CIRRELT

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

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

An Integrative Cooperative Search Framework for Multi-Decision-Attribute Combinatorial Optimization

Nadia Lahrichi
Teodor Gabriel Crainic
Michel Gendreau
Walter Rei
Gloria Cerasela Crişan
Thibaut Vidal

August 2012

CIRRELT-2012-42

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
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

An Integrative Cooperative Search Framework for Multi-Decision-Attribute Combinatorial Optimization

Nadia Lahrichi[1,2], Teodor Gabriel Crainic[1,3,*], Michel Gendreau[1,2], Walter Rei[1,3], Gloria Cerasela Crişan[4], Thibaut Vidal[1,5]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

[3] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[4] Vasile Alecsandri University of Bacau, Calea Marasesti 157, Bacau, 600115, Romania

[5] Laboratoire d'optimisation des systèmes industriels (LOSI), Université de Technologie de Troyes, 12, rue Marie Curie, B.P. 2060, 10010 Troyes, Cedex, France

Abstract. We introduce the Integrative Cooperative Search (ICS), a multi-thread cooperative search method for multi-attribute combinatorial optimization problems. ICS musters the combined capabilities of a number of independent exact or meta-heuristic solution methods. A number of these methods work on sub-problems defined by suitably selected subsets of decision-set attributes of the problem, while others combine the resulting partial solutions into complete ones and, eventually, improve them. All these methods cooperate through an adaptive search-guidance mechanism, using the central-memory cooperative search paradigm. Extensive numerical experiments explore the behavior of ICS and how the interest of the method through an application to the multi-depot, periodic vehicle routing problem, for which ICS improves the results of the current state-of-the-art methods.

Keywords: Multi-attribute combinatorial optimization, integrative cooperative search, decision-set decomposition, multi-depot, periodic vehicle routing

Acknowledgements. While working on this project, T.G. Crainic was the Natural Sciences and Engineering Research Council of Canada (NSERC) Industrial Research Chair in Logistics Management, ESG UQAM, N. Lahrichi and G.C. Crişan were postdoctoral fellows with the Chair, and M. Gendreau was the NSERC/Hydro-Québec Industrial Research Chair on the Stochastic Optimization of Electricity Generation, MAGI, École Polytechnique. Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair, Collaborative Research and Development, and Discovery Grant programs, by our partners CN, Rona, Alimentation Couche-Tard, la Fédération des producteurs de lait du Québec, and the Ministry of Transportation of Québec, and by the Fonds de recherche du Québec - Nature et technologies (FRQNT) through its Team Research Project program.

---

* Corresponding author: TeodorGabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2012

1 Introduction

Combinatorial optimization problems prominently appear in many theoretical and real-life settings. A large number of methodological developments targeted these problems proposing exact, heuristic, and meta-heuristic solution methods. Parallel computing enhanced these optimization methods providing the means to accelerate the resolution process and, for meta-heuristics, to obtain higher-quality solutions for a broad range of problems (Crainic and Nourredine, 2005; Crainic and Toulouse, 2010).

Yet, although solution methods become more powerful, the combinatorial problems one faces grow continuously in size and difficulty, as defined by the number of interacting characteristics defining their feasibility structures and optimality criteria. Such increasingly larger sets of characteristics severely challenge our methodological capability to efficiently address the corresponding problem settings. Thus, the general approach when addressing such *multi-attribute*, also informally known as *rich*, problem settings is to either simplify them, or to sequentially solve a series of restricted ones where part of the overall problem might be fixed, ignored, or both (Section 2 further discusses the issue).

It is well-known, however, that such approaches lead to sub-optimal solutions. Moreover, one observes in many application settings, including vehicle routing, network design, and carrier service network design, the need to comprehensively address the associated combinatorial optimization formulations to simultaneously account for "all" relevant attributes. The current literature does not offer a satisfactory answer to this challenge in terms of methods able to efficiently address multi-attribute problem settings and provide good solutions. The goal of this paper is to contribute toward addressing this challenge.

We focus on decision-based characteristics, or *decision-set attributes*, i.e., the sets of decisions defining the particular problem setting, and introduce the *Integrative Cooperative Search* (*ICS*), a multi-thread cooperative search method for multi-attribute combinatorial optimization problems. ICS musters the combined capabilities of a number of independent solution methods, exact or meta-heuristic. A number of these methods work on sub-problems defined by suitably selected subsets of decision-set attributes of the problem, while others combine the resulting partial solutions into complete ones and, eventually, improve them. These methods cooperate through an adaptive search-guidance mechanism, using the central-memory cooperative search paradigm. Our goal is to present and discuss the ICS concept, its structure, main building blocks, and operating principles, as well as to present a proof-of-concept of its efficiency.

The main contributions of this paper are to: 1) Introduce and formally describe a new meta-heuristic solution framework for multi-attribute combinatorial optimization problems; ICS is general, flexible, and scalable in the number of attributes defining the problem at hand; 2) Define and exploit a functional decomposition of such problems along decisional attributes; 3) Show the interest of ICS through an application to a well-known

multi-attribute case, the *multi-depot, periodic vehicle routing problem* (*MDPVRP*), for which ICS improves the results of the current state-of-the-art methods, and discuss how to apply the methodology to other combinatorial optimization problem classes; 4) Experimentally examine the role and impact of various ICS components.

This paper is organized as follows. Section 2 discusses the motivation for our work, and identifies the sources of inspiration for the methodology we propose. Section 3 introduces the fundamental concepts underlining the Integrative Cooperative Search methodology, particularly the decision-set attribute-based decomposition and the ICS algorithmic structure. We illustrate these concepts and methods through an application to the MDPVRP in Section 4. Extensive experiments are presented and discussed in Section 5. We finally conclude.

2 Motivation and Inspiration

We aim to address extended versions of classical combinatorial-optimization problems, which are NP-hard in their basic forms. When real-world cases are considered, "new" characteristics have to be considered while searching for feasible solutions of high quality. These characteristics take usually the form of a broad set of conditions defining the solution feasibility or optimality, or both, and may generally be represented through sets of decision variables. The resulting set of decisions is then much broader than for the classical problem settings usually found in the literature, each new group of decisions compounding the difficulty of the problem and complicating the solution process, particularly when one aims to address them simultaneously.

Two examples to illustrate, selected from two major problem classes of significant methodological interest and widely encountered in actual applications. Network design aims to select among a set of possible facilities on arcs, nodes, or both, such that the demand for utilization of the resulting network is satisfied at minimum total cost accounting for the cost of both selecting the facilities and using the network. Many particular design and location-decision problem settings have been defined during the years, the vast majority focusing on the facility-selection and demand-flow decisions (Magnanti and Wong, 1984; Grötschel et al., 1995; Drezner and Hamacher, 2002; Crainic, 2000). Crainic et al. (2006) describe a realistic setting of a wireless network design problem, where the decision set includes not only decisions of the selection of base stations to cover a given territory, but also on selecting the number of antennae for each of these, as well as their height, power, tilt, and orientation.

A similar richness of attributes may be increasingly observed in the vehicle routing field (where the "rich" qualification was first linked to a VRP setting; Hartl et al., 2006). There exists an extensive literature on the Capacitated Vehicle Routing Problem

(CVRP), which aims to construct cost-efficient routes to deliver the demand of a given set of customers with a fleet of vehicles of limited capacity operating out of a unique depot (Toth and Vigo, 2002; Golden et al., 2008). An even greater volume of contributions address extensions of the CVRP reflecting the extreme variety of actual applications, including but not limited to multiple depots, vehicle fleets, or commodities, customer requirements for multi-period visits or within-period time windows, route restrictions on total distance or time, and so on and so forth (Vidal et al., 2012a). Several of these generalizations yielded problem classes of their own, the Vehicle Routing Problem with Time Windows (VRPTW) being probably the most well known of these.

Most generalizations found in the literature either add decisional features, e.g., selecting the number of antennae in addition of the location and number of stations in the wireless network design problem, or are concerned with adding characteristics to particular problem elements, e.g., the timing concerns in the VRPTW sequence of customer visits. While not making problems easier, the latter type of generalization may be addressed within particular algorithmic components, e.g., route generation and sequencing. Adding an extra characteristic to the definition of this group of problem elements might require a more involved algorithmic component (Vidal et al., 2011), but does not change fundamentally the nature of the problem.

Adding more decision sets to the problem, on the other hand, makes it significantly harder to address and, thus, the general approach when addressing such multi-attribute problems is to either simplify them, or to sequentially solve a series of particular cases, where part of the overall problem is fixed or ignored, or both (e.g. Golden et al., 2002; Hadjiconstantinou and Baldacci, 1998; Hartl et al., 2006; Homberger and Gehring, 1999, 2005). It is well-known that this leads to sub-optimal solutions. Moreover, one observes in many application settings, including vehicle routing, network design, and carrier service network design, the need to comprehensively address the corresponding combinatorial formulation accounting for "all" relevant decision-set attributes simultaneously.

Among the few exceptions to the state of the literature described above, we single out the *Multi-Depot Periodic VRP* (*MDPVRP*) problem class, with or without time windows, encompassing decisions on selecting patterns of multi-period visits for customers, as well as on assigning each customer to a depot for each of its selected visits. This multi-decision-set problem was considered until quite recently as difficult to address as a whole. New exact (Baldacci and Mingozzi, 2009; Baldacci et al., 2011) and meta-heuristic (Vidal et al., 2012c) contributions have re-defined the state-of-the-art for this class of problems. We have selected the MDPVRP to illustrate the performance of the methodology we propose to take advantage of this new set of best-known solutions (BKS).

So, how to proceed to build a method, which is general, scalable, and efficient, to address multi-attribute combinatorial optimization problems in a comprehensive manner? We were inspired by three major methodological concepts, decomposition, simplification,

and cooperative search.

Decomposition is a fundamental technique in mathematical programming and parallel/distributed computing. Proceeding through various strategies, e.g., projection and relaxation in mathematical programming, its main objective is to transform a difficult-to-address formulation into a number of much simpler ones. All such methods encompass three main mechanisms. The first defines how the problem is transformed and how derived sub-problems are specified. The second specifies the information exchanged among the sub-problems. The third brings together the results obtained working on the sub-problems to create complete solutions to the original problem and, eventually, continues the decomposition-based algorithm. A so-called master problem (actually, one of the sub-problems resulting from the decomposition) is performing this task in the context of mathematical-programming decomposition, as well as for low-level and most domain-decomposition parallel strategies (Crainic and Toulouse, 2010).

Simplification is and has always been widely used to transform difficult problems into settings easier to address. As part of the modeling component of operations research, and of all disciplines based on formal representations of the problems contemplated, it is indeed an indispensable instrument for problem solving. The challenge in using simplification is reaching the right equilibrium between an easy-to-address problem setting and a high-usefulness of results for decision making for the original problem.

Cooperative search (Crainic, 2005; Crainic and Toulouse, 2008, 2010) has emerged as one of the most successful meta-heuristic methodologies to address hard optimization problems (e.g., Crainic and Nourredine, 2005; Crainic, 2008). Described generally as a parallel strategy for meta-heuristics, cooperative multi-search is based on harnessing the "solving" capabilities of several solution methods through mechanisms to asynchronously share information during the course of addressing a given problem instance and, in the most advanced settings, to create new information out of the exchanged data. The nature of the information shared, how the sharing proceeds, as well as the global and local (i.e., at the level of each collaborating solution method) utilization of the exchanged and received information, respectively, are the main characteristics of cooperative-search strategies. We focus on a widely-used class of cooperative strategies where the asynchronous communications are generally triggered individually by the cooperating algorithms, taking the form of exchanges of solutions or elements of solutions, and proceeding through a common data repository, often referred to as *adaptive* or *central memory* (Rego, 2001; Le Bouthillier and Crainic, 2005; Le Bouthillier et al., 2005; Jin et al., 2012b; Cordeau and Maischberger, 2012; Groër and Golden, 2011). Notice that, while "central"-memory mechanisms are clearly adaptive, "adaptive memory" is still sometimes used in the original sense of gathering fragments of good solutions, which are then used to construct new search starting points (Rochat and Taillard, 1995; Badeau et al., 1997). We therefore use "central memory" in this paper.

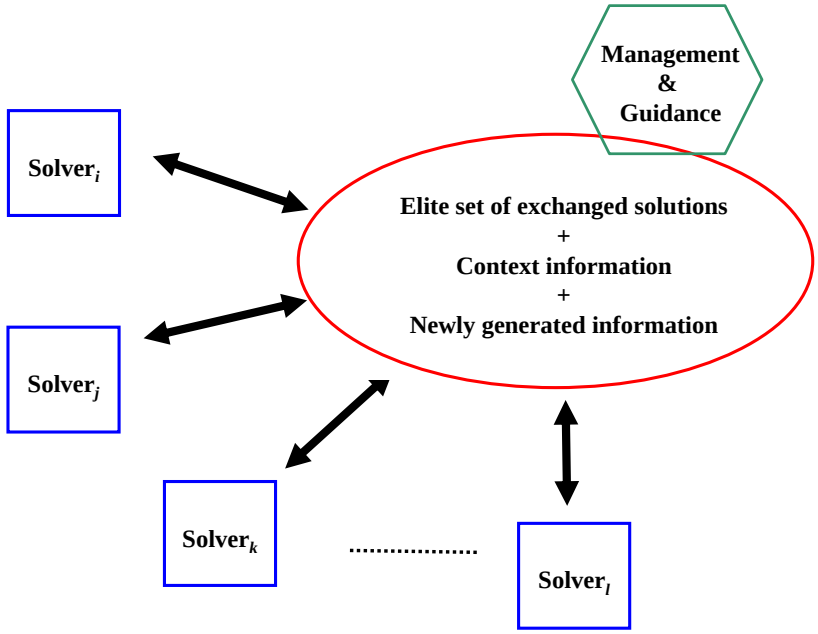

Figure 1: The Central-Memory Multi-Search Cooperative Scheme

Figure 1 illustrates the structure of a central-memory-based cooperative search algorithm. Several solution methods (four in the figure) participate to the collaborative search. As illustrated by the double-ended arrows, the flow of information exchange proceeds through the central-memory structure, a management-and-guidance module monitoring the traffic, managing the information deposited in the central memory and, eventually, using it to generate new relevant information, e.g., new solutions, performance measures on solution components, promising areas of the search space, and so on. Each method, which may be a meta-heuristic, an exact algorithm, or any other method and is simply identified as *solver*, thus makes available relevant information to the other participating methods by sending it to (depositing it into) the central memory. This information generally includes improved solutions or solution components, as well as so-called contextual information, e.g., performance measures and memories, to contribute building an image of the status of the search. The sending of information is triggered by the internal logic of the method, e.g., on identifying a new best solution or before a diversification phase, as is the request for new information sent to the central memory (the two events are usually coupled). The required information is used to inflect the search trajectory and may take the form of complete or partial solutions, or indications on regions of the search space to explore or to avoid.

Illustrations of successful application of these principles to problems with several attributes may be found, e.g., in Berger and Barkaoui (2004) and Crainic et al. (2006). The former addressed a vehicle routing problem with time windows by evolving two populations, one focusing on minimizing the total traveled distance, the other on minimizing the

violation of temporal requirements to generate feasible solutions. The latter addressed the wireless network design problem mentioned previously through a parallel cooperative meta-heuristic that uses Tabu Search (TS) solvers on limited subsets of attributes only, while a Genetic Algorithm (GA) amalgamates the partial solutions attained by the TS procedures into complete solutions to the initial problem.

3 ICS Fundamental Concepts and Structure

We now describe the fundamental concepts and structure of the proposed ICS methodology, which builds on the ideas and methodological developments reviewed in the previous section.

3.1 The ICS idea

The fundamental ICS concept combines the ideas of decomposition, intelligent simplification and opportunism, and cooperative search. We aim for a decomposition mechanism that yields simpler but meaningful problem settings for which one can either opportunistically use existing high-performing methodology or more easily develop new solution approaches. The resulting simpler problem settings together with the required mechanisms to reconstruct and enhance full solutions are then brought together into a cooperative-search framework aimed to address the initial formulation. Figure 2 illustrates the main elements of the methodology and their interconnection links.

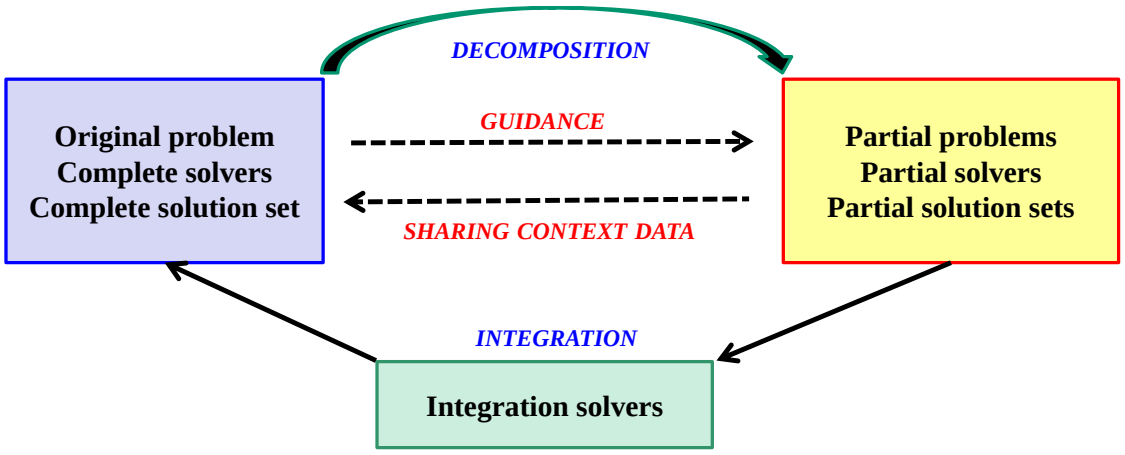

Figure 2: The ICS Methodology Idea

Decomposition is the first item that needs to be addressed, in particular, the problem

dimensions along which it is to be applied and the mechanisms to reconstruct complete solutions to the initial problem. Given our objective of opportunist simplification for an intelligent use within a cooperative- search framework, we propose a structural problem decomposition along sets of decisions variables.

Let $x \in \mathcal{X}$ be the set of decision variables of the problem at hand, and let δ be a set of indices identifying a particular subset of x. The *decision-set attribute decomposition* then identifies $|\Delta|$ sets of decision variables, and defines a *partial-problem* formulation $\Pi_\delta(\tilde{x_\delta})$ for each decision set $x_\delta,\ \delta \in \Delta$, by fixing the corresponding variables to suitably selected values $\tilde{x_\delta}$ (and appropriately adjusted constraints, if needed). Notice that this definition of decomposition does not change the dimensionality of the partial problems with respect to that of the original setting. Actually, any solution to any partial problem constructed according to this definition may be considered for the complete formulation.

The selection of the decision sets is specific to each application case. Thus, in particular, defining Δ according to spatio-temporal characteristics of the problem elements (e.g., node coordinates in a space-time network representation) yields the well known data-decomposition procedure used in a number of parallel and sequential solution methods. The choice for ICS is governed by the objective of opportunistic simplification indicated above and, thus, decision variables are clustered to yield known or identifiable optimization problem settings. Thus, for example, fixing the customer-to-depot assignments in the MDPVRP illustration of Section 4 yields a PVRP, while fixing the patterns for all customers yields a MDVRP.

Notice that the proposed decomposition does not require the sets $\delta \in \Delta$ to induce a partition of the feasible domain. Furthermore, with respect to the mathematical programming literature, the decision-set decomposition may be viewed as a multiple simultaneous projection (Geoffrion, 1970) performed through variable fixing. Further notice that there is no a priori limit on the number of sets. Actually, the decomposition process should aim for a "best" compromise between the ease of addressing each partial problem and the difficulty in building complete solutions to the original problem from the solutions to the partial problems.

Indeed, decomposition very often needs to be coupled with *integration* providing an efficient way to use the solutions obtained by addressing the partial problems to build a complete solution to the initial formulation. This functionality makes up the second main element of the ICS methodology. Integration is generally not an easy task, however. Integration solvers need to address three, possibly contradictory, challenges: 1) solution quality, 2) transmission of critical features (which may be incompatible) from the partial solutions, and 3) computational efficiency. Thus, the simple integrator consisting in transferring directly to the complete-solution set a solution (feasible or not) to a partial problem achieves the latter but fails in most cases to achieve the first two goals. Even when this simple integrator achieves solution quality, it is generally due to a good

partial solution and not through combining, and respecting critical features of, several partial solutions, as often required by decomposition schemes. More advanced methods are thus required and evolutionary methods, genetic algorithms and path relinking, in particular, have proved their flexibility and stability in combining solution characteristics to yield high-quality solutions, often at the price of higher computational efforts. A formal definition of integration problems for meta-heuristics is provided by Crainic et al. (2012).

We complete the ICS concept with a third main element, a purposeful-evolution mechanism geared to produce high-quality complete solutions while avoiding a heavy-handed control of the process (which has been shown to be unproductive for most meta-heuristics). We build upon the central-memory cooperative search meta-heuristic paradigm presented above, integrating a dynamically adaptive guidance mechanism based on monitoring the activities of the partial solvers, the partial and complete solutions produced, and the context information shared by the solvers. The ICS methodology, its structure, elements, and operational mechanisms are detailed next.

3.2 The ICS Method

ICS implements the decision-set attribute decomposition described in the previous section, and takes the form of a self-adaptive cooperative meta-heuristic concurrently evolving and combining several populations, one corresponding to solutions to the original problem, each of the others addressing specific dimensions of the problem resulting from the decision-set attributes used to decompose the problem.

Figure 3 illustrates the structure and components of ICS as inspired by the central-memory cooperative search meta-heuristic paradigm. Following the initial decomposition of the original problem into partial problems through decision-variable fixing as defined previously, each partial problem is addressed by one or several solution methods within a *Partial Solver Group* (*PSG*), two of which are illustrated in Figure 3. Concurrently with PSG activities, *integrators* select partial solutions from PSGs (represented by full arrows in the figure), combine them to create complete ones, which are then sent (short slashed arrow) to the *Complete Solver Group* (*CSG*). (All solutions are "complete" in our setting; we use the terms "partial" and "complete" solution, however, to indicate the solver group of origin.) The latter could enhance these solutions, when appropriate solvers are available, but its main task is to extract out of the complete-solution set the information required by the smooth but purposeful guidance of the partial and global searches. Dotted-line arrows represent the exchange of information supporting the cooperation under the supervision and lightly-handed guidance of the *Global Search Coordinator* (*GSC* - the Global M&G hexagonal box in the figure).

According to this decomposition-cooperation strategy, the problem is concurrently tackled by several solvers, which indirectly and asynchronously interact through a two-

layer central-memory and guidance mechanism. It is noteworthy that the representation of solutions for all solvers and in all central memories is identical no matter the particular partial or complete problem addressed or the specific solution method used. This characteristic, derived from the choice of fixing rather than eliminating variables when defining partial problems, facilitates communications, information extraction and knowledge creation from exchanged solutions, as well as the development of generic operators and solvers. The cooperation is built on these indirect exchanges through collections of elite solutions, communications being triggered either by the internal logic of each (partial) solver deciding when to send its "good" (e.g., just improved current best) solutions to the central memory or to request new solutions from the same population, or by the reaction of a local or global search coordinator monitoring the status of the memories and the search trajectory.

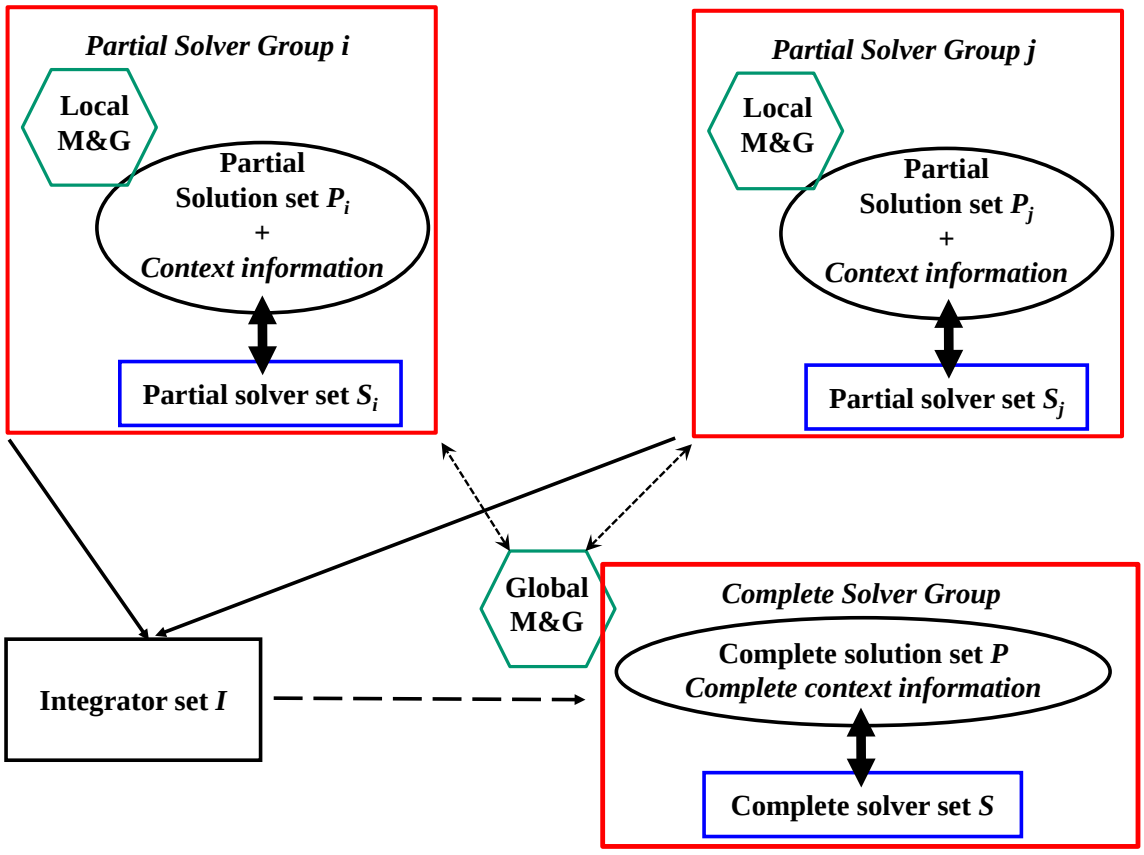

Figure 3: The Integrative Cooperative Search Structure

This framework ensured a high search efficiency, as solvers never interact directly, and has been shown to display very good performance in terms of solution quality, speed, and robustness in its "classical" central-memory incarnation (see the surveys of, e.g., Crainic and Nourredine, 2005; Crainic, 2008). ICS expands this concept to include Partial Solvers and Integrators into the cooperation, as well as a richer set of tasks for the Search Coordinators. We now examine each component in more details.

Partial Solver Group. Each PSG investigates a partial problem obtained through the decision-set decomposition procedure. It thus focuses on a particular subset of the original decision variables, the values of the other ones being fixed, directly or indirectly, through communications with the Global Search Coordinator. The PSG i is composed of a set of *Partial Solvers* $\mathcal{S}_i$ that work with complete solutions, but may modify only the values for the decision subset they are assigned to. Partial Solvers construct $\mathcal{P}_i$, the corresponding set (population) of elite *partial solutions*.

Each PSG operates according to the central-memory cooperative search paradigm. Thus, Partial Solvers may be simple constructive methods (providing initial solutions), metaheuristics or exact methods tailored to the subset of attributes they are dedicated to, or post-optimization methods. The type of solvers depends only on the available methods to efficiently address the particular problem. The number of solvers depends on the availability of appropriate solution methods and the choice of the analyst. ICS does not impose any actual limit, but the usual concerns related to an efficient implementation of parallel methods (e.g., computer architecture, communication protocol and latency, programming language, etc.) also apply in this context.

Communications and exchanges among Partial Solvers are performed through the central-memory mechanism made up of the population of elite solutions, context information, and the *Local Search Coordinator* (*LSC*) providing supervision, management, communications, and guidance functionalities. The actual Partial Solvers involved in cooperation and the corresponding cooperation mechanism are application specific and, thus, so is the local context information. The latter may consist in simple pointers to the best solutions and a relative order of the elements in the population, or may be enriched with memories related to, e.g., solution elements, solutions, performance measures for the Partial Solvers involved, and partial solution structures (see, e.g., Le Bouthillier et al., 2005; Jin et al., 2012a), to be used in guiding the Partial Solvers and the search trajectory of the group.

Communications between the central memory and the Partial Solvers is generally initiated by the latter to deposit or request, or both, solutions and context information. Guidance instructions issued by the local or global search coordinators may reverse this general policy. Indeed, the LSC may include functionalities to monitor the performance of the Partial Solvers and act when this performance becomes locally unsatisfactory (e.g., a given Partial Solver did not send any solution for a given time or all its latest solutions received by the LSC are weak compared to the best solutions in memory). The LSC could then, for example, force a Partial Solver to restart from a suitable solution in memory, modify the search parameters, or even change the solution method.

Similar behavior could also be triggered by messages from the GSC that monitors the progress of the global search. We discuss in more depth the role of the GSC later in this section, but want to emphasize that a thorough exploration of the original problem

solution space would often require the modification of the focus of particular Partial Solver Groups. The modification will generally proceed through changes to the $\tilde{x}_\delta$ values defining the search space of given $\Pi_\delta(\tilde{x}_\delta)$ PSGs, but could also involve the definition of the Δ partition. This modification is communicated to the corresponding LSC that, in turn, will instruct Partial Solvers, as well as work on the solutions and context information (initialize or modify values) in memory.

Integrators select solutions in the populations of one or more PSGs, combine them to yield complete solutions, and transmit some or all of these new solutions to the Complete Solver Group. Integrators are defined by the particular rules and procedures used to perform these tasks. Similarly to the Partial Solvers, Integrators can be very simple procedures selecting partial solutions to pass directly to the CSG, or comprehensive exact or heuristic solution methods.

Integrators play an essential role in the ICS methodology. While Partial Solvers address a single aspect of the original problem, with a number of attributes fixed, Integrators build complete solutions by mixing partial solutions with promising features from these various populations. The selection operators thus need to take into account this objective by picking up not only the currently best solution in a given population, but rather a small number of diverse elite solutions. Similarly, in order to contribute to the progress of the global search, the usual quality and diversity criteria should guide the choice of complete solutions to be sent to the CSG (when the Integrator yields more than a single solution). More than one Integrator can be involved in an ICS implementation. Using different solution methodologies would then contribute toward the diversity objective.

Complete Solver Group and **Global Search Coordinator**. The CSG is the component of ICS where complete solutions are kept, and sometimes enhanced (e.g., Crainic et al., 2006), and from where the final solution to the original problem is obtained once the stopping conditions are verified.

The CSG is organized similarly to the PSGs, but the complete-solver set is optional or could hold a few methods only. Indeed, in most applications of interest for ICS, methods targeting the respective problem, if they exist, would be inadequate in computing efficiency or solution quality, or both. ICS is intended to replace them and, in such cases, the inclusion of solvers would not be warranted. On the other hand, post-optimization techniques may profitably be used to locally enhance solutions (e.g., customer sequencing for VRP or flow distribution for network design). Post-optimization methods could therefore belong to the complete-solver set.

Most importantly, it is the CSG that produces the *complete context information* required by the GSC for the global guidance of the search. Indeed, the purposeful-evolution objective of ICS requires the monitoring and guidance of the progress of the search. This is the main role of the Global Search Coordinator. More specifically, the GSC

monitors the evolution of the complete solution set, and those of the partial populations, as well as the behavior of the solver groups and integrators. This activity enables it to build and maintain the context information of the global search. This later may include various performance measures (e.g., the cost or elaborate functions that reflect multiple characteristics) and indicators (e.g., membership to a specific class of solutions, or information on the solver that yielded it) for each solution in the complete solution set. It may also include an image of the global search through statistical information (memories) on the evolution of solutions in the complete and partial populations, the contribution of solutions and their components (e.g., routes or arcs in VRP) to the evolution of the search, the relative performances of Partial Solvers and Integrators, etc.

The complete context information provides the means to detect undesired situations, e.g., loss of diversity in the partial or complete elite population, stagnation in the improvement of the best solution quality, awareness that some zones of the solution space - defined by particular values for particular decision sets - have been scarcely explored, if at all, and the search should be diversified in that direction, and so on. Thresholds on such global performance measures trigger guidance operations for ICS performed by sending "instructions" to Partial Solvers and Integrators. The particular type of guidance is application specific, but instructions may modify the values $\tilde{x_\delta}$ of the fixed attributes for a specific Partial Solver to orient the search toward a different area, change the attribute subset under investigation (i.e., change the decomposition of the decision-set attributes), or modify/replace the solution method in a Partial Solver or Integrator. The last two types of instructions significantly modify the structure of the global search and should reflect major observations in the behavior of the method, e.g., a solver is constantly under-performing compared to the others. Their usage should therefore be rather infrequent.

The first type of instructions, on the contrary, is the core guidance mechanism of the method and is implemented in the application discussed later on in this paper. In their simplest form, instructions take the form of a particular solution or solution subset being sent to re-initialize a particular partial population (re-initialization may be complete or partial, retaining in the latter case a few very good and diverse solutions) and thus restart the Partial Solver searches. Additional context information (e.g., the promising arc patterns of Le Bouthillier et al., 2005) may complete the guiding instructions, to further inflect the search trajectory toward regions that appear promising from the point of view of the global search.

This general ICS framework can be applied to any multi-attribute combinatorial problem class for which a decision set decomposition may be defined. The next section illustrates the application of the ICS methodology to such a problem class.

4 Application to the MDPVRP

We illustrate the ICS methodology with an application to the multi-depot periodic vehicle routing problem (MDPVRP). Our aim is double. First, to document the instantiation of ICS components and general method on a well-known multi-attribute problem and, second, to evaluate the performance of the method. We first briefly recall the MDPVRP setting and then detail the ICS application.

4.1 The multi-depot periodic vehicle routing problem

Briefly, see Vidal et al. (2012c) for a full description, the MDPVRP (Mingozzi, 2005) is defined on a multi-period planning horizon, each customer requiring service several times during this planning horizon according to one of a particular set of pre-defined visit patterns (i.e., lists of periods when visits may occur). Service is provided out of a set of depot, operating in all periods, by a homogeneous fleet of limited-capacity vehicles. The distribution of the fleet among depots is known and the same for all periods. The MDPVRP aims to select a depot and a visit pattern for each customer, with services in different periods to the same customer being required to originate at the same depot, such that the total cost of distribution is minimized.

The literature, we refer the reader to Vidal et al. (2012c) for a detailed review, shows a richer set of contributions for two restrictions of the MDPVRP, the periodic VRP (PVRP) where service proceeds out of a single depot, and the multi-depot VRP (MDVRP) where the planning horizon has a single period only. Most contributions to the MDPVRP did not consider all attributes simultaneously, but rather applied a successive-optimization approach (e.g., Hadjiconstantinou and Baldacci, 1998; Kang et al., 2005; Yang and Chu, 2000). We are aware of only two methods that address problems similar to the MDPVRP with a comprehensive approach. Parthanadee and Logendran (2006) implemented a tabu search method for a complex variant of the MDPVRP with backorders. Vidal et al. (2012c) proposed a hybrid genetic methodology, named *Hybrid Genetic Search with Adaptive Diversity Control* (*HGSADC*), combining the exploration capability of population-based evolutionary search, the aggressive-improvement capabilities of neighborhood-based meta-heuristics to enhance (educate) solution newly created by genetic operators, and advanced population-diversity management mechanisms to evaluate and select solutions. HGSADC is the current state-of-the-art method for the MDPVRP, as well as for the PVRP and the MDVRP. We therefore use HGSADC as solver in our ICS implementation for the MDPVRP and give more details in the next subsection.

4.2 ICS for the MDPVRP

To apply ICS to a given problem, we have to specify the decision-set attributes for the decomposition, the state of the art algorithms making up the solvers to address the resulting partial problems, the organization of each Partial Solver Group, the integrators to reconstruct solutions, and finally the Complete Solver Group and the coordination mechanisms.

We are opportunistic in this implementation and decompose the MDPVRP along the depot and period decision sets to create two partial problems, managed through two PSGs called *PSG-fixDep* and *PSG-fixPat*, respectively, which complement the CSG dedicated to the complete MDPVRP. The partial problem addressed in the *PSG-fixDep* group is a set of independent PVRPs optimizing pattern assignment and routing decisions, given customer-to-depot assignments. (The depot-specific PVRPs can be optimized independently when depot assignments are fixed.) Symmetrically, the second partial problem, addressed in *PSG-fixPat*, is a multiple MDVRP optimizing depot assignment and routing decisions for each period given pattern-to-customer selections. Notice that, fixing the pattern-to-customer selections simplifies the problem, but does not lead to a separable formulation as previously, since the depot assignment for each customer must be consistent among different periods.

Solver Groups. Two algorithms are used in this implementation, namely, GUTS, a generalized version of the Unified Tabu Search (UTS) of Cordeau et al. (2001) and the Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) of Vidal et al. (2012c). UTS is a tabu search-based meta-heuristic implementing advanced insertion neighborhoods and allowing the exploration of infeasible solutions by dynamically adjusted penalties on violation of vehicle capacity and route duration constraints. We implemented a generalized version, GUTS, which can be used either as complete or partial solver by fixing the appropriate attributes. HGSADC harnesses the strength of local-search improvement procedures and the exploration capacities of population-based methods, including advanced crossover operators and diversity management mechanisms. A new evaluation of individuals driven by both solution quality and contribution to the population diversity is proposed to further promote innovation within selection and population management procedures. This strategy was shown to be highly successful on periodic and multi-depot problems. We therefore rely on it to manage individuals in the elite sets of partial and complete solver groups with the same parameters as in Vidal et al. (2012c).

As illustrated in Algorithm 1, both GUTS and HGSADC can be used to work on a solution S_{INI} received from an elite set with fixed depot or pattern assignments. These solvers are designed to use pattern-change, depot-change, and inter-route movements, as well as intra-route optimization, but only on the solution components they are allowed to work on. In the case of GUTS, S_{INI} is simply considered an initial solution to be enhanced by the algorithm. In the case of HGSADC, initial individuals are generated as in Vidal

et al. (2012c), considering the allowed pattern and depot assignment alternatives, to create an initial population to which S_{INI} is added. These two algorithms are then run until $It_{\text{END}} = 5000$ successive iterations (number of local-search moves in the case of GUTS or number of generated individuals for HGSADC) have been performed without improvement of the best solution. When this threshold is reached, the solver restarts from a new solution from the elite set. Any improving solution is sent to the elite set if it remains the best for at least $It_{\text{COMM}} = 200$ successive iterations, and thus the elite set only receives solutions known to require some effort for further improvement. Furthermore, the solver receives after each It_{COMM} iterations a new solution selected from the elite set. This solution becomes the new starting solution of GUTS and it is included in the population of HGSADC, if it improves upon the current best solution of the solver. The solutions sent to the solvers are selected from the elite sets by binary tournament, using the individual evaluation function of Vidal et al. (2012c). To avoid sending the same solution multiple times, any solution which a solver has worked on until the termination criteria is marked with a flag, and cannot be received again by the same solver. A solution, which has been considered by all types of solvers without success, remains in the elite set for the purpose of integration, migration, and bookkeeping only.

Various combinations of GUTS and HGSADC, or parts thereof, yield different Partial Solver Groups. Preliminary experiments showed that"pure" cooperation, involving only GUTS or HGSADC solvers, outperforms "mixed" ones. The speed and quality of the evolution performed by HGSADC is the main reason behind this observation, as including one more HGSADC was experimentally shown to be in all cases more profitable than one additional GUTS. We thus investigate in Section 5 different versions of ICS, one with only neighborhood-based solvers (UTS) in the solver groups, and the other with HGSADC only. For the latter implementation, two different strategies have been evaluated. The first is the previously described *encapsulated* setting, where HGSADC is used with a dedicated population to improve single solutions, exchanging solutions ("migrating" individuals) with the elite set. The second is the *sharing* setting, where the elite set directly serves as population for all HGSADC solvers involved in the cooperation. In this latter case, flags are not used to mark solutions, and the role of the partial solver simply comes to iteratively receiving a pair of individuals, performing a crossover and a local search, and sending the resulting solution to the elite set (Line 2-4 in the HGSADC iteration of Algorithm 1). Notice that, in the sharing versions, the two individuals selected in the respective elite set for the crossover operation do not necessarily have the same depot or patterns associated to customers, and that the crossover operator will further modify these assignments.

Integration. Four integrators operate in parallel in the proposed MDPVRP application. The first, named I_{BEST}, selects the best solution from the population of each PSG and transfers it directly to the complete solution set of the GSG. The best solutions of each PSG are thus rapidly made available for sharing with the other PSGs and for extraction of relevant information for global guidance. The other three integrators aim to create new

Algorithm 1 Partial solvers

UTS CASE:
Repeat
RECEIVE an initial solution S_{INI}
$S_{\text{CUR}} \leftarrow S_{\text{INI}}$ and $S_{\text{BEST}} \leftarrow S_{\text{INI}}$

While Number of Local Search moves since last best solution $\leq It_{\text{END}}$

//One GUTS iteration
For each customer, search for a better insertion position given the fixed attributes (pattern or depot assignments);
Apply best non-tabu move to S_{CUR};
Adjust diversification parameters;
Adjust infeasibility penalties;
Update tabu list and status

//Update best and eventual communication
If $cost(S_{\text{CUR}}) < cost(S_{\text{BEST}})$ **then**
$S_{\text{BEST}} \leftarrow S_{\text{CUR}}$
If WhenFound$(S_{\text{BEST}}) = It_{\text{COMM}}$ **then**
SEND S_{BEST}
If Nb LS moves $= k \times It_{\text{COMM}}$ **then**
RECEIVE a solution S_{INI}
If $cost(S_{\text{INI}}) < cost(S_{\text{BEST}})$ **then**
SEND S_{CUR} and $S_{\text{CUR}} \leftarrow S_{\text{INI}}$

HGSADC CASE:
Repeat
RECEIVE an initial solution S_{INI}
Create an initial local population $\mathcal{P}$
Add S_{INI} to $\mathcal{P}$ and $S_{\text{BEST}} \leftarrow S_{\text{INI}}$
While Number of generated individuals since last best solution $\leq It_{\text{END}}$

//One HGSADC iteration
$(S_1, S_2) \leftarrow$ Select two parents in $\mathcal{P}$ by binary tournament;
$S_{\text{CUR}} \leftarrow$ Crossover(S_1, S_2);
$S_{\text{CUR}} \leftarrow$ LocalSearch(S_{CUR})
Add S_{CUR} to $\mathcal{P}$
if $|\mathcal{P}| >$ MaxPopSize then manage the population by removing some individuals;
Adjust infeasibility penalties;

//Update best and eventual communication
If $cost(S_{\text{CUR}}) < cost(S_{\text{BEST}})$ **then**
$S_{\text{BEST}} \leftarrow S_{\text{CUR}}$
If WhenFound$(S_{\text{BEST}}) = It_{\text{COMM}}$ **then**
SEND S_{BEST}
If Nb generated indivs $= k \times It_{\text{COMM}}$ **then**
RECEIVE a solution S_{INI}
If $cost(S_{\text{INI}}) < cost(S_{\text{BEST}})$ **then**
SEND S_{CUR} and add S_{INI} to $\mathcal{P}$

complete solutions out of pairs of partial solutions S_{FIXDEP} and S_{FIXPAT} randomly selected among the best 25% of the *PSG-fixDep* and *PSG-fixPat* populations, respectively. The I_{CROSS} integrator relies on the crossover operator of HGSADC, which extracts from each parent a subset of promising solution elements and is particularly efficient for combining two solutions. I_{CROSS} consists in applying 50 different times the crossover operator on S_{FIXDEP} and S_{FIXPAT}, and improving the respective resulting solution with local search. The best obtained solution is sent to the global solver group. The last two integrators, detailed in Crainic et al. (2012), aim to promote attributes obtained by solvers working on different facets of the problem while modifying part of these solutions to move to a different search subspace. The I_{AND} integrator fixes the attributes shared by the selected partial solutions and, then, addresses the resulting restricted formulation by means of HGSADC, to return the best solution. The I_{PEN} integrator modifies the original complete formulation by adding cost incentives to direct HGSADC toward the depot-to-customer assignments of S_{FIXPAT} and the pattern-to-customer assignments of S_{FIXDEP}.

Global Solver Group and Search Coordination. The Global Search Coordinator assumes the fourfold role of collecting statistical information on the past search, monitoring the status of the solver groups, triggering migration of individuals when necessary, and building new guiding solutions to orient the search towards promising features. The statistical information is developed on (client, depot, pattern) triplets entering the complete solution set, by monitoring the number of occurrences (frequency) of each triplet in the population as well as in the best, average and worst sub-populations (Le Bouthillier et al., 2005), and finally the best cost of a solution containing any given triplet.

The GSC monitors the elite sets to check whether more than five non-flagged individuals are available (UTS and encapsulated-HGSADC implementations), and whether improving solutions have been received during the $It_{\text{CONTROL}} = 2000$ last solutions. If one of these two criteria is not fulfilled, the GSC sends three solutions to the corresponding elite set. The three solutions are either randomly selected (equiprobably) from the complete solution set, or are three *guiding* solutions, as defined in the next paragraph. In the shared-HGSADC implementations, this migration is preceded by a reset of the population, where all the solutions are replaced by new initial random solutions. It should be noted that the solutions to be migrated do not necessarily have the same values for the fixed attributes, and thus the solver groups process individuals with identical sets of fixed attributes, but potentially different attribute values.

The GSC generates guiding solutions by selecting *promising* triplets to create feasible pattern and depot customer assignments. "Promising" is based on the complete search history and is meant as appearing in at least one solution with a cost difference of less than 3% with respect to the best found solution. Once these patterns and depots are selected, complete solutions with routes are generated by local search, to serve as initial population (HGSADC implementations) or solution (UTS implementations) and be optimized with the solver of choice. The termination criterion for GUTS or HGSADC is here set to

$It'_{\text{END}} = 2000$ iterations without improvement. Since creating such an individual is time consuming, guiding individuals are generated by the GSC in background, and stored in a temporary pool of guiding individuals. These solutions significantly contribute to inflect the search trajectory towards different alternatives of assignment combinations.

We experimented with two settings of the CSG, with and without a complete-solver set. The same two algorithms, GUTS and HGSADC, are used in the former case, targeting the full MDPVRP. The results are presented next.

5 Experimental Results and Analyses

Extensive experimental analyses were conducted to 1) assess the performance of ICS when compared to state-of-the-art sequential methods and, 2) investigate several implementation alternatives, including the choice of solver type (GUTS vs. HGSADC) and of cooperation strategy in the case of HGSADC (encapsulated vs. shared), as well as the potential inclusion of global solvers in the complete solver group.

The ICS method was implemented in C++. Experiments were run on an Altix 4700 server, using double Core processors Itanium 2 with 1.5GHz cadence and a NUMAlink communication network. 14 processors were used for the MDPVRP experiments:

- 3 CPUs managing each one of the elite sets;
- 6 CPUs for solvers: three for each of the two PSGs when no global solver was used, two for each PSG and two for the CSG, otherwise;
- 3 CPUs for the integrators, I_{BEST} and I_{CROSS} being run on the same CPU;
- 2 CPUs for the GSC, one for creating the guiding individuals, and the other fulfilling the remaining tasks.

This choice of task-to-CPU assignment is mostly due to implementation simplicity. Computational experiments show that, in practice, only 10 CPU out of 14 perform computationally-intensive tasks. The other four CPUs, dedicated to the elite set management and GSC monitoring, could be implemented on a single core in a more advanced implementation. We therefore consider in analyzing the results that the effective workload of ICS is 10 times the load of a sequential method.

Experiments were performed on six versions of ICS for the MDPVRP. *HGSADC-ICS1*, *HGSADC-ICS2*, and *GUTS-ICS* represent the population-based HGSADC sharing and encapsulated versions and the GUTS implementation, respectively, without general solvers. Their counterparts, *HGSADC-ICS1+*, *HGSADC-ICS2+*, and *GUTS-ICS+*, include the MDPVRP solvers. Each method was run ten times on each instance, recording the best solutions after 5, 10, 15 and 30 minutes minutes of computation time.

The ten MDPVRP instances pr01 - pr10 of Vidal et al. (2012c), derived from the Cordeau et al. (2001) data sets, were used for these tests. These instances present a uniform geographical distribution of customers coupled with a few customer clusters. The largest instances include up to 288 customers, 6 depots, 6 days, and a total of 864 deliveries. Detailed descriptions are provided in the Annex.

Table 1 sums up the performance comparison of the ICS versions and the two sequential solvers at different instances of computing time. Solution quality is reported as the average percentage of cost deviation (on all instances and all runs) with respect to the best known solutions in the MDPVRP literature, reported in Vidal et al. (2012c). In addition, Tables 2 and 3 provide detailed comparisons of the quality of solutions between the GUTS-ICS+ version and the sequential GUTS, and between the "best" population-based ICS version, HGSADC-ICS2+, and the sequential HGSADC, respectively. The first column in these tables specifies the instance solved, while the next columns report the average solution quality, the run time, and the best solution found by the sequential method. The two next groups of columns report the average and best results of the ICS versions at different points in time, and finally the last column reports the previous BKS from the literature. For each line, the best solution is reported in boldface. New best known solutions are underlined. Detailed results for the other ICS implementations are provided in the Annex.

Version	Gap 5 min	Gap 10 min	Gap 15 min	Gap 30 min
HGSADC	NC	NC	NC	+0.42%
HGSADC-ICS1	+0.78%	+0.51%	+0.40%	+0.29%
HGSADC-ICS2	+1.12%	+0.65%	+0.49%	+0.32%
HGSADC-ICS1+	+0.59%	+0.37%	+0.29%	+0.21%
HGSADC-ICS2+	+0.79%	+0.34%	+0.23%	+0.17%
GUTS	NC	NC	NC	+2.77%
GUTS-ICS	+1.62%	+1.04%	+0.78%	+0.58%
GUTS-ICS+	+1.29%	+0.94%	+0.71%	+0.52%

Table 1: Average performance of ICS versions and sequential algorithms

The reported results underline the large contribution of the ICS framework in enhancing neighborhood-based searches such as GUTS. Thus, the solution quality improves from a 2.77% gap to a 0.52% gap when relying on the cooperative framework. The improvement in the case of population-based searches appears smaller but still significant, reducing the average gap from 0.42% to 0.17%. This is impressive, given the smaller potential for improvement due to the high-quality solutions provided by HGSADC. It should be noted that ICS also enables to obtain high-quality solutions in reduced time, the average solution quality being higher after 10 minutes of ICS than after 30 minutes of HGSADC. Finally, during the overall experimentation process, HGSADC-ICS2+ qallowed us to find three new best known solutions for the considered problems: 5558.02 for pr05, 8254.73 for pr09, and 9776.28 for pr10 (these solutions may be obtained from the

Inst	GUTS			GUTS-ICS+								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.14	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	3547.72	2.39	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4639.50	18.87	4578.14	4495.73	4488.85	4482.96	4482.21	4483.24	**4480.87**	**4480.87**	**4480.87**	4480.87
pr04	5246.39	20.03	5225.67	5187.92	5167.56	5161.60	5153.83	5154.93	5153.73	5150.40	**5149.09**	5134.17
pr05	5738.40	21.10	5666.35	5726.11	5693.80	5652.20	5630.05	5686.88	5624.05	5618.88	**5599.41**	5570.45
pr06	6759.41	20.06	6723.52	6640.94	6621.20	6604.15	6586.48	6566.44	6566.09	6566.09	**6565.69**	6524.92
pr07	4644.93	0.71	4644.93	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6217.23	20.82	6174.15	6052.42	6024.29	6024.20	6024.15	6024.41	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8644.82	23.85	8548.63	8408.71	8356.79	8338.22	8302.88	8324.61	8294.79	8278.58	**8277.63**	8257.80
pr10	10241.67	26.45	10120.30	10270.76	10178.57	10102.29	10040.94	9990.04	9946.60	9935.10	**9930.32**	9818.42
Gap	+2.77%		+2.10%	+1.29%	+0.94%	+0.71%	+0.52%	+0.57%	+0.37%	+0.33%	+0.28%	

Table 2: Performance of GUTS-ICS

Inst	HGSADC			HGSADC-ICS2+								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.35	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	**3547.45**	1.49	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4491.08	7.72	**4480.87**	4480.94	**4480.87**	**4480.87**	**4480.87**	**4480.87**	**4480.87**	**4480.87**	**4480.87**	4480.87
pr04	5151.73	22.10	**5144.41**	5152.10	5145.63	5145.15	5144.42	**5144.41**	**5144.41**	**5144.41**	**5144.41**	5134.17
pr05	5605.60	10.00	5581.10	5649.40	5597.52	5588.82	5582.86	5610.52	5573.79	5572.44	**5558.92**	5570.45
pr06	6570.28	10.00	6549.57	6615.02	6554.82	6542.43	6538.54	6582.20	6534.18	6533.67	**6529.67**	6524.92
pr07	4502.06	2.18	4502.06	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6029.58	7.96	**6023.98**	6024.03	**6023.98**	**6023.98**	**6023.98**	**6023.98**	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8310.19	27.79	8271.66	8312.31	8287.48	8283.34	8274.42	8283.95	8256.99	**8256.47**	**8256.47**	8257.80
pr10	9972.35	30.00	**9852.87**	10215.12	10003.86	9938.42	9907.24	10099.90	9962.31	9887.07	9868.98	9818.42
Gap	+0.42%		+0.13%	+0.79%	+0.34%	+0.23%	+0.17%	+0.50%	+0.19%	+0.11%	+0.06%	

Table 3: Performance of HGSADC-ICS2+ with encapsulated population-based solvers

authors).

We complete this analysis by examining the performance distribution of all these methods with respect to their respective trade-offs between solution quality and overall CPU effort. Figure 4 illustrates this analysis, by displaying average and best-solution results for GUTS and HGSADC-based methods. The computation effort is computed multiplying the run time and the number of effectively working CPUs. Because the total effort to achieve best solutions is that of all the corresponding runs, the computation effort for these results is scaled accordingly.

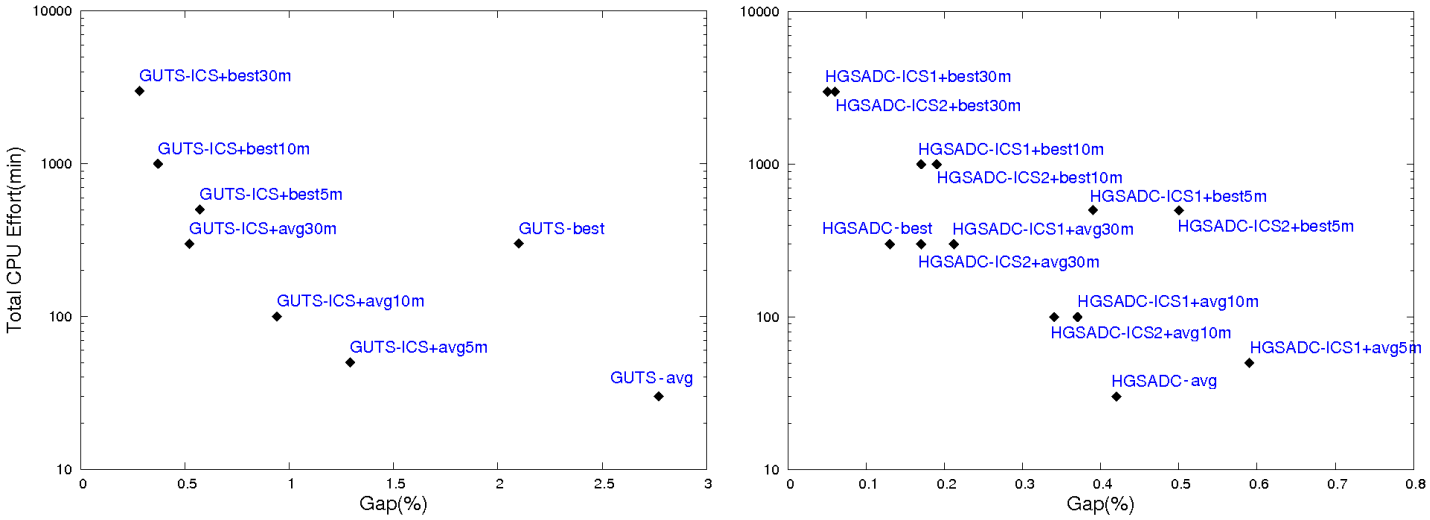

Figure 4: Solution Quality versus Computational Effort

Figure 4 shows that the exploration capabilities of the GUTS solvers are greatly enhanced by the cooperative framework, GUTS-ICS implementations leading to a variety of non-dominated (time/CPU consumption) trade-offs. Four variants of HGSADC constitute a Pareto set with respect to the twofold objective of solution quality and CPU effort: HGSADC-ICS1+best30m, HGSADC-best, HGSADC-ICS2+avg10m, and HGSADC-avg. For an equal CPU effort, a 10-processor parallel multi-search with HGSADC (HGSADC-best), seems to produce solutions of moderately higher quality than HGSADC-ICS2+ (deviation of 0.13% versus 0.17%). However, we must remark that HGSADC-ICS2+ with general solvers only rely on 2 solvers working on the general problem, while the best run of HGSADC out of 10 is comparable to the joint effort of 10 parallel general solvers. This emphasizes the excellent performance of ICS.

We now turn to discussing the results of a series of experiments targeting the impact of two particular ICS design features, the encapsulating or sharing of populations within PSGs, and the inclusion of general solvers in the CSG.

The results show that for short computation times (5 or 10 minutes), genetic algorithms that work directly on the elite sets (sharing ICS) seem to conduct to higher-quality

solutions. This observation relates to the fact that such a tight cooperation enables a fast exchange of solutions for an aggressive intensification. On longer runs, the encapsulated version of ICS, where each genetic solver operates its own population, seems to lead to better best solutions, thanks to an increased diversity of the search trajectories.

Finally, using general solvers within the CSG yields moderate increases in solution quality in all experiments. This is not surprising as it corroborates the general observation that addressing the complete problem formulation within the resolution approach is highly profitable, if possible. Indeed, of particular importance is the fact that, even when general solvers are not used, the ICS solution quality does only slightly decrease (a 0.08% difference in the worst case). This illustrates the remarkable ability of ICS to produce results of similar quality to the ones of the best methods addressing the complete problem, with only solvers that work on partial aspects of the problem. This observation is critical when dealing with multi-attribute, rich problems for which integrated solvers are either not available or not sufficiently efficient. ICS becomes the method of choice in these cases.

6 Conclusions

We introduced the Integrative Cooperative Search framework to efficiently address the challenges of rich, multi-attribute combinatorial optimization problems.

ICS decomposes such complex problems along decision-set attributes and musters, within a multi-thread cooperative search framework, the combined capabilities of a number of independent exact or meta-heuristic solution methods. A number of these methods work on sub-problems defined by suitably selected subsets of decision-set attributes of the problem, while others combine the resulting partial solutions into complete ones and, eventually, improve them. These methods cooperate through an adaptive search-guidance mechanism, using the central-memory cooperative search paradigm.

We illustrated the interest of the Integrative Cooperative Search methodology through an application to the multi-depot, periodic vehicle routing problem, for which ICS enhances the results of the current state-of-the-art methods.

To apply ICS to other combinatorial optimization problems is sufficient to identify decision-set attributes yielding suitable partial problems. Thus, for example, the recently proposed Unified Hybrid Genetic Search methodology (Vidal et al., 2012b) offers the means to rapidly devise high-performing meta-heuristics for a wide gamut of routing problem settings. Similarly, multi-attribute design problems may be addressed through ICS integrating efficient tabu search and path relinking meta-heuristics (Crainic et al., 2006). This emphasizes the generality and broad applicability of the proposed ICS

methodology.

Acknowledgments

While working on this project, T.G. Crainic was the NSERC Industrial Research Chair in Logistics Management, ESG UQAM, N. Lahrichi and G.C. Crişan were postdoctoral fellows with the Chair, and M. Gendreau was the NSERC/Hydro-Québec Industrial Research Chair on the Stochastic Optimization of Electricity Generation, MAGI, École Polytechnique. Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair, Collaborative Research and Development, and Discovery Grant programs, by our partners CN, Rona, Alimentation Couche-Tard, la Fédération des producteurs de lait du Québec, and the Ministry of Transportation of Québec, and by the Fonds de recherche du Québec - Nature et technologies (FRQ_NT) through its Team Research Project program.

References

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

R. Baldacci and A. Mingozzi. A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming A*, 120(2):347–380, 2009.

R. Baldacci, E. Bartolini, A. Mingozzi, and A. Valletta. An exact algorithm for the periodic routing problem. *Operations Research*, 59(1):228–241, 2011.

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

J.-F. Cordeau and M. Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & O.R.*, 39(9):2033–2050, 2012.

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

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

T.G. Crainic. 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.

T.G. Crainic. Parallel Solution Methods for Vehicle Routing Problems. In Golden, B.L., Raghavan, S., and Wasil, E.A., editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 171–198. Springer, New York, 2008.

T.G. Crainic and H. Nourredine. Parallel Meta-Heuristics Applications. In Alba, E., editor, *Parallel Metaheuristics: A New Class of Algorithms*, pages 447–494. John Wiley & Sons, Hoboken, NJ, 2005.

T.G. Crainic and M. Toulouse. Explicit and Emergent Cooperation Schemes for Search Algorithms. In Maniezzo, V., Battiti, R., and Watson, J.-P., editors, *Learning and Intelligent Optimization*, volume 5315 of *Lecture Notes in Computer Science*, pages 95–109. Springer-Verlag, Berlin, 2008.

T.G. Crainic and M. Toulouse. Parallel Meta-Heuristics. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of Metaheuristics (2nd Edition)*, pages 497–541. Springer, 2010.

T.G. Crainic, B. Di Chiara, M. Nonato, and L. Tarricone. Tackling Electrosmog in Completely Configured 3G Networks by Parallel Cooperative Meta-Heuristics. *IEEE Wireless Communications*, 13(6):34–41, 2006.

T.G. Crainic, M. El Hachemi, N. Gendreau, N. Lahrichi, W. Rei, and V. Thuibaut. Solution Integration in Combinatorial Optimization: Applications to Cooperative Search and Multi-Attribute Vehicle Routing. Publication CIRRELT-2012-, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada, 2012.

Z. Drezner and H. Hamacher, editors. *Facility Location: Application and Theory*. Springer Verlag, Berlin, 2002.

A.M. Geoffrion. Elements of Large-Scale Mathematical Programming. *Management Science*, 16:652–675, 1970.

B. L. Golden, A. A. Assad, and E.A. Wasil. Routing vehicles in the real world: Applications in the solid waste, beverage, food, dairy, and newspaper industries. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 245–286. SIAM, Philadelphia, PA, 2002.

B. L. Golden, S. Raghavan, and E.A. Wasil, editors. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, New York, 2008.

C. Groër and B. L. Golden. A parallel algorithm for the vehicle routing problem. *INFORMS Journal on Computing*, 23(2):315–330, 2011.

M. Grötschel, C. Monma, and M. Stoer. Design of Survivable Networks. In M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, editors, *Network Models*, volume 7 of *Handbooks in Operations Research and Management Science*, pages 617–672. North-Holland, Amsterdam, 1995.

E. Hadjiconstantinou and R. Baldacci. A multi-depot period vehicle routing problem arising in the utilities sector. *Journal of the Operational Research Society*, 49(12): 1239–1248, 1998.

R.F. Hartl, G. Hasle, and G K. Jansens (editors.). Special Issue on Rich Vehicle Routing Problems. *Central European Journal of Operations Research*, 13(2), 2006.

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

J. Homberger and H. Gehring. A Two-Phase Hybrid Metaheuristics for the Vehicle Routing Problem with Time Windows. *European Journal of Operational Research*, 162:320–238, 2005.

J. Jin, T.G. Crainic, and A. Løkketangen. A Cooperative Parallel Metaheuristic for the Capacitated Vehicle Routing Problem, 2012a.

J. Jin, T.G. Crainic, and A. Løkketangen. A Parallel Multi-Neighborhood Cooperative Tabu Search for Capacitated Vehicle Routing Problems. *European Journal of Operational Research*, 222(3):441–451, 2012b.

K. H. Kang, Y. H. Lee, and B. K. Lee. An exact algorithm for multi depot and multi period vehicle scheduling problem. In *Computational Science and Its Applications - ICCSA 2005*, Lecture Notes in Computer Science, pages 350–359. Springer, Berlin / Heidelberg, 2005.

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

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

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

A. Mingozzi. The multi-depot periodic vehicle routing problem. In *Abstraction, Reformulation and Approximation*, Lecture Notes in Computer Science, pages 347–350. Springer, Berlin / Heidelberg, 2005.

P. Parthanadee and R. Logendran. Periodic product distribution from multi-depots under limited supplies. *IIE Transactions*, 38(11):1009–1026, 2006.

C. Rego. Node ejection chains for the vehicle routing problem: Sequential and parallel algorithms. *Parallel Computing*, 27:201–222, 2001.

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

P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, PA, 2002.

T. Vidal, T.G. Crainic, M. Gendreau, and C. Prins. A Unifying View on Timing Problems and Algorithms. Publication CIRRELT-2011-43, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada, 2011.

T. Vidal, T.G. Crainic, M. Gendreau, and C. Prins. Heuristics for Multi-Attribute Vehicle Routing Problems: A Survey and Synthesis. Publication CIRRELT-2021-, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada, 2012a.

T. Vidal, T.G. Crainic, M. Gendreau, and C. Prins. A Unified Solution Framework for Multi-Attribute Vehicle Routing Problems. Publication cirrelt-2012-, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada, 2012b.

T. Vidal, T.G. Crainic, N. Gendreau, M.and Lahrichi, and W. Rei. A Hybrid Genetic Algorithm for Multi-Depot and Periodic Vehicle Routing Problems. *Operations Research*, 60(3):611–624, 2012c.

W. T. Yang and L. C. Chu. A heuristic algorithm for the multi-depot periodic vehicle routing problem. *Journal of Information & Optimization Sciences*, 22:359–367, 2000.

Annex

We first display the characteristics of the test instances and then present the detailed result tables for the four ICS versions not included in the body of the text. Each entry is the average of ten repetitions.

The MDPVRP benchmark instances introduced in Vidal et al. (2012c) were produced by merging the instances for the MDVRP and the PVRP described in Cordeau et al. (2001). The characteristics of the instances are summarized in Table 4. The number of customers in each instance is represented by n, while t is the number of periods in the PVRP and the number of depots in the MDVRP. m is the number of vehicles for the MDPVRP. This number of vehicles available each day at each depot is set to the minimum number of vehicles needed. D and Q are the capacity and the duration constraints on the vehicles, respectively.

Inst	**n**	**t**	**m**	**D**	**Q**
pr01	48	4	1	500	200
pr02	96	4	1	480	195
pr03	144	4	2	460	190
pr04	192	4	2	440	185
pr05	240	4	3	420	180
pr06	288	4	3	400	175
pr07	72	6	1	500	200
pr08	144	6	1	475	190
pr09	216	6	2	450	180
pr10	288	6	3	425	170

Table 4: MDPVRP instances

Inst	GUTS			GUTS-ICS								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.14	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	3547.72	2.39	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4639.50	18.87	4578.14	4492.07	4483.49	4482.74	4481.51	4483.29	4480.90	**4480.87**	**4480.87**	4480.87
pr04	5246.39	20.03	5225.67	5188.12	5171.35	5168.81	5163.29	5170.63	5156.16	5156.16	**5148.06**	5134.17
pr05	5738.40	21.10	5666.35	5762.78	5674.16	5644.62	5633.24	5663.68	5635.88	5626.66	**5616.02**	5570.45
pr06	6759.41	20.06	6723.52	6716.27	6615.74	6603.22	6585.77	6575.16	6559.11	6555.71	**6548.86**	6524.92
pr07	4644.93	0.71	4644.93	4502.06	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6217.23	20.82	6174.15	6038.41	6024.40	6024.19	6024.08	6024.00	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8644.82	23.85	8548.63	8430.12	8356.94	8322.31	8306.93	8323.97	8305.61	8286.66	**8277.93**	8257.80
pr10	10241.67	26.45	10120.30	10421.26	10324.62	10192.87	10071.34	10106.10	10092.50	10018.80	**10006.50**	9818.42
Gap	+2.77%		+2.10%	+1.62%	+1.04%	+0.78%	+0.58%	+0.69%	+0.55%	+0.43%	+0.36%	

Table 5: Performance of GUTS-ICS

Inst	HGSADC			HGSADC-ICS1								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.35	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	**3547.45**	1.49	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4491.08	7.72	**4480.87**	4485.72	4483.03	4482.07	4480.88	**4480.87**	**4480.87**	**4480.87**	**4480.87**	4480.87
pr04	5151.73	22.10	5144.41	5162.75	5156.36	5153.18	5149.17	5154.83	5146.29	5146.14	**5144.41**	5134.17
pr05	5605.60	10.00	5581.10	5674.21	5622.02	5608.19	5598.38	5606.55	5580.53	5577.83	**5574.61**	5570.45
pr06	6570.28	10.00	6549.57	6602.40	6579.26	6568.33	6555.72	6574.52	6548.81	6543.54	**6542.32**	6524.92
pr07	4502.06	2.18	4502.06	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6029.58	7.96	**6023.98**	6024.29	6024.03	6024.03	6023.99	**6023.98**	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8310.19	27.79	8271.66	8325.98	8297.34	8290.60	8281.60	8265.31	8262.99	**8261.74**	**8261.74**	8257.80
pr10	9972.35	30.00	9852.87	10138.28	10052.64	10003.22	9949.62	9970.82	9907.04	9904.36	**9830.38**	9818.42
Gap	+0.42%		+0.13%	+0.78%	+0.51%	+0.40%	+0.29%	+0.35%	+0.17%	+0.16%	+0.07%	

Table 6: Performance of HGSADC-ICS1 with shared populations

Inst	HGSADC			HGSADC-ICS1+								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.35	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	**3547.45**	1.49	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4491.08	7.72	**4480.87**	4482.33	4481.35	4480.99	**4480.87**	**4480.87**	**4480.87**	**4480.87**	**4480.87**	4480.87
pr04	5151.73	22.10	5144.41	5158.64	5152.78	5150.57	5148.79	5146.63	5145.62	5145.62	**5144.45**	5134.17
pr05	5605.60	10.00	5581.10	5631.65	5601.88	5593.40	5587.97	5613.77	5590.25	5576.38	**5573.91**	5570.45
pr06	6570.28	10.00	6549.57	6599.25	6567.80	6560.05	6551.72	6573.99	6550.10	6543.19	**6536.97**	6524.92
pr07	4502.06	2.18	4502.06	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6029.58	7.96	**6023.98**	6024.09	6024.03	6024.01	6024.01	**6023.98**	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8310.19	27.79	8271.66	8306.52	8287.13	8278.86	8273.77	8288.08	8263.17	8261.70	**8256.67**	8257.80
pr10	9972.35	30.00	9852.87	10065.30	9993.48	9950.46	9902.44	9995.04	9879.76	9859.08	**9826.14**	9818.42
Gap	+0.42%		+0.13%	+0.59%	+0.37%	+0.29%	+0.21%	+0.39%	+0.17%	+0.11%	+0.05%	

Table 7: Performance of HGSADC-ICS1+ with shared populations

Inst	HGSADC			HGSADC-ICS2								BKS
	Avg	T(min)	Best	Avg				Best				
				5 min	10 min	15 min	30 min	5 min	10 min	15 min	30 min	
pr01	**2019.07**	0.35	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	**2019.07**	2019.07
pr02	**3547.45**	1.49	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	**3547.45**	3547.45
pr03	4491.08	7.72	**4480.87**	4495.57	4482.45	4480.91	4480.91	**4480.87**	**4480.87**	**4480.87**	**4480.87**	4480.87
pr04	5151.73	22.10	5144.41	5168.41	5155.54	5150.79	5148.27	5154.51	**5144.41**	**5144.41**	**5144.41**	5134.17
pr05	5605.60	10.00	5581.10	5682.73	5639.67	5617.91	5604.86	5621.61	5570.54	5570.54	**5568.28**	5570.45
pr06	6570.28	10.00	6549.57	6638.89	6585.02	6569.59	6551.77	6605.94	6560.20	6541.38	**6528.58**	6524.92
pr07	4502.06	2.18	4502.06	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	**4502.02**	4502.02
pr08	6029.58	7.96	**6023.98**	6029.07	6024.01	**6023.98**	**6023.98**	**6023.98**	**6023.98**	**6023.98**	**6023.98**	6023.98
pr09	8310.19	27.79	8271.66	8341.98	8308.42	8292.10	8277.61	8291.09	8263.99	8261.47	**8261.10**	8257.80
pr10	9972.35	30.00	9852.87	10339.20	10139.79	10075.60	9984.54	10147.90	9986.01	9959.13	**9896.27**	9818.42
Gap	+0.42%		+0.13%	+1.12%	+0.65%	+0.49%	+0.32%	+0.63%	+0.25%	+0.19%	+0.10%	

Table 8: Performance of HGSADC-ICS2 with encapsulated population-based solvers