CIRRELT

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

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Explicit and Emergent Cooperation Schemes for Search Algorithms

Teodor Gabriel Crainic
Michel Toulouse

February 2008

CIRRELT-2008-04

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

Explicit and Emergent Cooperation Schemes for Search Algorithms

Teodor Gabriel Crainic[1,2,*], Michel Toulouse[1,3]

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

[2] Department of Management and Technology, Université du Québec à Montréal, C.P. 8888, succursale Centre-ville, Montréal, Canada H3C 3P8

[3] Department of Computer Science, E2-445 EITC, University of Manitoba, Winnipeg, Canada R3T 2N2

Abstract. Cooperation as problem-solving and algorithm-design strategy is widely used to build methods addressing complex discrete optimization problems. In most cooperative-search algorithms, the explicit cooperation scheme yields a dynamic process not deliberately controlled by the algorithm design but inflecting the global behaviour of the cooperative solution strategy. The paper presents an overview of explicit cooperation mechanisms and describes issues related to the associated dynamic processes and the emergent computation they often generate. It also identifies a number of research directions into cooperation mechanisms, strategies for dynamic learning, automatic guidance, and self-adjustment, and the associated emergent computation processes.

Keywords. Cooperative-search methods, meta-heuristics, emerging computation, learning, emerging cooperation.

Acknowledgements. This work was financially supported through the Industrial Research Chair and Discovery grant programs of the Natural Sciences and Engineering Research Council of Canada (NSERC).

---

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

Dépôt légal – Bibliothèque nationale du Québec,
Bibliothèque nationale du Canada, 2008

1 Introduction

Cooperation as problem-solving and algorithm-design strategy is widely used in many fields, including but not restricted to ad hoc wireless networks, swarm robotics, multi-agent systems, constraint programming, and exact and meta-heuristic methods addressing complex discrete optimization problems. While the cooperation paradigm may take different forms, they all share two features: a set of highly autonomous programs (APs), each implementing a particular solution method, and a cooperation scheme combining these APs into a single problem-solving strategy. This strategy is different from the "cooperation" found in distributed algorithms, which is predetermined by the decomposition of an algorithm into concurrent processes. Cooperative-search algorithms combine independent problem-solving strategies, that is, each element of the cooperation is a stand-alone method able, in most cases, to address, "solve", the problem instance considered. Consequently, cooperation mechanisms must be designed explicitly, which constitutes the core of writing a cooperative algorithm.

In most cooperative-search algorithms, the explicit, deliberately designed, cooperation scheme yields an associated stream of correlated interactions, i.e., reactive actions with a potential for emergent computation and, eventually, cooperation. This dynamic process, active simultaneously with the deliberate, optimization-oriented cooperative-search algorithm, is neither controlled by the algorithm design, nor spontaneously oriented toward the optimization of the problem at hand, but contributes to determine the global behaviour of the cooperative solution strategy.

Correlated and indirect interactions are similar to ripple (or side) effects in computing systems, defined as coherent processing activities partly independent of algorithmic rules. In distributed computation, for example, a delay to execute some instructions (due to workload variations of shared resources such as CPUs) can generate a ripple effect on the ordering of the execution of several other concurrent operations. Distributed algorithms are, in fact, notoriously difficult to debug because ripple effects make faulty behaviour almost impossible to reproduce. Ripple effects are a form of adaptive response of the processing activities to events not explicitly accounted for in the algorithm design.

In the context of cooperative-search methods, correlated asynchronous interactions occur according to system conditions, as well as the internal state of the cooperating APs and the information exchanged. Though they escape the direct control of cooperation schemes, indirect interactions influence the subsequent logical steps of the APs whether they interact with the computing environment or with other search programs. Now the question is: could indirect interactions, which are inevitable in cooperative search, be harnessed to yield a better performing method? Under which conditions could this behaviour be obtained? Could we understand and eventually "design" an implicit cooperation strategy emerging out of these interactions?

We believe two important research issues stand out in this context: On the one hand, the development of good cooperation mechanisms, including strategies for dynamic learning, automatic guidance, and self-adjustment; On the other hand, the study of the associated dynamic processes focusing on the possible emergence of a second, implicit, layer of cooperation among the APs and on how to harness it to obtain a "better" search method for the problem at hand.

The goal of this paper is to contribute to address these issues. We give an overview of the main explicit cooperation mechanisms encountered in the literature and describe issues related to the associated dynamic processes and possible emergent computation. We also identify a research agenda focused on these issues that we believe important and timely given the current interest not only in the parallelization of exact and meta-heuristic solution methods, but also in novel algorithmic schemes (e.g., the so-called swarm methods).

The paper is organized as follows. Section 2 briefly recalls the main cooperative parallel meta-heuristics mechanisms. Section 3 discusses the reactive component of cooperative-search algorithms and its potential for emergent cooperation. Section 4 revisits cooperative parallel meta-heuristics mechanisms focusing on dynamic learning, automatic guidance, and self-adjustment strategies as a partial response to emergent computation issues. Research avenues are summarized in Section 5 and we conclude in Section 6.

2 Explicit cooperation schemes in meta-heuristics

Providing a full-length literature review of contributions to cooperation and meta-heuristics is beyond the scope of this paper. Our objective is to synthesize the state-of-knowledge of the field as we see it. The interested reader may consult a number of survey papers on parallel meta-heuristics that dedicate (under various forms and names) significant space to cooperative methods, including two recent books [1, 30] that collect chapters on many issues in parallel computing for combinatorial optimization and [7, 8, 14, 17].

Parallelism in general, and cooperative strategies in particular, imply that both the individual APs and the resulting global search proceed most of the time with incomplete knowledge regarding the status of the search. The design of the information exchange mechanisms is thus a key element to the good performance of cooperative methods. Important cooperation design issues include its *content* (what information to exchange), *timing* (when to exchange it), *connectivity* (the logical inter-processor structure), *mode* (synchronous or asynchronous communications), *exploitation* (what each AP does with the received information), as well as its *scope*, that is whether new information and knowledge is to be extracted from the exchanged data to guide the search.

The importance of these issues is reflected in most parallel meta-heuristic taxonomies. To illustrate, consider the classification of Crainic and Nourredine [13], which generalizes that of [16, 7]; [33, 17] present classifications that proceed of the same spirit), where two of the three dimensions relate to cooperation issues. Thus, the *Search Control Cardinality* dimension examines how the global search is controlled: either by a single process or collegially by several processes that may collaborate or not. Cooperative methods belong to the second category denoted *p-control (pC)*. The *Search Control and Communications* dimension then addresses the issue of information exchanges according to four classes to reflect the quantity and quality of the information 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)*. As for the third dimension, *Search Differentiation*, it reflects the diversity of the initial solutions and search strategies: *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* (where "point" is used for neighbourhood-based methods).

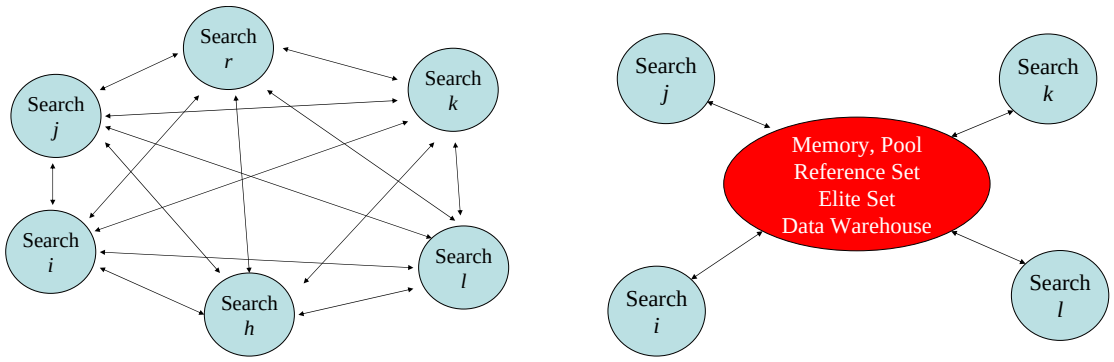
The most crude form of cooperation involves (almost) no cooperation at all and is identified as *pC-RS* with any of the previous search differentiation strategies. Such *independent* multi-search methods start several processes, using the same or different solution strategies, from different initial configurations. No attempt is made to take advantage of the multiple APs running in parallel, other than to identify the best overall solution once all processes stop. This parallelization of the classic sequential multi-start heuristic is easy to implement and may offer satisfactory results in terms of search acceleration.

pC-KS strategies obtain cooperation by adopting the same general approach as in the independent search case but taking advantage of the parallel exploration by synchronizing the APs 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 AP 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 neighbouring processors). Here, as in the more advanced cooperation mechanisms indicated bellow, *migration* is the term used to identify information exchanges in population-based parallel algorithms.

Synchronization was seen as a means to re-create a state of complete knowledge to share among all participating individual methods, and 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 strategies, synchronous cooperative methods display larger computational overheads, appear less reactive to the evolution of the global parallel search, and conduct to the premature

convergence of the associated dynamic process. It has been shown, for example, that frequent broadcasting of new solutions that stop individual methods from continuing to explore improving sequences leads to either a random search or premature convergence.

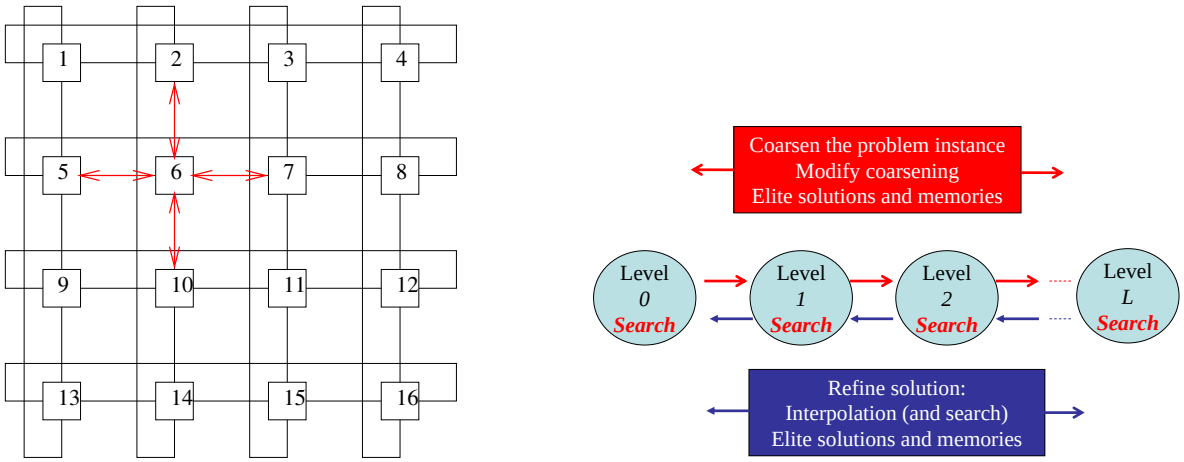
Controlled, parsimonious, and timely exchanges of meaningful information are thus characteristic of successful cooperative strategies. Asynchronous methods belong to this group and may be characterized according to the quantity and quality of the information exchanged and, eventually, the "new" knowledge inferred based on these exchanges. *pC-C* asynchronous cooperative methods exchange "good" solutions only or, when a memory mechanism exists, implement simple strategies to extract solutions from memory to pass to APs. More advanced designs, denoted *pC-KC*, add procedures to create new information and solutions based on the solutions exchanged, and implement guiding mechanisms based on this information.



(a) Many-to-Many (b) Memory Based

Figure 1: Direct Communication Schemes

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 a genetic algorithm form a so-called *island*. Each island may potentially communicate with any of the other islands, as illustrated in Figure 1a. Then, according to an exchange protocol (e.g., on demand from an island with low population diversity), a migration operator sends a "good" individual to another island. This parallel cooperative 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 2a. Such limitations are generally the result of particular topologies of the processor network, e.g., the 2-D torus of Figure 2a. Communications then take place only among adjacent processors according to a so-called *diffusion* mechanism. Notice that, islands tend to have very small populations in this case 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.



(a) 2-D Torus (b) Multi-level

Figure 2: Diffusion Communication Schemes

Many cooperative developments outside the evolutionary community are based on indirect communications and, currently, the largest number use some form of *memory* for inter-process 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 heuristic or exact methods are generally assigned each to a processor, as illustrated in Figure 1b. In the literature, so-called *adaptive-memory* methods [29] store partial elements of good solutions and combine them to create new complete solutions that are then improved by the cooperating programs, while *central-memory* approaches [15] exchange complete elite solutions that are then used to steer the search and, eventually, create new information.

Cooperation is achieved through asynchronous exchanges of information through the pool (which may share a processor with an AP or be assigned a particular one). Whenever a program desires to send out information (e.g., when a new local optimum is identified), it sends it to the pool. Similarly, when a program needs to access outside information (e.g., to diversify the search), it reaches out and takes it from the pool. Communications are initiated exclusively by the APs, 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 programs that will receive or send information and to control the co-operation. The pool 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 AP may access at all times the data previously sent out by any other AP.

Multi-level cooperative search [32] offers a different *pC-KC* cooperation approach

based on controlled diffusion of information principles (Figure 2b). Each AP works at a different level of aggregation of the original problem (one AP works on the original problem) and communicates exclusively with the APs working on the immediate higher and lower aggregation levels. Improved solutions are exchanged asynchronously at moments dynamically determined by each AP 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.

Strict and knowledge-synchronous mechanisms yield a rather strict control of the global search, the trajectory of each AP in the cooperation changing according to the state of all other APs, resulting in no or little emergent behaviour being observed. On the other hand, however, these approaches have been shown experimentally to generally yield inferior results to those of collegial and knowledge-collegial strategies. The behaviour of APs in the latter contexts depends on the information exchanged and, in the case of memory-based cooperation, on the information stored and its management. Moreover, the evolution of *pC-KS* cooperative systems creates new knowledge, new solutions, targets, and statistics, based on the information stored in the memory structure and uses this new knowledge to influence the trajectory of each AP in the cooperation and, thus, the trajectory of the global search. The information propagation inherent to these asynchronous cooperation mechanisms yields significant emergent behaviour as discussed in the next section.

3 Emergent computation and cooperation

All cooperation search mechanisms presented in the previous section involve *explicit* exchanges of information among the APs. These exchanges are defined by the design of the cooperation scheme, which details exactly what and when information is to be shared, as well as how this information is to be used. This explicit design exists even when exchanges are performed asynchronously and indirectly through a pool.

The information collected and communicated through an explicit exchange by one AP, the sender, generally modifies information available to, and thus the search trajectory of, at least one other program: the receiver. For example, the sender may communicate its newly improved best solution x and the receiver may re-initialize its search from it. In other words, the control and behaviour of the search performed by the receiver AP is modified by the search and information sharing behaviour of the sender AP. This phenomenon is denoted *direct* (explicit) AP *interaction*.

Explicit information exchanges are designed to improve the performance of the global search performed by the APs involved in the cooperation compared to their individual

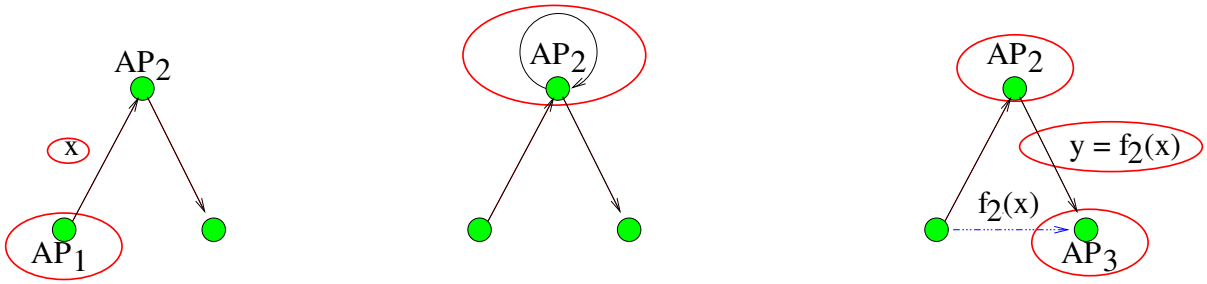


Figure 3: Information Propagation Process

performances. This goal if often attained but not always or not always at the level hoped for [31]. This is largely due to the fact that, in most cooperative search systems, APs interact not only through explicit information exchanges, but also indirectly through correlated cooperation actions. Information among cooperating APs is thus also shared implicitly through a propagation (or diffusion) process not explicitly defined by their design or that of the cooperation mechanism.

Figure 3 describes a simple propagation process. AP_1 sends information x to AP_2 via a direct interaction $1 \xrightarrow{x} 2$. AP_2 receives this information and uses it to guide its exploration. Later, it sends information $y = f_2(x)$ to AP_3 via a direct interaction $2 \xrightarrow{y} 3$. The notation indicates that the information sent by AP_2 to AP_3, y, results, at least partially, from an interaction between AP_1 and AP_2, which modified the trajectory executed by the search heuristic f_2 of AP_2. There is implicit information propagation because the second interaction is triggered by the modification to the search behaviour of AP_2 following an interaction with AP_1. The second interaction is correlated to the occurrence of the first one.

Correlated interactions propagate control actions of one program onto other programs. In the example of Figure 3, the search activities of AP_1 modify the search behaviour of AP_3, as indicated by the arrow between the two APs. We identify this control as *indirect* (implicit) AP *interaction*.

When direct interactions occur asynchronously according to the internal state of the interacting APs, chains of correlated interactions build up spontaneously among the APs. Sequences of bold arrows in Figure 4 illustrate such occurrences of correlated interactions under a 2-D torus interconnection network, while dashed arrows represent some of the associated indirect control activities. Figure 4(a) pictures a chain of correlated interactions generated by the sequence of direct interactions $5 \rightarrow 6$, $6 \rightarrow 10$, $10 \rightarrow 14$, $14 \rightarrow 15$, $15 \rightarrow 3$, $3 \rightarrow 4$, and $4 \rightarrow 8$, together with one of several associated indirect interactions: $5 \rightarrow 8$. Figure 4(b) illustrates the case where two chains of correlated interactions develop concurrently, while Figure 4(c) displays indirect interactions forming loops inside a network of correlated interactions. These are only illustrative examples,

of course. Yet, they help getting a sense of the complexity of the control activities associated to indirect interactions, the spontaneity of this control, the inter-connectivity of the programs in a cooperative search, and the dependence of the search performed by each AP on these emerging control structures that are the correlated interactions. In a central memory-based systems, where chains of correlated interactions are not restricted by the logical topology of the interconnection network, this self-organization of the search through correlated interactions is even more striking.

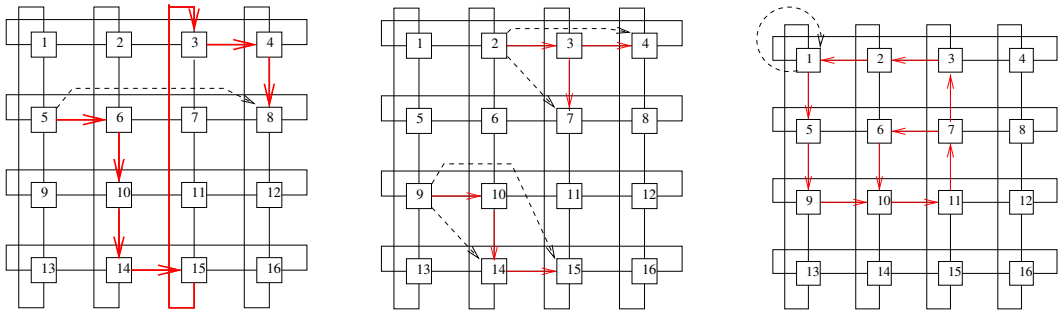


Figure 4: An Illustration of Indirect Information and Control Propagation

Chains of correlated interactions are the ripple effects of cooperative search and constitute the means by which information propagates among APs, information different from and in addition to that resulting from the interactions specified by the cooperation scheme. It must be emphasized that, unlike explicit information exchanges, the sharing of information through propagation is **not** specified in the cooperation scheme. Nor are specified the search control activities that emerge spontaneously from these information-propagation processes. Moreover, this spontaneous organization of the control activities plays a more important role in the global exploration of the search space performed by cooperation mechanisms implementing asynchronous *pC-C* and *pC-KC* strategies, where the cooperation scheme is executed independently and asynchronously by each cooperating AP.

Interesting questions arise from the realization that emergent control behaviour occurs in systems of cooperating APs and that the global exploration of the search space performed by the cooperating APs is thus the result of the interplay between the two types of AP interactions, the direct ones specified by the design of the cooperation and the complex system of indirect interactions emerging from them. Could indirect interactions act similarly to direct ones and, by modifying the search behaviour of the receiving AP improve its performance? Could indirect interactions emerge as an implicit cooperation scheme, that is, could indirect interactions support a global cooperative exploration of the search space? Does or could this emergent control help improve the exploration of the search space? In what circumstances could the answer to some of these questions by "yes" ? Understanding how such global behaviour may emerge and how (if) it could be

harnessed to support the search for good solutions to the problem in hand, could prove important in enhancing the performance of cooperating search methods.

4 Advanced cooperation mechanisms and learning

Studies on emergent cooperation issues have been performed within a number of scientific fields (Section 5) but, with the notable exception of the multi-level paradigm designed to control the indirect diffusion of information among cooperating APs, few efforts were dedicated to these issues within the operations research or parallel meta-heuristic communities. Most efforts were rather directed toward improving the cooperative meta-heuristic mechanisms to enhance their optimization capabilities. Learning was and continues to play a central role in these processes, particularly for memory-based mechanisms.

Consider the adaptive-memory approach where the pool contains solution components (e.g., tours) of good solutions identified by APs (e.g., multi-tours for VRPTW found by tabu searches) that are ranked according to attribute values, including the objective values of their respective solutions.. Each APs then probabilistically selects components in the memory, constructs a new initial solution (e.g., by solving a set-covering heuristic), improves it, and returns the components of its best solution to the memory.. The learning mechanism of adaptive-memory approaches is thus composed of the partial solutions kept in the memory together with the continuously updated rank values, combined to a new-solution creation feature.

Algorithms based on the central-memory approach keep full solutions and attributes sent by the APs involved in cooperation. APs may construct new solutions, execute a neighbourhood-based improving meta-heuristic, implement a population-based meta-heuristic, or perform post-optimization procedures on solutions in the pool. Improving meta-heuristics aggressively explore the search space, while population-based methods (e.g., genetic algorithms [10, 21] and path relinking [9]) contribute toward increasing the diversity of solutions exchanged among the co-operating methods. Exact solution methods may participate to the cooperation either to build solutions or to seek out optimal ones (on restricted versions of the problem, eventually). The information exchanged among cooperating APs has to be meaningful, in the sense that it has to be useful for the decision process of the receiving programs, the evolution of the shared data, and thus the evolution of the global search, or both. Information indicative of the current status of the global search or, at least, of some individual search program is, in this sense, meaningful. The basic mechanisms thus only implement exchanges of local good solutions (local optima) together with ranking procedures and probabilistic solution-extraction procedures in the central memory, thus implementing the same learning mechanisms described previously.

More advanced mechanisms may involve exchanges of good solution together with their respective context (e.g., memories recording recent behaviour of solution attributes), or a comprehensive history search. Memories recording the performance of individual solutions or solution components may be added to the pool, as well as procedures to generate new solutions or to compute various statistics on solutions, solution components, individual AP performance, and the trajectory of the global search. This information may then be used to build guidance mechanisms or even to feed external learning programs, e.g., neural networks. Not all these ideas have been thoroughly developed and included in the latest methods found in the literature. They constitute an active field of research, however, as illustrated by the two following examples. First, Le Bouthiller, Crainic, and Kropf proposed a dynamically-adaptive learning and guidance mechanism based on atomic elements (e.g., the arcs present in the routes of VRPTW solutions in the pool [22]). Patterns of arcs present in good or bad solutions in the pool are built and are then sent to the individual APs to intensify or diversify the global search. The particular pattern and guidance directive depends upon the stage of the search as measured by the evolution of the elite population in the pool. The mechanism is general in the sense that, being based on atomic elements, it is independent of any particular problem structure. Second, Crainic *et al.* have shown for the first time the capability of memory-based cooperative search to handle complex, multi-characteristic problems [9]. Their study of the design of third-generation wireless networks aims to optimize the number and configuration (location, power and number of antennas, plus the orientation and tilt of each antenna) of base stations to guarantee level of service and minimize the impact of the electromagnetic emissions of the system on human health. The cooperative system involves several tabu search APs to explore particular parts of the solution space where only a few of the configuration parameters are allowed to vary. A genetic algorithm and a path relinking method are then used to combine partial solutions into complete ones and generate new solutions for the pool.

Many interesting research challenges may be identified in relation to these issues. A first group continues the work in designing intelligent cooperation and learning mechanisms to enhance the optimization capabilities of cooperative meta-heuristics. Promising avenues include, but are not limited to, the integration of adaptive and central memory principles, the enhancement of atomic-based guidance, the integration of memory and multi-level search concepts, and the development of advanced learning mechanisms that 1) build a dynamic image of the performance of each AP to, eventually, modify its search parameters or principles, 2) combine statistics (memories) and artificial intelligence methods (e.g., neural networks), and 3) integrate the distributed, i.e., the APs', memories to the global search knowledge and guidance. A second direction aims to build on the capabilities of such intelligent cooperation and learning mechanisms to build characterizations of the solution space already visited and the dynamic performance of the search. This could then be used to steer the global search accordingly as well as to study the linkages between explicit cooperation and emergent computation. Last but not least, it would be very interesting to develop a theory of learning within the context of multi-

AP cooperation. This would provide the means to go beyond the limits of experimental settings in designing effective parallel cooperation search methods for difficult problems. The next section identifies some of these latter possibilities in more detail.

5 Research directions in emergent computation and cooperative search

We believe that research directions on the dynamics and emergent computation behaviour of cooperative search should be inspired by research conducted in other fields, as well as build on the learning mechanisms described in the previous section and on experimental and empirical validation processes for particular problems.

The programming of dynamics in computing systems could offer a first direction. As indicated earlier, the performance of the global exploration of the search space by cooperating APs is obtained from the interplay of a complex system of direct and indirect interactions. So far, however, the design of cooperative algorithms has focused on the components of such systems, e.g., the APs, the cooperation scheme, and the interconnection network, taken individually. Little effort, if any, has been dedicated to developing design strategies of cooperative algorithms by considering, "programming", the system as a whole. This is a bit surprising since dynamic interactions among autonomous computing elements and their potential for emergent computation have been investigated for several computing system contexts. Thus, recurrent neural networks with emergent search behaviour are programmed as a whole by directly adjusting their parameters (e.g., the logical interconnection network, the weights on the interconnection links, and the transition functions on the nodes) based on learning algorithms and adjustment procedures built from the problem optimization model. For cooperative search, this translates into working directly on the logical interconnection network, the cooperation mechanism, the APs, and the asynchronous mode of information exchange, that is work directly on the optimization logic of the cooperation. The methods to perform this global design are an important research topic *per se*, which, for us, is strongly related to the learning issues identified previously.

We thus turn to research areas such as multi-robot systems [2, 3, 6, 23], reactive multi-agent systems [5, 27], artificial life [18, 25, 19] and ad hoc implementations of cooperative algorithms in various computer science applications. Emergent cooperation behaviour has been synthesized in the field of behaviour-based robotics [2, 3, 23] using a methodology denoted behaviour-Based AI, derived from theories on the modular decomposition of intelligence [4, 20]. Behaviour-decomposition methods are first used to analyze observed emergent cooperation behaviours in natural social systems made up of individuals displaying relatively simple behaviours (e.g., ants). Similar cooperative

behaviours are then synthesized in societies of robots [24]. For example, the foraging behaviour of ants has been synthesized into an object-gathering behaviour in social robotics by combining two basic forms of direct interactions among robots (APs), dispersion (interact to diversify the exploration relative to the others) and homing (aim for the goal by, for example, sharing the good solutions) [23]. These results could provide the theoretical foundations for cooperation mechanisms that include more than one form of information sharing strategy among APs which, hopefully, will display global dynamics that adequately approximate the desired emergent problem-solving strategy.

More ideas for research directions come from the field of computing systems where ad hoc bottom-up and emergent computing strategies are increasingly being proposed with significant success for various applications in telecommunication network routing, reliability and power supply limitations in wireless networks, access security to computer systems, and so on. For example, self-assembly in nanothechonology, a bottom-up nanofabrication process in which components self-assemble based on shape complementarity, could provide the basic strategies to specify which interactions are allowed to occur at run time and, thus, to specify the logical interconnection networks. Similarly, the research efforts in autonomic computing, which aims to be able to tell a system **what** to do and let it to find **how** to do it, could maybe inspire a new definition of problem solving for cooperative meta-heuristics where it is the emergent behaviour of the system that finds its way to how address a given problem.

Unlike cellular automata and artificial neural networks, logical interconnection network topologies in cooperative algorithms tend to vary widely. So far, empirically, the best cooperative search results have been obtained through interconnection networks that are configured dynamically at run time, e.g., the memory-based approaches. This suggests adaptive approaches to program system dynamics by adjusting dynamically the network topology of cooperative algorithms to meet the desired attributes of global cooperation. Turning to the learning mechanisms of the previous section, we believe that these or similar learning schemes could be applied to system dynamics. Thus, for example, learning could be used to identify system attractors (regions of the search space to which the search returns often) and the path ways leading to them. The mechanism could then be used to adjust dynamically the interconnection topology by blocking these path ways, and thus prevent the occurrence of some chains of correlated interactions that appear to attract the search is the same region of the search space. Learning could also be used to develop interaction policies that favor those that can lead to the emergence of cooperation and block those that are harmful to it. Intelligent control of the system dynamics of cooperative search through learning mechanisms is certainly one of the most promising research avenues in the effort to obtain a cooperative exploration of the search space through spontaneous interactions among APs.

Research on self-organization often analyze natural systems with observed self-organized behaviours to discover strategies for designing distributed systems with particular glob-

ally emergent behaviours. However, it is well known that computing systems yield spontaneous activities, e.g., the ripple effects described earlier on. This raises the issue whether we should study the dynamic behaviour of cooperative search systems *per se*, as a research object, similarly to biological systems through comprehensive laboratory-based simulations. An associated question is whether we should seek to understand the complex behaviour of cooperative systems in order to sustain emergent cooperation or, rather, should we seek to discover new search heuristics based on occasionally occurring coherent search behaviour at the global level? A combination of these two methodological approaches has proved to be successful. Thus, the study of locally emergent cooperation has led to ideas to design a new cooperation mechanism, which yielded the highly successful multi-level cooperative search method. The final challenge, obviously, is to bring together the research on explicit and implicit cooperation mechanisms and behaviours and apply the resulting methodology within the context of various solution methods. Even though the research in this area is still at the very beginning, this challenge has been met with some success, producing new methods out of the study of cooperative search [28, 12, 26, 11, 22].

6 Conclusion

Cooperative algorithms as computerized problem-solving strategies are becoming ubiquitous in several problem domains, in particular for addressing complex discrete optimization problems. Cooperation has well-known advantages, short development cycle through the re-utilization of existing exact or heuristic methods and high adaptability to different problems and problem characteristics, in particular. Yet, these systems also generate series of indirect interactions that may reduce their performance. We have described cooperation mechanisms, discussed the relations between direct and indirect interactions, and have identified a number of challenges and interesting research directions for the development of the next generation of cooperative search methods.

Acknowledgments

This work was financially supported through the Industrial Research Chair and Discovery grant programs of the Natural Sciences and Engineering Research Council of Canada (NSERC)

References

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

[2] R.C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1998.

[3] Brooks, R.A. Intelligence without Representation. *Artificial Intelligence*, 47(1-3):139–159, 1991.

[4] Brooks, R.A. *Cambrian Intelligence: The arly History of the New AI*. MIT Press, Cambridge, MA, 1999.

[5] Brooks, R.S. A robust layered control system for a mobile robot. In *Readings in Uncertain Reasoning*, pages 204–213. Morgan Kaufmann Publishers Inc., San Francisco, CA, 1990.

[6] Cao, U.Y., Fukunaga, A.S., and Kahng, A.B. Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, 4(1):7–23, 1997.

[7] 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.

[8] Crainic, T.G. Parallel Solution Methods for Vehicle Routing Problems. In Golden, B., Raghavan, S., and Wasil, E., editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, 2007. to appear.

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

[10] Crainic, T.G. and Gendreau, M. Towards an Evolutionary Method - Cooperating Multi-Thread Parallel Tabu Search Hybrid. In S. Voß, S. Martello, C. Roucairol, and Osman, I.H., editors, *Meta-Heuristics 98: Theory & Applications*, pages 331–344. Kluwer Academic Publishers, Norwell, MA, 1999.

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

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] Dagaeff, T. and Chantemargue, F. Performance of Autonomy-based Systems: Tuning Emergent Cooperation. Technical Report 98-20, Computer Science Department, University of Fribourg, 1998.

[19] Engelbrecht, A.P. *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, 2006.

[20] Fodor, J.A. *The Modularity of Mind*. MIT Press, Cambridge, MA, USA, 1983.

[21] 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.

[22] 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.

[23] Mataric, M.J. Designing Emergent Behaviors: From Local Interactions to Collective Intelligence. In *Proceedings of the Second International Conference on From Animals to Animats 2: Simulation of Adaptive Behavior*, pages 432–441. MIT Press, Cambridge, MA, 1993.

[24] Mataric, M.J. Issues and Approaches in the Design of Collective Autonomous Agents. *Robotics and Autonomous Systems*, 16(2-4):321–331, 1995.

[25] Nitschke, G. Emergence of Cooperation: State of the Art. *Artificial Life*, 11(3):367–396, 2005.

[26] 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*, 55(5):1019–1033, 2008.

[27] Oliveira, E.C., Fischer, K., and Stepánková, O. Multi-agent systems: which research for which applications. *Robotics and Autonomous Systems*, 27(1-2):91–106, 1999.

[28] M. Ouyang, M. Toulouse, K. Thulasiraman, F. Glover, and J. S. Deogun. Multilevel cooperative search for the circuit/hypergraph partitioning problem. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(6):685–694, 2002.

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

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

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

[32] 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.

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