraints and objectives that are added to the generic formulations the community usually focuses on. The term *rich vrp* has been coined to identify these new and challenging problem variants. Two additional trends may be observed that contribute to making VRP a challenging and interesting field. On the one hand, the time available to reach a decision is limited for several problem settings, such as the real-time routing and re-routing problems. These settings require therefore solution methods displaying high computational efficiency without a decrease in solution quality. On the other hand, there is the need for "simpler" but efficient and robust methods, that is, solution approaches that do not require complex and extensive calibration procedures, while still offering high performance over a broad range of problem instances. Beyond their intrinsic elegance, such methods provide for added flexibility in practical use.

Parallel optimization contributes toward meeting these challenges. As illustrated by the survey papers indicated further in this Introduction, parallel exact, e.g., branch-and-bound, and meta-heuristic methods have been successfully applied to many diverse combinatorial optimization problems. One is surprised to realize, however, that relatively few developments have targeted vehicle routing problems, most of those that do having been proposed from the turn of the millennium on. This is certainly the case for parallel branch-and-bound methods. The author is not aware of any contribution related to VRP published before the year 2000 and of very few published after that (Section 3). The case is not as extreme regarding parallel meta-heuristics. Even though the contributions related to VRP published before the year 2000 are not as numerous as for other combinatorial optimization problems [21], a few particular ones have had a significant impact on the field (e.g., the adaptive memory

concept - see Section 4). Yet, a much larger number of contributions have been proposed starting around the turn of the millennium. "New" strategies have been proposed and many of the best known solutions to classical formulations have been obtained by applying them.

The objective of this paper is twofold. First, to describe and discuss the main strategies used to parallelize exact and meta-heuristic solution methods for vehicle routing problems. Second, to provide an up-to-date survey of contributions to this rapidly evolving field and point to a number of promising research directions.

The paper is organized as follows. Section 2 recalls basic concepts in parallel computing and indicates a number of general references. Section 3 introduces parallel tree-search based algorithms and surveys the few VRP applications proposed so far. Section 4 presents parallel meta-heuristic strategies, while Section 5 surveys recent parallel meta-heuristic contributions to VRP variants. Section 6 concludes the chapter.

# 2 A Very Brief Tour of Parallel Computing

To limit the length of the paper, we restrict to a minimum the discussion of general parallel computing issues (see, e.g., [10] for a more in-depth presentation), as well as the presentation of general parallel branch-and-bound and meta-heuristic strategies (Sections 3 and 4, respectively).

Parallel/distributed computing applied to problem solving means that several processes work simultaneously on several processors with the common goal of solving a given problem instance. Parallelism thus follows from a decomposition of the total workload and the distribution of the resulting tasks to the available processors. The decomposition may concern the algorithm or the problem-instance data. In the former case, denoted *functional parallelism*, different tasks, possibly working on the "same"' data, are allocated to different processors and run in parallel, possibly exchanging information. The latter is denoted *data parallelism* or *domain decomposition* and refers to the case where the feasible domain of the problem considered is partitioned and a particular solution methodology is used to address the problem on each of the components of the partition. According to how "large" the tasks are, a few instructions or a sizable part of the algorithm, the parallelization is called *fine-* or *coarse-grained*, respectively. When the tasks are independent or weakly correlated (e.g., partitioning two matrices to speed up their multiplication), the same work is performed in parallel and in sequential, but the former is faster, the wall-clock time being reduced proportionally to the average number of tasks that are run concurrently during the computation.

There are very few cases of such "pure" parallelism in optimization, however. Consequently, the volumes of work performed by the sequential and parallel versions of a solution method, are different in all but the simplest cases, e.g., low-level parallelism where only the work of computing-intensive tasks, such as the computation of a bound or the evaluation of

a neighborhood, is decomposed. This is particularly true when the parallel method implements a *multi-search* strategy rather than a direct parallelization of a given algorithm. In case, several search methods, including but not restricted to branch-and-bound and metaheuristics, explore simultaneously the solution space of the same problem instance. The methods proceed independently but may engage in various types of communication and information sharing, the intensity, scope, and form of communication defining the particular multi-search strategy.

Information must be exchanged among tasks to provide the necessary data for computations or the estimation of the global status of the search. Communications may be performed *synchronously* and *asynchronously*. In the former case, all concerned tasks have to stop and engage in some form of communication and information exchange at moments (number of iterations, time intervals, specified algorithmic stages, etc.) exogenously determined, either hard-coded or determined by a control (master) task. In the latter case, each task is in charge of establishing communications with other tasks, according to its internal logic. The frequency of communication and the volume of exchanged information vary with the particular solution method and parallelization strategy, and are often central to the success of the parallel method. Communications must be carefully controlled, however, to avoid that the associated search overhead obliterates the gains of decomposition.

Recall that the traditional goal when designing parallel solution methods is to reduce the time required to "solve", exactly or heuristically, given problem instances or to address larger instances without increasing the computational effort. For exact solution methods run until the optimal solution is obtained, this translates into the well-known *speedup* performance measure, computed as the ratio between the wall-clock time required to solve the problem instance in parallel with $p$ processors and the corresponding solution time of the best-known sequential algorithm; A somewhat less restrictive measure replaces the latter with the time of the parallel algorithm run on a single processor. See [6] for a detailed discussion of this issue, including additional performance measures.

Speedup measures are more difficult to define when the optimal solution is not guaranteed or the exact method is stopped before optimality is reached. Indeed, for most parallelization strategies, the sequential and parallel versions of a heuristic yield solutions that are different in value, composition, or both. Thus, an equally important objective when parallel heuristics are contemplated is to design methods that outperform their sequential counterparts in terms of solution quality and, ideally, computational efficiency (i.e., the parallel method should not require a higher overall computation effort than the sequential method or should justify the effort by higher quality solutions). Search robustness is another characteristic increasingly expected of parallel heuristics. Robustness with respect to a problem variant is meant here in the sense of providing "equally" good solutions to a large and varied set of problem instances, without excessive calibration, neither during initial development, nor when addressing new problem instances. See [22, 23] for a discussion of these issues.

The reader may consult a number of surveys, taxonomies, and syntheses of parallel metaheuristics, some addressing methods based on particular methodologies, while others address

the field in more comprehensive terms. Two recent books [1] and [77] collect chapters on many issues in parallel computing for combinatorial optimization. Syntheses of strategies for parallel branch-and-bound and branch-and-cut may be found in [19] and [64], respectively, while [4], [44], [45], and [67] address parallel simulated annealing; [13], [52], [54], and [73] parallel evolutionary and genetic algorithms; [25], [18], [42], and [80] parallel tabu search; [12] and [33] ant-based methods; and [53] parallel Variable Neighborhood Search (VNS). Surveys and syntheses that address more than one methodology may be found is [17], [22], [23], [21], [26], [46], [59], and [79].

# 3 Exact Search Methods

Research into exact methods for vehicle routing problems is advancing at a steady pace focusing on branch-and-bound methods with column generation (branch-and-price) plus, eventually, the addition of valid inequalities to strengthen the formulations and the bounds (branch-and-cut and branch-and-cut-and-price). Yet, very few efforts have been dedicated to developing parallel algorithms for VRP and variants. This is in contrast to many other combinatorial optimization areas for which a rather rich literature on parallel branch-and-bound exists. (For the sake of simplicity of presentation, unless otherwise specified, we discuss issues from the point of view of branch-and-bound search.) Still, we chose to address this topic because we believe it offers interesting perspectives in terms of efficient vehicle routing problem solving, either by itself or combined to the co-operative meta-heuristics described in Section 4. We initiate the section by briefly discussing sources of parallelism in branch-and-bound methods, and conclude with the presentation of recent results.

Two basic, by now classic, approaches are known to accelerate the branch-and-bound search: *node* and *tree*-based strategies. Domain decomposition is not mentioned in the parallel branch-and-bound literature because, in fact, it may be seen as a particular tree-based strategy in which branching at the root node of the tree partitions the problem domain. As for *multi-search* strategies, even though the topic is often mentioned as an interesting research direction, significant research efforts have yet to be dedicated to their development.

Node-based strategies aim to accelerate the branch-and-bound search by executing concurrently operations associated to subproblems (nodes) of the tree: evaluation, bounding, and branching. The operations contemplated range from "simple" numerical tasks (e.g., matrix inversion), to the decomposition of computing-intensive tasks (e.g., the generation of cuts), to parallel mathematical programming (e.g., simplex, Lagrangean relaxation, capacitated multicommodity network flow) and meta-heuristic (e.g., tabu search) methods used to compute lower bounds and to derive feasible solutions. This class of strategies has also been identified as *low-level* (or *type 1*) parallelization, because it does not aim to modify the search trajectory, the dimension of the branch-and-bound tree, or its exploration.

Most parallel branch-and-bound algorithms implement some form or another of tree ex-

ploration in parallel. The fundamental idea is that, since in most applications of interest the size of the branch-and-bound tree grows rapidly to unmanageable proportions, distributing the tree-exploration work among several concurrent processes would reduce the total computation time.

Recall that sequential branch-and-bound is fundamentally a procedure that first selects and extracts a subproblem, a node, from a data structure, the *pool*. It then performs a series of operations, evaluation, computation of upper bound, branching, and so on, to finally complete the loop by inserting into the pool one or several nodes, the new subproblems yielded by the branching operation. Subproblems in the pool are usually kept and accessed according to their *priority* based on node attributes (e.g., lower-bound value) and the search tree exploration strategy (e.g., best or depth first). An evident property of sequential branch-and-bound is that each node-selection decision is taken with a complete knowledge (information) regarding the status of the search, that is, with the global view of all the nodes generated so far. In a parallel environment, both the search decisions and information may be distributed. Relative to the former, different processors may decide more or less simultaneously what node to select and process next. The main source of information, the pool, may also be distributed. Consequently, not all the relevant information may be available at the time and place (i.e., the processor) a decision is taken. Moreover, work may come in short supply for some processors, while others experience excessive loads, which requires a *load-balancing* procedure to be undertaken. Parallel tree exploration methods may therefore be described in terms of *search-control* (or *scheduling*) strategies made up of *decision* and *pool* management rules, respectively.

The pool of nodes is the main source of knowledge and may be kept in a unique data structure, a *centralized* pool, and thus be made available to all associated processors. Alternatively, the pool may be *distributed*, in which case a processor or a group of processors has direct access to its local pool only. We may then define the *search knowledge* of a given processor as the available information relative to the pool of nodes with their priorities (plus the incumbent value and the other global status variables of the search). We define also the *workload knowledge* as the information available to a given processor regarding the load factors of all relevant processors involved in the search. When branch-and-cut methods are contemplated, similar concepts are introduced for the sets (pools) of local and global cuts.

The control of a branch-and-bound search is made up of a number of decisions: node selection, allocation of node work (i.e., the operations related to subproblem evaluation and bound computation), incumbent update, termination determination and, for distributed-pool-based strategies, load balancing. *Search control* is then primarily defined by the number of processors that collaborate to guide the search and the quantity of available information. Generally speaking, decision-making may be *centralized* or *distributed* and may be based on *complete* or *partial* knowledge.

A centralized search control refers to the case when a single processor, the so-called "master", makes all the search decisions, based on the complete search and workload knowledge given by a centralized pool, but distributes node work to a number of processors. It

is the classical master-slave strategy. We refer to distributed-search control when the node selection decision is distributed among several processors. Providing complete information in this case is very difficult and usually involves significant search overheads in terms of synchronization and information exchange. Thus, most distributed-search control strategies are associated to distributed node pools, partial information, and asynchronous communication schemes. The term "collegial" is often used to denote such strategies. Of course, load-balancing mechanisms, based on the currently available workload knowledge, have to be included into the parallel algorithm design to avoid processor idle time due to lack of work or processors performing unnecessary work due to the poor quality of the nodes in their local pools.

Few parallel branch-and-bound algorithms targeted vehicle routing problems. This has started to change recently, with the VRP being used as one of the testbeds in the development of strategies and libraries (Ralphs, Ladány, and Saltzman [66]) for parallel branch-and-cut (and price) algorithms (Ralphs [63], Ralphs, Ladány, and Saltzman [65]). The current implementation uses a centralized pool of nodes, as well as a centralized pool of cuts. This provides for complete knowledge during the search. A master processor than guides the branch-and-bound search, dispatching node-related work to a number of other processors. An alternate strategy making use of multiple cut pools has also been tested. The advantage of this approach is to limit the number of "slave" processors a cut pool must service when nodes are examined, which makes for a more scalable algorithm. Furthermore, multiple pools also make for a more efficient procedure to scan the cuts and verify whether they apply to a given node and facilitate the utilization of local nodes (local to a node and its descendants). Extensive experimental work showed that, as expected, the centralized-knowledge strategy preserves the logic of the sequential tree exploration and avoids most redundant work. On the other hand, the procedure was efficient on a limited number of processors, only. The authors are thus developing tools to implement hierarchical parallel strategies where, at the first level a number of processors collegially explore the tree (a "master" is in charge of load balancing) while, at the second level, each of these processors distributes node-related work to several other "slave" processors. Asynchronous communications, distributed node pools (first level) and multiple cut pools are part of the in-development design.

# 4 Parallel Meta-heuristics

Given the difficulty of routing problems, most solution methods that are proposed are based on heuristics and meta-heuristics. It is therefore not surprising that most parallel solution methods proposed are based on meta-heuristics as well. This section recalls and briefly discusses the principal parallel meta-heuristic strategies proposed in the literature. Applications to VRP and variants published before 2000 are also indicated.

To describe parallel meta-heuristic strategies, we adopt the classification of Crainic and Nourredine [21], which generalizes that of Crainic, Toulouse, and Gendreau [25] (see also

[17], [22], and [23]; Verhoeven and Aarts [79] and Cung *et al.* [26] present classifications that proceed of the same spirit). This classification is sufficiently general to encompass all meta-heuristic classes, while avoiding a level of detail incompatible with the scope and dimension limits of the paper. The classification and parallel meta-heuristic strategies will be used in Section 5 to analyze recent contribution to VRP and variants.

The three dimensions of the classification indicate how the global problem-solving process is controlled, how information is exchanged among processes, and the variety of solution methods involved in the search for solutions, respectively. The first dimension, *Search Control Cardinality*, thus examines how the global search is controlled: either by a single process or collegially by several processes that may collaborate or not. The two alternatives are identified as *1-control (1C)* and *p-control (pC)*, respectively. The second dimension, relative to the type of *Search Control and Communications*, addresses the issue of information exchange according to four classes to reflect the quantity and quality of the information exchanged and shared, as well as the additional knowledge derived from these exchanges (if any): *Rigid (RS)* and *Knowledge Synchronization (KS)* and, symmetrically, *Collegial (C)* and *Knowledge Collegial (KC)*.

Because more than one solution method or variant (e.g., different parameter settings) may be involved in a parallel meta-heuristic, the third dimension indicates the *Search Differentiation*: do search threads start from the same or different solutions and do they make use of the same or different search strategies? The four cases considered are: *SPSS, Same initial Point/Population, Same search Strategy*; *SPDS, Same initial Point/Population, Different search Strategies*; *MPSS, Multiple initial Points/Populations, Same search Strategies*; *MPDS, Multiple initial Points/Populations, Different search Strategies*. Obviously, one uses "point" for neighborhood-based methods such as Simulated Annealing, Tabu Search, Variable Neighborhood Search, GRASP, Guided Local Search, etc., while "population" is used for Evolutionary methods (e.g., Genetic Algorithms), Scatter Search, and Ant Colony methods.

Typically, 1-control strategies implement a classical master-slave approach that aims solely to accelerate the search. Here, a "master" processor executes a sequential meta-heuristic but dispatches computing-intensive tasks to be executed in parallel by "slave" processes. The master receives and processes the information resulting from the slave operations, selects and implements moves or, for population-based methods, selects parents and generates children, updates the memories (if any) or the population, and decides whether to activate different search strategies or stop the search.

In the context of neighborhood-based search, the operation most widely targeted in such approaches is the neighborhood evaluation. At each iteration, the possible moves in the neighborhood of the current solution are partitioned into as many sets as the number of available processors and the evaluation is carried out in parallel by slave processes (e.g., the master-slave parallelization scheme of Garcia, Potvin, and Rousseau [36] for the VRP with time-window constraints). For population-based methods, it is the fitness evaluation that is most often targeted in 1C/RS/SPSS strategies.

Probing or look-ahead strategies belong to the 1C/KS class with any of the search differentiation models identified previously. For neighborhood-based methods, such an approach may allow slaves to perform a number of iterations before synchronization and the selection of the best neighbor solution from which to proceed (one may move directly to the last solution identified by the slave process or not). For population-based methods, the method may allow each slave process to generate child solutions, "educate" them through a hill climbing or local search procedure, and play out a tournament to decide who of the parents and children survive and are passed back to the master.

The impact of such "low level", 1-control strategies has proved limited, however. The search trajectory of the parallel procedure is quite similar to that of its sequential counterpart (it is the same, for 1C/RS/SPSS strategies). Of course, when neighborhoods are large or neighbor-evaluation procedures are costly, the corresponding gain in computing time may prove interesting. Then, when a sufficiently large number of processors is available, it might prove worthy to combine such an approach to more sophisticated strategies into hierarchical solution schemes (e.g., Rego and Roucairol [68] who used low-level parallelism to accelerate the move evaluations of the individual searches engaged into an independent multi-thread procedure for the VRP).

*Multi-search* or *multi-thread* parallel strategies for meta-heuristics have generally offered better performances, in terms of solution quality and computing times, than the methods introduced above. Historically, independent and synchronous co-operative multi-search methods were proposed first. The emphasis currently is on asynchronous communications and co-operation. Most applications of such strategies fall into the $pC$ category.

*Independent* multi-search methods belong to the pC/RS class of the taxonomy. Most implementations start several independent search processes, all using the same search strategy, from different, randomly generated, initial configurations. No attempt is made to take advantage of the multiple threads running in parallel other than to identify the best overall solution once all processes stop. This earns independent search strategies their rigid synchronization classification. This parallelization of the classic sequential multi-start heuristic is easy to implement and may offer satisfactory results, as has been demonstrated by Rego and Roucairol [68] for the VRP. Each thread implemented their ejection-chain-based tabu search with a different set of parameter settings but the same initial solution. The pC/RS/SPDS method was implemented such that each thread executed a complete sequential tabu search. The overall-best solution was then selected, the threads using it as initial solution for a new search. Low-level parallelism was used to accelerate the move evaluations of the individual searches, as well as in a post-optimization phase. Experiments showed the method to be competitive on a set of standard VRP problems [14].

Co-operative strategies often offer superior performances compared to independent search. pC/KS co-operative strategies adopt the same general approach as in the independent search case but attempt to take advantage of the parallel exploration by synchronizing processors at pre-determined intervals. An information exchange mechanism then determines the best current overall solution and the search is restarted from that point. The mechanism may

use a designated process to gather information, extract the best solution, and broadcast it to all search processes. Alternatively, each search process may be empowered to initiate synchronization (e.g., using a broadcast) of all or a pre-specified subset of processes (e.g., processes that run on neighboring processors). Here, as in the more advanced co-operation mechanisms indicated bellow, *migration* is the term used to identify information exchanges in population-based parallel algorithms.

Asynchronous co-operative multi-thread search methods belong to the pC/C or pC/KC classes of the taxonomy according to the quantity and quality of the information exchanged and, eventually, the "new" knowledge inferred based on these exchanges. Co-operative multi-thread strategies exchanging "good" solutions only and implementing simple strategies to extract solutions from the pool belong to the PC/C class of methods. When procedures are added to extract information or create new information and solutions based on the solutions exchanged, the corresponding methods are said to belong to the pC/KC class.

The design of the information communication and exchange mechanism is a key element to the good performance of multi-thread parallel meta-heuristics. Questions relative to what information to exchange, when to exchange it and among what processors, as well as what to do with the received information are of the highest importance when designing parallel meta-heuristics. Parallelism in general, and multi-thread strategies in particular, imply that both the individual searches and the resulting global search proceed most of the time with incomplete knowledge regarding the status of the search. Synchronization has been seen as a means to re-create a state of complete knowledge to share among all participating search threads. It was hopped that performances, in terms of computing efficiency and solution quality, would be improved. This did not materialize, however. In fact, compared to independent and most asynchronous search strategies, synchronous methods display larger computational overheads, appear less reactive to the evolution of the search, and conduct to the premature convergence of the dynamic process representing the parallel search. The issue is also relevant for co-operative search. It has been shown, for example, that frequent broadcasting of new solutions that stop individual threads from continuing to explore improving sequences leads to either a random search or premature convergence. Controlled, parsimonious, and timely exchanges of meaningful information are thus characteristic of successful co-operative multi-thread meta-heuristics.

Communications may be undertaken either directly or indirectly. Strategies based on the evolutionary paradigm generally use direct communications. The population is divided into subsets, each assigned to a processor (alternatively, relatively small populations are generated for each processor), and a genetic algorithm runs on each. An individual population and genetic algorithm form a so-called *island*. Each island may potentially communicate with any of the other islands, as illustrated in Figure 1. Then, according to an exchange protocol (e.g., on demand from an island with low population diversity), it sends a "good" individual to another island. This exchange mechanism is called *migration* and the parallel strategy is known as *coarse grained*. Islands (processors) may also be allowed to communicate with a limited number of other islands (processors) only, as illustrated in Figure 2. Such limitations are generally the result of particular topologies of the processor network (e.g., hypercube,

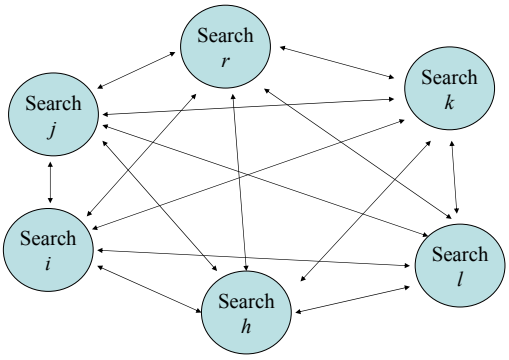


Figure 1: Many-to-many Direct Communication Scheme

torus, and so on). Communications then take place only among adjacent processors according to a so-called *diffusion* mechanism. Notice that, in this case, islands tend to have very small populations and the strategy to be denoted *fine grained*. When populations are down to single individuals, the genetic operators are applied to individuals on adjacent islands.

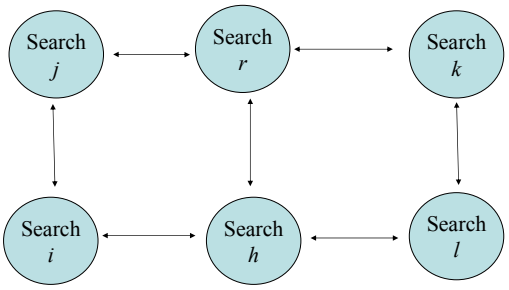


Figure 2: Diffusion Communication Scheme

Most co-operative multi-thread developments outside the evolutionary community are based on indirect communications and, currently, the largest number use some form of *memory* for inter-thread communications (the terms *pool* and *solution warehouse* are also used; due to the role assigned to the elements it contains, the terms "reference" and "elite set" are also sometimes used, while the artificial intelligence community uses a similar concept under the name "blackboard"). The individual searches are generally assigned each to a processor, as illustrated in Figure 3. A search tread either heuristically constructs new solutions, or executes a neighborhood-based improving meta-heuristic, or implements a population-based meta-heuristic, or performs post-optimization procedures on solutions in the pool. Improving meta-heuristics aggressively explore the search space, while population-based methods contribute toward increasing the diversity of solutions exchanged among the co-operating methods. When the same meta-heuristic is used by several search threads, the initial solution and particular setting of a number of important search parameters differentiate each search thread from the others.

The co-operation aspect of the parallelization scheme is achieved through asynchronous

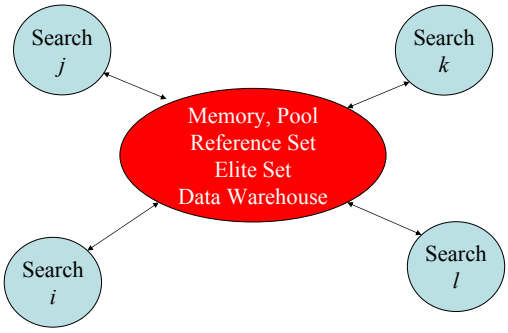


Figure 3: Diffusion Communication Scheme

exchanges of information through the pool (which could be assigned to a different processor or could share one with an individual search thread). Whenever a thread desires to send out information (e.g., when a new local optimum is identified), it sends it to the pool. Similarly, when a thread accesses outside information (to diversify the search, for example), it reaches out and takes it from the pool. Communications are initiated exclusively by the individual threads, irrespective of their role as senders or receivers of information. No broadcasting is taking place and there is no need for complex mechanisms to select the threads that will receive or send information and to control the co-operation. The solution warehouse is thus an efficient implementation device that allows for a strict asynchronous mode of exchange, with no predetermined connection pattern, where no process is interrupted by another for communication purposes, but where any thread may access at all times the data previously sent out by any other search thread.

The information exchanged among co-operating procedures has to be meaningful, in the sense that it has to be useful for the decision process of the receiving threads or the evolution of the shared data (and thus the evolution of the global search). Information indicative of the current status of the global search or, at least, of some individual search thread is, in this sense, meaningful. The information exchanged may be simply a "good" solution, a solution and its context (e.g., memories recording recent behavior of solution attributes), or a comprehensive history search. Memories recording the performance of individual solutions, solution components, or even search threads may be added to the pool and statistics and guidance mechanisms may be gradually built.

Historically, two main classes of co-operation mechanisms are found in the literature, based on partial and complete solutions, respectively. Adaptive memory methods (Rochat and Taillard [71]) store partial elements of good solutions and combine them to create new complete solutions that are then improved by the co-operating threads. Central memory approaches exchange complete elite solutions among neighborhood and population-based meta-heuristics (Crainic, Toulouse, and Gendreau [24], Crainic and Toulouse [23], Crainic [17]). The differences between the two approaches tend to become somewhat blurred, however.

A different approach to co-operation has been proposed recently by Toulouse, Thulasiraman, and Glover [78]. The mechanism is called *multi-level co-operative search*, belongs to the pC/KC with potentially any search differentiation strategy (the authors used MPSS), and is based on the principle of controlled diffusion of information. Each search thread works at a different level of aggregation of the original problem (one processor works on the original problem) and communicates exclusively with the processes working on the immediate higher and lower aggregation levels. Improved solutions are exchanged asynchronously at various moments dynamically determined by each process according to its own logic, status, and search history. Received solutions are used to modify the search at the receiving level. An incoming solution will not be transmitted further until a number of iterations have been performed, thus avoiding the uncontrolled diffusion of information. No application to vehicle routing problems has been proposed yet, but excellent results have been obtained for graph and hypergraph partitioning problems [57, 58], network design [20], feature selection in biomedical data [56], and covering design [28]. It all these cases, the proposed method is either the current best or is on the par with the best meta-heuristics for the problem.

We complete this section with two notes. The first concerns the decomposition of the problem domain. Despite its interest when problem instances are very large, relatively few contributions can be found. For routing problems, Taillard [75] proposed a pC/KS/MPSS parallel tabu search where the domain was partitioned and vehicles were allocated to the resulting regions. Once the initial partition was performed, each subproblem was solved by an independent tabu search. All processors stopped after a number of iterations that varied according to the total number of iterations already performed. The partition was then modified by an information exchange phase, during which tours, undelivered cities, and empty vehicles were exchanged between adjacent processors (corresponding to neighboring regions). At the time, this approach did allow to address successfully a number of problem instances, but the synchronization inherent in the design of the strategy hindered its performance. Clearly, more work is required on how to best combine domain decomposition and the other parallelization strategies, co-operation in particular. We report in the next section on some contributions that address this issue.

The second note is related to the so-called *hybrid* methods. The term is much used but its meaning varies widely. In a strict sense, all meta-heuristics are hybrids since they involve at least two methods. Closer to most applications, a hybrid involves at least two methods that belong to different methodological approaches. Thus, for example, using genetic operators to control the temperature evolution in a parallel simulating annealing method yields a hybrid. Notice, however, that by this definition, all population-based methods that include an "educational" component, that is, an enhancement of new solutions through a hill climbing, a local search, or even a full-blown meta-heuristic, are hybrids. Most co-operative parallel strategies could be qualified as hybrids as well. Yet, since, other than "more than one method is used", the term does not offer any fundamental insight into the design of parallel strategies for meta-heuristics, we do not use it to qualify the contributions reviewed in this paper.

# 5 Parallel Meta-heuristics for the VRP

This section is dedicated to a review of recent parallel meta-heuristic contributions to vehicle routing problems. "Recent" means that we focus on the period since the change of millennium, except when the contribution is still of significant interest. The contributions are presented according to the VRP variant concerned.

## 5.1 The vehicle routing problem

Drummond, Ochi, and Vianna [34, 35] (see also Ochi *et al.* [55]) proposed a pC/KS/MPSS coarse-grained parallel genetic algorithm based on the island model for the VRP with heterogeneous fleet. A petal decomposition procedure was used to build the initial population, which was then divided into several disjoint subpopulations. Each genetic thread evolved a subpopulation and triggered migration when subpopulation renewal was necessary. An island in this case would broadcast its need and receive the best individual of every other island. The incoming individuals would replace the worst individuals of the receiving population. Computational tests showed encouraging results in terms of solution quality and computing effort.

Alba and Dorronsoro [2] addressed the VRP in which routes have to be limited by a predefined travel time and proposed a fine-grained, cellular parallel genetic algorithm. The population was arranged in a 2 dimentional toroidal grid, each individual having 4 neighbors. Binary tournament selection was applied when selecting the mate for the first parent. Crossover was applied for these parents, then mutation and local search for the offspring. Two local search procedures were tested, 2-opt and 2-opt+$\lambda$-Interchange, with $\lambda \in \{1, 2\}$. Elitist replacement was used. The authors compared their algorithm to several heuristics, parallel or not: the tabu search of Rochat and Taillard [71], the genetic algorithms of Prins [62] and Berger and Barkaoui [8], the ant algorithms of Bullnheimer, Hartl, and Strauß [11] and Reimann, Doerner, and Hartl [69]. Computational results on benchmark problem instances showed high performance quality for both local search versions. Best performance (solution quality and rapidity) was observed for 2-opt+1-Interchange.

Jozefowiez, Semet, and Talbi [48, 49] addressed a vehicle routing problem in which the total length of routes is to be minimized, as well as the balance of route lengths, that is the difference between the maximal and minimal route lengths. The authors proposed an island method where each island had its own population and run the multi-objective pareto genetic algorithm NSGA II [29] enhanced with an elitist feature for greater diversity. The islands were organized into a ring network. Communications took place at regular intervals, determined by the number of iterations. Each island synchronously exchanged information with its two neighbors by sending its best solutions (the number of solutions was pre-defined) and receiving the best solutions of its neighbors. The imported solutions replaced the worst ones in the receiving population. Experiments on the Christofides, Mingozzi, and Toth

problem instances [14] showed the parallel versions to outperform the sequential one, even though increasing the number of processors over the 4 to 8 range did not seem beneficial.

¿From a parallel computing point of view, ant-based methods may be viewed as a particular form of population-based methods, the ant colony being made up of a population of ants and the update of the pheromone matrix taking the place of the usual evolutionary operators. Parallel ant-based methods start to be proposed based on these ideas, some of which are dedicated to vehicle routing problems. Doerner *et al.* [31] (see also Doerner *et al.* [30] and Benkner *et al.* [7]) studied fine and coarse-grained 1C/KS/MPDS parallelizations with synchronous communications for their savings-based heuristic (Reimann, Stummer, and Doerner [70]). In the fine-grained approach, the ant colony was partitioned into small sub-colonies (the sparsely populated islands of parallel genetic algorithms) and the savings-based heuristic was executed on each. The same pheromone matrix was used for all sub-colonies, but was replicated to decrease communications. Once all ants found their solutions, the local best for each sub-colony was found. Best solutions were sent to a "root" node, which determined the global best and an elite set of ants that were broadcasted back to the sub-colonies.

Two coarse-grained strategies were also studied. The first is a classical pC/RS/MPDS independent multi-colony approach. The second follows a pC/KS/MPDS co-operation scheme where several independent colonies communicate at regular intervals (pre-defined number of iterations). To speed up computations, a hierarchical strategy was used, in which a number of processors were allocated to each colony to implement the fine-grained parallelization described above. The authors compared several strategies with respect to shared information: the global-best solution, the global best solution plus the elite solutions, each of the first two data sets plus the re-initialization of the pheromone matrix, and the global-best solution plus the corresponding pheromone matrix. Experimentations were performed on the four largest problem instances of [14] (single depot, 150 or 199 customers, limits on vehicle capacity and tour length (2 problems), zero service time at all customers) and on four of the largest problem instances of [43] (single depot, between 320 or 480 customers, various customer distributions). Two colonies were used for the experiments.

Computational results showed, yet again, the co-operative strategy outperforming the independent search method. The comparison of the information-sharing strategies also showed that the knowledge relative to parallel meta-heuristics accumulated in the last ten years is equally valid when the parallelization of ant-colony methods is concerned. Thus, sharing the elitist solutions outperformed the strategy where only the global best was shared. Further improvements were obtained by also re-initializing the pheromone matrix. On the other hand, broadcasting the pheromone matrix of the best performing sub-colony (the one that identified the current global best solution) was detrimental to performance due to the premature "convergence" of the dynamic process. As for the fine-grained strategy, as expected, intensive communications were required to keep the pheromone matrix up to date. This makes the approach not really suitable when the number of processors (and ants) increases, but may contribute to hierarchical approaches.

In the same paper [31] (see also Doerner, Hartl, and Lucka [32]), the authors examined a parallelization of their *D-Ants* algorithm (Reimann, Doerner, and Hartl [69]), which applies the domain-decomposition ideas of Taillard [75] to the savings-based heuristic (Reimann, Stummer, and Doerner [70]) significantly improving its performance. A hierarchical coarse-grained approach similar to that described previously was proposed. Two or four sub-problems were defined, and the parallel savings algorithm was used on each, thus implementing the fine-grained parallel strategy described previously. In all their developments, the authors aimed for parallel strategies that sped up computations but did not change the behavior of the corresponding method. Strict synchronization communication schemes were imposed to this effect. Consequently, in all experiments, solution quality was sensibly the same and moderate speed ups were observed. To conclude, the authors point out to the need to develop more sophisticated parallel strategies based on asynchronous co-operation mechanisms.

## 5.2 Vehicle Routing with Time Constraints

Also known as the *Vehicle Routing Problem with Time Windows* (*VRPTW*), this problem specifies that service at customer sites must take place within given time intervals. Most time constraints specify that service cannot begin before a certain moment (but vehicles may wait "outside", in most cases) and must be over by a given deadline. In soft-constrained versions, the time limits may be violated at a penalty.

Czech and Czarnas [27] proposed a pC/KS/MPSS co-operative multi-thread parallel simulated annealing implemented on a master-slave platform. The master sent the initial solution to the salves. It was also in charge of controlling the annealing temperature schedule, collecting the best local solution from each slave after $n^2$ iterations for each temperature level ($n$ was the number of customers) and updating the global best solution. Each slave run a simulated annealing algorithm with the same parameters. Each slave $j$ co-operated with slaves $j-1$ and $j+1$ (slave 1 co-operated with slave 2 only) by exchanging best solutions. Co-operation was triggered every $n$ iterations. Computational tests with few (five) processors showed good performance, in terms of solution quality, compared to the best-known solutions of the Solomon benchmarks.

Berger and Berkaoui [9] presented a low-level parallel hybrid genetic method that used two population. The first aimed to minimize the total traveled distance, while the second aimed to minimize the violation of the time window constraints. A different fitness function was associated with each population. A master-slave platform was applied, where the master controlled the execution of the algorithm and coordinated the genetic operations. The slave concurrently executed the reproduction and mutation operators. Computational tests were conducted on a cluster of heterogeneous machines (19 computers). The authors compared their algorithm to the best-known methods in the literature for Solomon's benchmark. Their results showed that the proposed technique was very competitive.

Polacek *et al.* [60] focused on parallel algorithms for the multi-depot VRPTW, starting from a Variable Neighborhood Search (VNS) meta-heuristic proposed earlier on [61]. The authors studied two approaches, both based on co-operation and asynchronous exchanges through a central memory. The first approach implemented full VNS threads, each searching through a limited number of neighborhoods. The VNS threads collaborated through exchanges of best solutions via the central memory. Each VNS sent its best solutions. When the overall best was improved, it was broadcasted to all. In the second approach, the VNS threads sent their best solutions to the central memory (functioning as a "master") at regular intervals (number of iterations). The objective was to reproduce the behavior of the sequential method only faster. Performance was good, best known solutions being reached or improved. The first, full co-operation methods performed best due, in particular, to its higher adaptability to the problem instance.

Gehring and Homberger [37, 38] proposed a pC/C/MPDS co-operative parallel strategy where concurrent searches were performed with differently configured two-phase meta-heuristics. The first phase tried to minimize the number of vehicles by using an evolutionary meta-heuristic, while the second phase aimed to minimize the total traveled distance by means of a tabu search. The parallel meta-heuristic was initiated on different threads with different starting points and values for the search time available for the first and second search phases. Threads co-operated by exchanging solutions asynchronously through a "master" process according to the central-memory concept. Notice that this is different from most evolutionary-based parallel meta-heuristics proposed in the literature. For now, this approach has produced, on average, the best solutions for the Solomon problems with respect to, first, the number of vehicles and, second, the total distance. Results were also presented on larger instances, generated similarly to the original Solomon problems, but varying in size from 200 to 1000 customers. It is worth mentioning, however, that this method is rather time consuming compared to other meta-heuristics, tabu search in particular.

Rochat and Taillard [71] proposed what may be considered as the first fully developed adaptive memory-based approach for the VRPTW. The adaptive memory contained tours of good solutions identified by the tabu search threads. The tours were ranked according to attribute values, including the objective values of their respective solutions. Each tabu search process then probabilistically selected tours in the memory, constructed an initial solution, improved it, and returned the corresponding tours to the adaptive memory. Despite the fact that it used a rather simple tabu search, this method produced many new best results at publication time. Taillard *et al.* [76] and Badeau *et al.* [5] refined this method by enriching the neighborhood and the intensification phase and by adding a post-optimization procedure. A similar approach was used with good results by Schulze and Fahle [72]. The routes generated by the tabu search threads were collected in a pool, and were recombined by solving a set covering heuristic whenever a new solution was needed. Badeau *et al.* [5] also reported that their method run significantly faster when using more search processes than the number of available processors, because this allowed to overcome the bottlenecks created when several threads attempted to access the memory simultaneously. Furthermore, computational evidence showed that running a search thread concurrently with the adaptive memory management procedure on the same processor was not a good idea, because it

contributed to block the access to the memory.

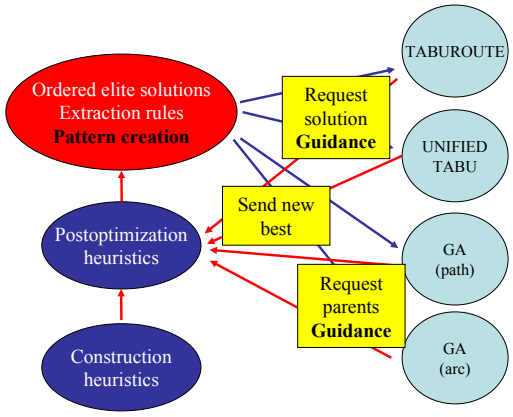


Figure 4: Central Memory with Guidance for the VRPTW

Le Bouthiller and Crainic [50] and Le Bouthiller, Crainic, and Kropf [51] aimed to study central memory co-operative mechanisms enhanced with strategies to guide the global search. Le Bouthiller and Crainic proposed a central memory pC/C/MPDS co-operative parallel method for the VRPTW based on the mechanism presented at Section 4 and illustrated in Figure 4. The co-operation involved two tabu search methods that perform well sequentially, the Unified Tabu of Cordeau, Laporte, and Mercier [15] and Taburoute of Gendreau, Hertz, and Laporte [40], two simple evolutionary algorithms with order and edge recombination crossovers, respectively, as well as a number of post-optimization methods (2-opt, 3-opt, or-opt, and ejection chains) that were used to reduce the number of vehicles and the total traveled distance. Four simple construction algorithms were used to provide initial solutions to the population. The threads shared information about their respective good solutions identified so far. When a thread improved the solution, it sent it to the post-optimization algorithms present in the central memory. These solutions were considered in-training until they were post-optimized and identified as adult solutions. The pool of solutions formed an elite population from which the independent procedures required solutions when needed. The Unified Tabu requested a new solution at regular intervals, while Taburoute did so at diversification time. The adult solutions in memory formed the population for the genetic operators. This algorithm, without any calibration or tailoring, proved to be competitive with the best meta-heuristics of its day.

The goal of Le Bouthiller, Crainic, and Kropf [51] was to improve upon this simple co-operating scheme by extracting new knowledge from the information exchanged, in order to guide the individual threads and, hopefully, yield a more efficient global search. Moreover, the authors aimed for a guidance mechanism independent of particular features of the problem class at hand, such as the routes in vehicle routing problems. They selected therefore to work with one of the atomic elements of the problem: the arc.

The basic idea was that an arc that appears often in good solutions and less frequently in bad solutions may be worthy of inclusion in a tentative solution, and vice versa. To

implement this idea, the authors considered the frequency of inclusion of arcs in three subsets of solutions in the pool, the elite (e.g., the 10% best), average (between the 10% and 90% best), and worst (the last 10%) groups of solutions. An arc with a high frequency in a given group signals that the meta-heuristics participating to the co-operation have often produced solutions that include that arc. Patterns of arcs were then defined, representing subsets of arcs with similar frequency of inclusion or not in particular population groups. Guidance was obtained by transmitting arc patterns to the individual threads indicating whether the arcs in the pattern should be "fixed" to intensify the search or, on the contrary, they should be prohibited to diversify the search ("fix" and "prohibit" were performed by using the patters to bias the selection of arcs during moves or reproduction). The computing time allocated to the co-operative method was divided into four phases: Two phases of diversification at the beginning to broaden the search, followed by two intensification phases to focus the search around promising regions. Figure 4 illustrates the flow of information and guidance indications.

Experiments were carried out with this pC/KC/MPDS co-operative method on the standard set of 100-customer test problems proposed by Solomon [74], as well as on the extended set produced by Homberger and Gehring [47] with 300 problem instances that vary from 200 to 1000 customers. Runs of 12 min wall-clock time were performed by the co-operative meta-heuristic for each of the 100 city problems. Longer running times, equal to those reported by Homberger and Gehring were allowed for the larger problem instances. These times go up to 50 min wall-clock time for the 1000-city problem. Comparisons were carried on with the best performing methods and the results were very good. In linear speed up and without any calibration, the guided co-operative search was globally performing on a par with the best. This is very encouraging, because patterns of attributes may be constructed for many problem classes, independently of particular solution structures.

## 5.3 Dynamic Problems

Gendreau, Laporte, and Semet [41] addressed the deployment problem for a fleet of emergency vehicles and proposed a 1C/KS/MPSS parallel tabu search method based on domain decomposition. In this problem, when a call is received, an ambulance is assigned to it according to relatively simple dispatching rules. Then, the remaining available ambulances can be relocated to other waiting sites to provide a better coverage of the expected demand. The parallel algorithm was based on a pure master-slave scheme. The master managed global data structures with pre-calculated information on each ambulance and sent the relocation problems to the slaves. The time allotted to slaves was controlled by fixing the number of iterations in the tabu search. Computational tests showed high solution quality as indicated by territory coverage measures.

Attanasio *et al.* [3] addressed the multi-vehicle dial-a-ride-problem and proposed two multi-thread tabu search parallel strategies following pC/KS/SPDS and pC/KS/MPSS frameworks. In the former, each processor run a different tabu search strategy from the same

initial solution. Once a processor found a new best solution, it broadcast it. Re-initialized searches were then launched. Every $\kappa$ iterations, a diversification procedure was applied to the first half of the processors, while an intensification was run on the remaining half. The pC/KS/MPSS strategy consisted in running a tabu search algorithm from different starting points. Each processor run the same tabu search algorithm with the best known parameter settings. Every $\eta$ iterations, processors exchanged information in order to perform a diversification procedure. According to the computational results, both strategies outperformed the sequential tabu search of Cordeau and Laporte [16].

Gendreau *et al.* [39] proposed a co-operative multi-thread parallel tabu search method for real-time routing and vehicle dispatching problems. The problem was motivated by courier services where customer requests for the transportation of small items must be accommodated in real-time and incorporated into the current planned routes of a fleet of vehicles. Due to the presence of soft time constraints for servicing a customer, the problem was modeled as an Uncapacitated Vehicle Routing Problem with Soft Time Windows. The objective function to be minimized related to the total distance traveled (or total travel time) for servicing the customers plus penalties for lateness at customer locations. The authors followed the co-operative adaptive memory approach championed by Taillard *et al.* [76] and Badeau *et al.* [5]. The dynamic problem was addressed as a series of static problems, a new one being defined each time a new request was received. A two-level parallelization scheme was proposed to implement this problem-solving framework. At the first level, a pC/KC/MPSS co-operating adaptive memory scheme was implemented. At the second level, each individual tabu search thread benefited of the work of several slave processors and the route decomposition of Taillard [75] was implemented. The results showed that the proposed procedure provides substantial benefits over simpler dispatching approaches.

# 6 Perspectives

We have presented a survey of exact and meta-heuristic parallel methods applied to vehicle routing problems. Although not necessarily comprehensive, it includes the major contributions and trends in the field, most of which have been proposed from 2000 onwards.

We found few contributions targeting parallel exact methods for VRP and variants. The proposed branch-and-price parallelization schemes are not very sophisticated yet, but we expect them to represent initial steps on what promises to be a very fruitful research road.

The parallel meta-heuristic field is much richer, of course, as illustrated by the number of contributions and by the increasing variety of the methodologies used. This richness notwithstanding, the survey points out that not all VRP variants have been addressed with comparable fervor. Indeed, many important topics have seen only a few of contributions, if at all. Moreover, even for topics for which the number of contributions is larger, these are not evenly distributed among meta-heuristic classes. Interesting research avenues and

promising developments may thus go unexplored, and appropriate tools may be missing in some areas. It should be a challenge of the profession to explore as comprehensively as possible as many problem variants, search methodologies, and parallelization strategies as possible. While taking up this challenge, one should make sure that methods are compared across methodological approaches and that such comparisons are performed fairly, that is, all algorithmic developments are at the same level of sophistication.

To sum up the observations relative to parallel meta-heuristics, it appears that methods based on asynchronous co-operation mechanisms display the most interesting performance, independently of the methodology used in the initial sequential method. This conclusion is strongly supported by the results obtained by multi-thread co-operative strategies. It also appears that one can build simple but meaningful statistics and indicators to learn from the solutions already explored and to globally guide the search. This research direction is at the very beginning and should yield interesting results.

To conclude, parallel exact and meta-heuristic solution methods offer versatile, robust, and powerful tools to address large and complex vehicle routing problems. Many fascinating research avenues are still open for investigation, however. Other than those indicated above, we may mention applying more sophisticated branch-and-bound parallelization strategies to VRP variants, studying co-operation mechanisms based on multi-level concepts, combining branch-and-bound and meta-heuristic threads within a co-operative search framework, developing enhanced guidance mechanisms, and applying these solution methods to new problem classes. We hope that this paper has contributed to illustrate these opportunities and challenges.

# Acknowledgments

Funding for this project has been provided by the Natural Sciences and Engineering Council of Canada. The author wishes to thank three anonymous referees. Their comments have contributed to make a better paper.

# References

[1] Alba, E., editor. *Parallel Metaheuristics. A New Class of Algorithms.* John Wiley & Sons, Hoboken, NJ, 2005.

[2] Alba, E. and Dorronsoro, B. Solving the Vehicle Routing Problem by Using Cellular Genetic Algorithms. In Gottlieb, J. and Günther, R.R., editors, *Evolutionary Computation in Combinatorial Optimization, 4th European Conference, EvoCOP 2004, Coimbra, Portugal, April 5-7, 2004*, volume 3004 of *Lecture Notes in Computer Science*, pages 11–20. Springer-Verlag, Heidelberg, 2004.

[3] Attanasio, A., Cordeau, J.F., Ghiani, G., and Laporte, G. Parallel Tabu Search Heuristics for the Dynamic Multi-Vehicle Dial-a-ride Problem. *Parallel Computing*, 30:377–387, 2004.

[4] Azencott, R. *Simulated Annealing Parallelization Techniques.* John Wiley & Sons, New York, NY, 1992.

[5] Badeau, P., Gendreau, M., Guertin, F., Potvin, J.-Y., and Taillard, É.D. A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. *Transportation Research Part C: Emerging Technologies*, 5(2):109–122, 1997.

[6] Barr, R.S. and Hickman, B.L. Reporting Computational Experiments with Parallel Algorithms: Issues, Measures, and Experts Opinions. *ORSA Journal on Computing*, 5(1):2–18, 1993.

[7] Benkner, S., Doerner, K., Hartl, R.F., Kiechle, G., and Lucka, M. Communication Strategies for Parallel Cooperative Ant Colony Optimization on Clusters and Grids. In *Complimentary Proceedings of PARA04*, pages 3–12, 2004.

[8] Berger, J. and Barkaoui, M. Hybrid Genetic Algorithm for the Capacitated Vehicle Routing Problem. In Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Standish, R.K., Kendal, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A.C., Dowsland, K.A., Jonoska. N., and Miller, J.F., editors, *Genetic and Evolutionary Computation - GECCO 2003, Genetic and Evolutionay Computation Conference, Chicago, IL, USA, July 12-16, 2003, Proceedings, Part I*, volume 2723 of *Lecture Notes in Computer Science*, pages 646–656. Springer-Verlag, Heidelberg, 2003.

[9] Berger, J. and Barkaoui, M. A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows. *Computers & Operations Research*, 31(12):2037–2053, 2004.

[10] Bertsekas, D.P. and Tsitsiklis, J.N. *Parallel and Distributed Computation, Numerical Methods.* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[11] Bullnheimer, B., Hartl, R., and Strauß, C. An Improved Ant System Algorithm for the Vehicle Routing Problem. *Annals of Operations Research*, 89:319–328, 1999.

[12] Bullnheimer, B., Kotsis, G., and Strauß, C. Parallelization Strategies for the Ant System. *Applied Optimization*, 24:87–100, 1998.

[13] Cantú-Paz, E. A Survey of Parallel Genetic Algorithms. *Calculateurs Parallèles, Réseaux et Systèmes répartis*, 10(2):141–170, 1998.

[14] Christofides, N., Mingozzi A., and Toth, P. The Vehicle Routing Problem. In N. Christofides, Mingozzi A., P. Toth, and C. Sandi, editors, *Combinatorial Optimization*, pages 315–338. John Wiley, New York, 1979.

[15] Cordeau, J.-F., Laporte, G., and Mercier, A. A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. *Journal of the Operational Research Society*, 52:928–936, 2001.

[16] Cordeau, J.F. and G. Laporte, G. A Tabu Search Heuristics for the Static Multi-vehicle Dial-a-ride Problem. *Transportation Research Part B: Methodological*, pages 579–594, 2003.

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

[18] Crainic, T.G., Gendreau, M., and Potvin, J.-Y. Parallel Tabu Search. In Alba, E., editor, *Parallel Metaheuristics*, pages 298–313. John Wiley & Sons, Hoboken, NJ, 2005.

[19] Crainic, T.G., Le Cun, B., and Roucairol, C. Parallel Branch and Bound Algorithms. In EL-Ghazali Talbi, editor, *Parallel Combinatorial Optimization*, pages 1–28. John Wiley & Sons, New York, 2006.

[20] Crainic, T.G., Li, Y., and Toulouse, M. A First Multilevel Cooperative Algorithm for the Capacitated Multicommodity Network Design. *Computers & Operations Research*, 33(9):2602–2622, 2006.

[21] Crainic, T.G. and Nourredine, H. Parallel Meta-Heuristics Applications. In Alba, E., editor, *Parallel Metaheuristics*, pages 447–494. John Wiley & Sons, Hoboken, NJ, 2005.

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

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

[24] Crainic, T.G., Toulouse, M., and Gendreau, M. Parallel Asynchronous Tabu Search for Multicommodity Location-Allocation with Balancing Requirements. *Annals of Operations Research*, 63:277–299, 1995.

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

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

[27] Czech, Z.J. and Czarnas, P. Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows. In *10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, pages 376–383, 2002.

[28] Dai, C., Li, B., and Toulouse, M. A Multilevel Cooperative Tabu Search Algorithm for the Covering Design Problem. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 2007.

[29] Deb, K., Pratab, A., Agrawal, S., and Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[30] Doerner, K., Hartl, R.F., Kiechle, G., Lucka, M., and Reimann, M. Parallel Ant Systems for the Capacitated Vehicle Routing Problem. In Gottlieb, J. and Raidl, G.R., editors, *Evolutionary Computation in Combinatorial Optimization: 4th European Conference, EvoCOP 2004, Proceedings*, volume 3004 of *Lecture Notes in Computer Science*, pages 72–83. Springer-Verlag, Berlin, 2004.

[31] Doerner, K.F., Hartl, R.F., Benkner, S., and Lucka, M. Cooperative Savings based Ant Colony Optimization - Multiple Search and Decomposition Approaches. *Parallel Processing Letters*, 16(3):351–369, 2006.

[32] Doerner, K.F., Hartl, R.F., and Lucka, M. A Parallel Version of the D-Ant Algorithm for the Vehicle Routing Problem. In Vajtersic, M., Trobec, R., Zinterhof, P., and Uhl, A., editors, *Parallel Numerics '05*, pages 109–118. Springer-Verlag, New York, NY, 2005.

[33] Dorigo, M. and Stuetzle, T. The Ant Colony Metaheuristic. Algorithms, Applications, and Advances. In F. Glover and G. Kochenberger, editors, *Handbook in Metaheuristics*, pages 251–285. Kluwer Academic Publishers, Norwell, MA, 2003.

[34] Drummond, L.M.A., Ochi, L.S., and Vianna, D.S. A Parallel Hybrid Evolutionary Metaheuristic for the Period Vehicle Routing Problem. In Rolin, J., editor, *International Workshop on Formal Methods for Parallel Programming: Theory and Applications (FMPPTA'99)*, volume 1586 of *Lecture Notes in Computer Science*, pages 183–191. Springer-Verlag, Heidelberg, 1999.

[35] Drummond, L.M.A., Ochi, L.S., and Vianna, D.S. An Asynchronous Parallel Metaheuristic for the Period Vehicle Routing Problem. *Future Generation Computer Systems*, 17(4):379–386, 2001.

[36] Garcia, B.L., Potvin, J.-Y., and Rousseau, J.M. A Parallel Implementation of the Tabu Search Heuristic for Vehicle Routing Problems with Time Window Constraints. *Computers & Operations Research*, 21(9):1025–1033, 1994.

[37] Gehring, H. and Homberger, J. A Parallel Two-Phase Metaheuristic for Routing Problems with Time Windows. *Asia-Pacific Journal of Operational Research*, 18(1):35–47, 2001.

[38] Gehring, H. and Homberger, J. Parallelization of a Two-Phase Metaheuristic for Routing Problems with Time Windows. *Journal of Heuristics*, 8:251–276, 2002.

[39] Gendreau, M., Guertin, F., Potvin, J.-Y., and Taillard, É.D. Tabu Search for Real-Time Vehicle Routing and Dispatching. *Transportation Science*, 33(4):381–390, 1999.

[40] Gendreau, M., Hertz, A., and Laporte, G. A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science*, 40:1276–1290, 1994.

[41] Gendreau, M., Laporte , G., and Semet, F. A Dynamic Model and Parallel Tabu Search Heuristic for Real-time Ambulance Relocation. *Parallel Computing*, 27(12):1641–1653, 2001.

[42] Glover, F. and Laguna, M. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, 1997.

[43] Golden, B.L., Wasil, E.A., Kelly, J.P., and Chao, I.M. Metaheuristics in Vehicle Routing. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 33–56. Kluwer Academic Publishers, Norwell, MA, 1998.

[44] Greening, D.R. Asynchronous Parallel Simulated Annealing. *Lectures in Complex Systems*, 3:497–505, 1990.

[45] Greening, D.R. Parallel Simulated Annealing Techniques. *Physica D*, 42:293–306, 1990.

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

[47] Homberger, J. and Gehring, H. Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows. *INFOR*, 37:297–318, 1999.

[48] Jozefowiez, N., Semet, F., and Talbi, E.-G. Parallel and Hybrid Models for Multi-objective Optimization: Application to the Vehicle Routing Problem. In J.M. Guervos et al., editor, *PPSN VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 271–280. Springer-Verlag, Berlin, 2002.

[49] Jozefowiez, N., Semet, F., and Talbi, E.-G. Parallel and Hybrid Models for Multi-objective Optimization: Application to the Vehicle Routing Problem. In E.-G. Talbi et al., editor, *EA 2005*, volume 2871 of *Lecture Notes in Computer Science*, pages 131–142. Springer-Verlag, Berlin, 2006.

[50] Le Bouthillier, A. and Crainic, T.G. A Cooperative Parallel Meta-Heuristic for the Vehicle Routing Problem with Time Windows. *Computers & Operations Research*, 32(7):1685–1708, 2005.

[51] Le Bouthillier, A., Crainic, T.G., and Kropf, P. A Guided Cooperative Search for the Vehicle Routing Problem with Time Windows. *IEEE Intelligent Systems*, 20(4):36–42, 2005.

[52] Lin, S.-C., Punch, W., and Goodman, E. Coarse-Grain Parallel Genetic Algorithms: Categorization and New Approach. In *Sixth IEEE Symposium on Parallel and Distributed Processing*, pages 28–37. IEEE Computer Society Press, 1994.

[53] Moreno-Pérez, J.A., Hansen, P., and Mladenović, N. Parallel Variable Neighborhood Search. In Alba, E., editor, *Parallel Metaheuristics*, pages 247–266. John Wiley & Sons, Hoboken, NJ, 2005.

[54] Mühlenbein, H. Parallel Genetic Algorithms in Combinatorial Optimization. In O. Balci, R. Sharda, and S. Zenios, editors, *Computer Science and Operations Research: New Developments in their Interface*, pages 441–456. Pergamon Press, New York, NY, 1992.

[55] Ochi, L.S., Vianna, D.S., Drummond, L.M.A., and Victor, A.O. A Parallel Evolutionary Algorithm for the Vehicle Routing Problem with Heterogeneous Fleet. *Future Generation Computer Systems*, 14(3):285–292, 1998.

[56] Oduntan, I.O., Toulouse, M., Baumgartner, R., Somorjai, R., and Crainic, T.G. A Multilevel Tabu Search Algorithm for the Feature Selection Problem in Biomedical Data Sets. *Computers & Mathematics with Applications*, 2006.

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

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

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

[60] Polacek, M., Benkner, S., Doerner, K.F., and Hartl, R.F. A Cooperative and Adaptive Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows. Working paper, Institute of Management Science, University of Vienna, Vienna, Austria, 2006.

[61] Polacek, M., Hartl, R.F., , Doerner, K.F., and Reimann, M. A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows. *Journal of Heuristics*, 10(6):613–627, 2004.

[62] Prins, C. A Simple and Effective Evolutionary Algorithm for the Vehicle Routing Problem. *Computers & Operations Research*, 31(12):1985–2002, 2004.

[63] Ralphs, T.K. Parallel Branch and Cut for Capacitated Vehicle Routing. *Parallel Computing*, 29:607–629, 2003.

[64] Ralphs, T.K. Parallel Branch and Cut Algorithms. In E.-G. Talbi, editor, *Parallel Combinatorial Optimization*, pages 53–101. Wiley-Interscience, Wiley & Sons, Hoboken, NJ, 2006.

[65] Ralphs, T.K., L. Ladány, and Saltzman, M.J. Parallel Branch, Cut, and Price for Large-Scale Discrete Optimization. *Mathematical Programming*, 98(1-3):253–280, 2003.

[66] Ralphs, T.K., L. Ladány, and Saltzman, M.J. A Library Hierarchy for Implementing Scalable Parallel Search Algorithms. *Journal of Parallel and Distributed Computing*, 37:207–212, 2004.

[67] Ram, D.J., Sreenivas, T.H., and Subramaniam, K.G. Parallel Simulated Annealing Algorithms. *Journal of Parallel and Distributed Computing*, 37:207–212, 1996.

[68] Rego, C. and Roucairol, C. A Parallel Tabu Search Algorithm Using Ejection Chains for the VRP. In I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory & Applications*, pages 253–295. Kluwer Academic Publishers, Norwell, MA, 1996.

[69] Reimann, M., Doerner, K., and Hartl, R. D-Ants: Savings Based Ants Divide and Conquer the Vehicle Routing Problem. *Computers & Operations Research*, 31(4):563–591, 2004.

[70] Reimann, M., Stummer, M., and Doerner, K. A Savings Based Ants System for the Vehicle Routing Problem. In Langton, C., Cantú-Paz, E., Mathias, K.E., Roy, R., Davis, L., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M.A., Schultz, A.C., Miller, J.F., Burke, E.K., and Jonoska. N., editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, July 9-13, 2002*, pages 1317–1326. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2002.

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

[72] Schulze, J. and Fahle, T. A Parallel Algorithm for the Vehicle Routing Problem with Time Window Constraints. *Annals of Operations Research*, 86:585–607, 1999.

[73] Shonkwiler, R. Parallel Genetic Algorithms. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 199–205. Morgan Kaufmann, San Mateo, CA, 1993.

[74] Solomon, M.M. Time Window Constrained Routing and Scheduling Problems. *Operations Research*, 35(2):254–265, 1987.

[75] Taillard, É.D. Parallel Iterative Search Methods for Vehicle Routing Problems. *Networks*, 23:661–673, 1993.

[76] Taillard, É.D., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows. *Transportation Science*, 31(2):170–186, 1997.

[77] Talbi, E.-G., editor. *Parallel Combinatorial Optimization*. Wiley-Interscience, Wiley & Sons, Hoboken, NJ, 2006.

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

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

[80] Voß, S. Tabu Search: Applications and Prospects. In D.-Z. Du and P.M. Pardalos, editors, *Network Optimization Problems*, pages 333–353. World Scientific Publishing Co., Singapore, 1993.