Noname manuscript No.
(will be inserted by the editor)

Memetic Cooperative Models for the Tool Switching Problem

Jhon Edgar Amaya · Carlos Cotta · Antonio J. Fernández-Leiva

Received: date / Accepted: date

Abstract This work deals with memetic-computing agent-models based on the cooperative integration of search agents endowed with (possibly different) optimization strategies, in particular memetic algorithms. As a proof-of-concept of the model, we deploy it on the tool switching problem (ToSP), a hard combinatorial optimization problem that arises in the area of flexible manufacturing. The ToSP has been tackled by different algorithmic methods ranging from exact to heuristic methods (including local search metaheuristics, population-based techniques and hybrids thereof, i.e., memetic algorithms). Here we consider an ample number of instances of this cooperative memetic model, whose agents are adapted to cope with this problem. A detailed experimental analysis shows that the meta-models promoting the cooperation among memetic algorithms provide the best overall results compared with their constituent parts (i.e., memetic algorithms and local search approaches). In addition, a parameter sensitivity analysis of the meta-models is developed in order to understand the interplay among the elements of the proposed topologies.

Keywords Memetic computing · Memetic algorithm · Local search · Cooperation · Tool Switching Problem

Jhon Edgar Amaya
Laboratorio de Computación de Alto Rendimiento. Universidad Nacional Experimental del Táchira. Av. Universidad. Paramillo. San Cristóbal. Venezuela
Tel.: +58-276-3552791
Fax: +58-276-3530266
E-mail: jedgar@unet.edu.ve

Carlos Cotta, Antonio J. Fernández-Leiva
Dept. Lenguajes y Ciencias de la Computación, ETSI Informática, University of Málaga, Campus de Teatinos, 29071. Málaga. Spain
Tel.: +34-952-13 7158/3315
Fax: +34-952-13 1397
E-mail: {ccottap, afdez}@lcc.uma.es

1 Introduction

It is well known that exact methods are inherently limited by the complexity of solving to optimality many hard optimization problems. The use of alternative techniques that might eventually overcome this limitation has been thus explored. In particular, the utilization of meta-heuristic techniques has been shown to be specially adequate to attack large instances of such complex optimization problems; their success lies in the fact that they can provide high-quality (near-optimal) solutions in reasonable time, at the expense of not proving their optimality. Of course, in order to achieve this practical effectiveness, meta-heuristics are required to exploit problem-specific knowledge [1–3]. This has been one of the main lessons learned in the last years and has paved the way for designing optimization methods intrinsically concerned with exploiting problem knowledge in order to adjust to the optimization task at hand. Memetic algorithms [4–7] are a prime example of such methods.

Memetic algorithms (MAs) are optimization techniques that blend together ideas from different meta-heuristics, most notably from local-search and population-based techniques. They thus provide a natural framework where diverse search techniques can be integrated. Each of the underlying techniques in a such a hybrid model can provide a different perspective of both the optimization process and the search space, and can actually make a different (complementary) contribution to the search process in terms of intensification/diversification. Thus, by combining the corresponding different exploration patterns, the search can benefit from an increased capability both for finding high-quality solutions and for escaping from local optima. Of course, these capabilities are more useful whenever the problem tackled poses a challenging optimization task to the individual search algorithms. Otherwise, computational power is diversified in unproductive explorations.

From a wider perspective, these collaborative schemes can be considered as "memetic" regardless of whether they adjust or not to the stereotypical model of an evolutionary algorithm endowed with a local search procedure. Certainly, regarding a MA as a collection of individual search agents engaged in periods of isolated search punctuated by cooperation/competition stages is the primeval definition of this paradigm [6, 8]. This original definition –also regarded as the wide interpretation of MAs– also fits within the current notion of memetic computing [9] as a paradigm that uses memes as units of information encoded in computational representations for the purpose of problem-solving, often incarnated in a co-evolving system of intelligent agents. Therein, a meme is interpreted as a lifetime learning procedure capable of improving individual solutions – see [10–15]. Again, a wider interpretation is also possible in this context, more akin to the definition of meme as a unit of cultural information [16]. Such wider interpretation would also include higher-level information units that are neither bound to specific genes nor to the lifetime conservation of inherited genetic information. Thus, memes can be regarded as formae [17] subject to lifetime learning and improvement (a process that can be controlled by the meme itself), thus capturing the plasticity of non-genetic information

and its suitability for Lamarckian transmission. Incidentally, Radcliffe and Surry's work [18] on MAs contributed to settle the classical definition of MAs sketched above.

In retrospect, models adhering to the aforementioned philosophy can be found in the literature of the last two decades. Thus, Toulouse *et al.* [19] considered using multiple instances of tabu search running in parallel, eventually exchanging some of the attributes stored in tabu memory. Later on, Toulouse *et al.* [20] proposed a hierarchical cooperative model based on problem decomposition. Crainic and Gendreau [21] presented a cooperative parallel tabu search method for capacitated network design problem that was shown to outperform independent search strategies. Crainic *et al.* [22] also proposed a method for asynchronous cooperative multi-search using variable neighborhood search with application to the p-median problem. Le Bouthillier and Crainic [23] described a cooperative parallel meta-heuristic for the vehicle routing problem with time windows. Pelta *et al.* [24] presented a cooperative multi-thread search-based optimization strategy, in which several solvers were controlled by a higher-level coordination algorithm which collected information on their search performance and altered the behavior of the solvers accordingly (see also [25]). More recently, Lu *et al.* [26] presented a hybrid cooperative version of quantum particle swam optimization aimed to improving the diversity of the swarms. Multi-agent systems are also well suited in order to implement this kind of cooperative models. This way, Milano and Roli [27] developed a multi-agent system called MAGMA (multiagent meta-heuristic architecture) allowing the use of meta-heuristics at different levels (creating solutions, improving them, defining the search strategy, and coordinating lower-level agents). Malek [28] introduced a multi-agent system like MAGMA which considered particular meta-heuristics implemented by individual agents and the exchange of solutions between these. Also, Sbihi [29] proposed a cooperative local search-based algorithm Multiple-Scenario Max-Min Knapsack Problem. From a more general point of view, an analogy can also be drawn between these models and hyperheuristics [30,31], namely a high-level heuristic that controls the application of a set of low-level heuristics to solutions, using strategies ranging from pure random to performance-based rules. It is also worth mentioning the work on parallel MAs, such as the asynchronous blackboard model of Bradwell and Brown [32] –where a genetic algorithm and tabu search cooperate using a shared blackboard as a means for communication (note the connection with A-Teams as well [33])– or island-based (and in general spatially-structured) MAs [34–36]. Particularly in [34] it was noted that each individual in the population could use a different local improvement method, very much in line with heterogenous memetic models.

This paper deals with the uniform tool switching problem (ToSP), a hard combinatorial optimization problem that appears in Flexible Manufacturing Systems (FMSs), an alternative to rigid production systems that has the capability to be adjusted for generating different products and/or for changing the order of product generation. Different examples of the problem can be

found in diverse areas such as electronics manufacturing, metalworking industry, computer memory management, and aeronautics, among others [37–40].

This problem represents a challenging task that has already been attacked by many different optimization methods including exact methods ranging from integer linear programming (ILP) techniques to heuristic constructive algorithms. References to the problem can be found as early as in the 1960s [37]; since then, it has been tackled via many different techniques [38,39]. Tang *et al.* [39] proposed an ILP formulation of the problem, and [38] formulated the ToSP as a non-linear integer program with a dual-based relaxation heuristic. More recently, [41] proposed two exact algorithms: a branch-and-bound approach and a linear programming-based branch-and-cut algorithm.

Despite the moderate success of exact methods, it must be noted that they are inherently limited, since [42] and [43] proved formally that the ToSP is NP-hard in general. This limitation was already highlighted by Laporte *et al.* [41] who reported that their algorithm was capable of dealing with instances with 9 jobs, but provided very low success ratios for instances with more than 10 jobs. Some *ad hoc* heuristics have been devised in response to this complexity barrier (e.g., [40,44,45]). Meta-heuristics have been also used to solve the ToSP [46,47]. For example, a version based in tabu search (TS) [47] provided good-quality results; Al-Fawzan and Al-Sultan defined three different versions of TS that arose from the inclusion of different algorithmic mechanisms such as long-term memory and oscillation strategies. A different, and very interesting, approach has been described by [48], who proposed a beam search algorithm. Beam search (BS) is a derivate of branch-and-bound that uses a breadth-first traversal of the search tree, and incorporates a heuristic choice to keep at each level only the best (according to some *quality* measure) β nodes (β is the so-called *beam width*).

In [49] we proposed three methods to tackle the ToSP: a simple local search (LS) scheme based on hill climbing, a genetic algorithm and a MA based on the hybridization of the two latter methods. This MA represented a killer approach for the ToSP. More recently, in [50] we proceeded along the memetic computing avenue by considering composite models in which different local search techniques cooperate for solving the ToSP. A number of different topologies for agent communication were considered. These cooperatives models showed a good performance, better that their constituent parts working alone and statistically comparable to the state-of-the-art approach (i.e., the MA presented in [49]). To the best of our knowledge, this was the first attempt of such a memetic scheme to tackle the ToSP.

Now, this paper continues in the search of more complex memetic computing models, and here we propose a multi-level memetic approach in which collaborating agents are themselves memetic (hence the model can be considered in some sense meta-cooperative, as the individual agents are cooperative techniques themselves) and analyze the effect of using different interconnection topologies. The goal of this paper is thus to synergistically combine the cooperative model presented in [50] with the powerful and flexibility of classical MAs, exploiting the capability of composite schemes for identifying probably

good regions of the search space and the potential of MAs for exploring and exploiting these.

We consider here different topologies for the communication of our memetic agents; to analyze the goodness of our cooperative memetic model we consider the ToSP and test the performance of a wide number of cooperative MAs (i.e., also called indistinctly meta-cooperative algorithms) that were naturally generated as instances of the proposed model. Moreover, the resulting collaborative algorithms depend on several parameters that can be tuned to control the memetic model, and hence we have conducted a sensitivity analysis of the parameters, in the solving of the ToSP, that can shed some light on the inner working of the meta-models.

The remainder of the paper is organized as follows: Sect. 2 provides an overview of the ToSP; then in Sect. 3 we detail the architecture of the memetic model considered, as well as the constituent algorithms involved in it; subsequently, Sect. 4 describes the experiments performed and their results, including the sensitivity analysis mentioned before; Sect. 5 closes the paper with conclusions and some prospects for future developments.

2 The Tool Switching Problem

Before proceeding, let us firstly describe more in depth the ToSP. As already mentioned the uniform tool switching problem (ToSP) is a hard combinatorial optimization problem that appears in Flexible Manufacturing Systems (FMSs). This problem arises in a single machine that has several slots into which different tools can be loaded. Each slot just admits one tool, and each job executed on that machine requires a particular set of tools to be completed. Jobs are sequentially executed, and therefore each time a job is to be processed, the corresponding tools must be loaded in the machine magazine. The ToSP consists of finding an appropriate job sequence in which jobs will be executed, and an associated sequence of tool loading/unloading operations that minimizes the number of tool switches in the magazine.

There are two major elements in the problem: a machine M and a collection of jobs $J = \{J_1, \cdots, J_n\}$ to be processed. Regarding the latter, the relevant information for the optimization process is the tool requirements for each job. We assume that there is a set of tools $T = \{\tau_1, \cdots, \tau_m\}$, and that each job J_i requires a certain subset $T^{(J_i)} \subseteq T$ of tools to be processed. As to the machine, we will just consider one piece of information: the capacity C of the magazine (i.e., the number of available slots). Given the previous elements, we can formalize the ToSP as follows: let a ToSP instance be represented by a pair, $I = \langle C, A \rangle$ where C denotes the magazine capacity, and A is a $m \times n$ binary matrix that defines the tool requirements to execute each job, i.e., $A_{ij} = 1$ if, and only if, tool τ_i is required to execute job J_j.

We assume that $C < m$; otherwise the problem is trivial. The solution to such an instance is a sequence $\langle J_{i_1}, \cdots, J_{i_n} \rangle$ (where $i_1, \ldots, i_n$ is a permutation of numbers $1, \ldots, n$) determining the order in which the jobs are executed, and

a sequence $T_1, \cdots, T_n$ of tool configurations ($T_i \subset T$) determining which tools are loaded in the magazine at a certain time. Note that for this sequence of tool configurations to be feasible, it must hold that $T^{(J_{i_j})} \subseteq T_j$.

Let $\mathbb{N}_h^+ = \{1, \cdots, h\}$ henceforth. We will index jobs and tools with integers from $\mathbb{N}_n^+$ and $\mathbb{N}_m^+$ respectively. An ILP formulation for the ToSP is shown below, using two sets of zero-one decision variables – x_{jk} ($j \in \mathbb{N}_n^+$, $k \in \mathbb{N}_n^+$), and y_{ik} ($i \in \mathbb{N}_m^+$, $k \in \mathbb{N}_n^+$) – that respectively indicate whether a job j is executed at time k or not, or whether a tool τ_i is in the magazine at time k or not. Notice that since each job makes exclusive use of the machine, time-step k can be assimilated to the time at which the kth job is executed.

In order to process each job, a particular collection of tools is required to be loaded in the magazine. Obviously, it is assumed that no job requires a number of tools higher than the magazine capacity, i.e., $\sum_{i=1}^{m} A_{ij} \leqslant C$ for all $j \in \mathbb{N}_n^+$. Tool requirements are reflected in Eq. (5). Following [38], we assume the initial condition $y_{i0} = 1$ for all $i \in \mathbb{N}_m^+$. This initial condition amounts to the fact that the initial loading of the magazine is not considered as part of the cost of the solution (in fact, no actual switching is required for this initial load). The objective function $F(\cdot)$ counts the number of switches that have to be done for a particular job sequence:

$$\min \; F(y) = \sum_{k=1}^{n} \sum_{i=1}^{m} y_{ik}(1 - y_{i,k-1}) \tag{1}$$

$$\forall j \in \mathbb{N}_n^+ : \; \sum_{k=1}^{n} x_{jk} = 1 \tag{2}$$

$$\forall k \in \mathbb{N}_n^+ : \; \sum_{j=1}^{n} x_{jk} = 1 \tag{3}$$

$$\forall k \in \mathbb{N}_n^+ : \; \sum_{i=1}^{m} y_{ik} \leqslant C \tag{4}$$

$$\forall j, k \in \mathbb{N}_n^+ \; \forall i \in \mathbb{N}_m^+ : \; A_{ij} x_{jk} \leqslant y_{ik} \tag{5}$$

$$\forall j, k \in \mathbb{N}_n^+ \; \forall i \in \mathbb{N}_m^+ : \; x_{jk}, y_{ij} \in \{0, 1\} \tag{6}$$

This general definition shown above corresponds to the *uniform ToSP* in which each tool fits in just one slot. The ToSP can be divided into three subproblems [51]: the first subproblem is *machine loading* and consists of determining the sequence of jobs; the second subproblem is *tool loading*, consisting of determining which tool to switch (if a switch is needed) before processing a job; finally, the third subproblem is *slot loading*, and consists of deciding where (i.e., in which slot) to place each tool. Since we are considering the uniform ToSP, the third subproblem does not apply (all slots are identical, and the order of tools is irrelevant). Moreover, and without loss of generality, the cost of switching a tool is considered constant (the same for all tools) in the uniform ToSP. Under this assumption, the tool loading subproblem can also be obviated because if the job sequence is fixed, the optimal tool switching policy can be determined in polynomial time using a greedy procedure termed *Keep Tool Needed Soonest* (KTNS) [38,39]. Therefore, we can concentrate on

the machine loading subproblem, and use KTNS as a subordinate procedure to solve the subsequent tool loading subproblem.

3 Model Architecture

We have considered four memetic cooperative architectures in which agents attached to a certain spatial structure are endowed with a LS mechanism or a MA. These architectures are defined on the basis of the particular optimization methods used, and thus in the following we describe the local search methods considered as well as the basic MAs that are part of the cooperative model. The architectures also depend on their interaction topology and thus this aspect is defined below.

3.1 Local searchers

LS meta-heuristics are based on exploring the neighborhood of a certain "current" solution. It is thus convenient to address firstly the representation of solutions and the structure of this neighborhood, and subsequently of the underlying search space. A permutational encoding arises as the natural way to represent solutions. Thus, a candidate solution for a specific ToSP instance $I = \langle C, A\rangle$ is simply a permutation $\pi = \langle \pi_1, \cdots, \pi_n\rangle \in \mathbb{P}_n$ where $\pi_i \in \mathbb{N}_n^+$, and $\mathbb{P}_n$ is the set of all permutations of elements in $\mathbb{N}_n^+$. The KTNS algorithm is used to obtain the actual tool configuration of the machine for the corresponding job sequence.

Having defined the representation, we now turn our attention to the neighborhood structure. In general, we have considered the well-known *swap* neighborhood $\mathcal{N}_{swap}(\cdot)$, in which two permutations are neighbors if they just differ in two positions of the sequence, that is, for a permutation $\pi \in \mathbb{P}_n$, $\mathcal{N}_{swap}(\pi) = \{\pi' \in \mathbb{P}_n \mid H(\pi, \pi') = 2\}$ where $H(\pi, \pi') = n - \sum_{i=1}^{n}[\pi_i = \pi'_i]$ is the Hamming distance between sequences π and π' (the number of positions in which the sequences differ), and $[\cdot]$ is Iverson bracket (i.e., $[P] = 1$ if P is true, and $[P] = 0$ otherwise). Given the permutational nature of sequences, this implies that the contents of the two differing positions have been swapped. For some specific applications (named when necessary), we have also considered a specific neighborhood called *block* neighborhood $\mathcal{N}_{block}(\cdot)$. This is a generalization of the swap neighborhood in which two non-overlapping blocks (i.e., subsequences of adjacent positions) of a randomly chosen length $b_l \in \mathbb{N}_{n/2}^+$ are selected at random within a permutation, and swapped.

These neighborhoods are exploited within two different LS frameworks. The first one is *steepest-ascent Hill Climbing* (HC), in which given a current solution π, its neighborhood $\mathcal{N}(\pi)$ is explored, and the best solution found is taken as the new current solution, provided it is better than the current one (ties are randomly broken). If no such neighboring solution exist, the search is considered stagnated, and can be restarted from a different initial point.

Algorithm 1: Pseudocode of MA

```
for i ∈ N_μ^+ do
    pop[i] ←RANDOM-SOLUTION();
    LOCAL-IMPROVEMENT (pop[i]);
end for
i ← 0;
while i < MaxEvals do
    RANK-POPULATION (pop); // sort population according to fitness
    parent_1 ←SELECT (pop);
    if Rand[0,1] < p_X then // recombination is done
        parent_2 ← SELECT (pop);
        child ← RECOMBINE (parent_1, parent_2);
    else
        child ← parent_1;
    end if
    child ← MUTATE (child, p_M); // p_M is the mutation probability per gene
    if Rand[0,1] < p_LS then // LS is applied
        LOCAL-IMPROVEMENT (child); // Local Improvement
    end if
    pop[μ] ← child; // replace worst
end while
return best solution in pop;
```

The second LS technique considered is a *Tabu Search* (TS) method along the lines of the proposal in [47]. This TS method is based on a strategic oscillation mechanism which switches between the two neighborhoods defined before. A deterministic criterion based on switching the neighborhood structure after a fixed number of iterations was reported by [47] to perform better than a probabilistic criterion (i.e., choosing the neighborhood structure in each step, according to a certain probability distribution). Given the nature of the two neighborhoods considered (the block neighborhood being a huge superset of the swap neighborhood), this TS algorithm has some resemblance with variable neighborhood search (VNS) [52]: a VNS approach could be actually defined on the basis of nested neighborhoods considering blocks of increasing size (the swap neighborhood being the base case). We implement a long term memory scheme using a frequency based memory structure with a mechanism based in swapping to select new candidate solutions [47]. No aspiration criterion is used in this referred algorithm.

3.2 Basic Memetic Algorithms

The simplest form of MA that we consider here consists of endowing a genetic algorithm (GA) with a local search method. The general scheme of this MA is shown in Figure 1.

With respect to the GA, we have considered a steady-state genetic algorithm to evolve promising job sequences: a single solution is generated in each generation, and inserted in the population replacing the worst individual. Se-

lection is done by binary tournament. As to recombination, we have opted for using alternating position crossover (APX) [53], an operator based on the creation the offspring by selecting alternately the next element of the first parent and the next element of the second parent, omitting elements already present in the offspring. Notice that this operator is not transmitting positional information [17], hence including an additional degree of diversification. Other choices of recombination operator are possible (see [54] for a description of numerous permutation-based recombination operators).

For the purposes of mutation we have considered the *block* neighborhood (the same as in our proposals described in [49]). In all our meta-heuristics proposals (i.e., HC, TS, GA and MA) the fitness of the candidate is obtained by the value returned after applying the KTNS method to the candidate. The objective is thus minimizing this value.

Different versions of this MA were analyzed by considering the two LS techniques described previously (note that in [49] we only considered a hill climbing method). The LS methods have been applied just after the mutation stage. Local search is applied to any individual with a probability p_{LS}; in case of application, the improvement uses up a number of $Evals_{LS}$ evaluations (or in the case of HC until it stagnates, whatever comes first). Notice that this improvement strategy is intentionally simple. Indeed there exists ample literature on the use of more sophisticated strategies to determine to which individuals should be applied, or which improvement strategy or neighbor structure to use if several are available [55,56].

3.3 Interaction Topology for the Cooperative Memetic Models

Let R be an architecture with n agents; each agent a_i ($0 \leqslant i < n$) in R consists of one of the meta-heuristics described in Sections 3.1 and 3.2. These agents engage in periods of isolated exploration followed by synchronous communication. We denote as Θ the maximum number of such exploration/communication cycles in a certain cooperative model. Also, let S_i be the pool of solution candidates associated to agent a_i (i.e., if the agent is loaded with a LS method then $\#S_i = 1$, and if the agent is loaded with a population based method – e.g., a MA– then $\#S_i \geqslant 1$, where $\#S_i$ represents the cardinality of S_i), and let $\mathbf{T}_R \subseteq \mathbb{N}_n^+ \times \mathbb{N}_n^+$ be the communication topology over R (i.e., if $(i,j) \in \mathbf{T}_R$ then a_i can send information to agent a_j). The general architecture of the model is then described in Algorithm 2. Firstly all the agents are initialized with random solution(s) (lines 1-3). Then, the algorithm is executed for a maximum number of iteration cycles (lines 5-15) where, in each cycle, the search technique kept in each agent is executed to update its associated pool of solutions (lines 6-8; note that if the agent contains an LS method this basically means an improvement of its incumbent solution but if the agent contains a population based method then a new pool of solutions is generated), and solutions are fed from an agent to another according to the topology considered (lines 9-13). Note that an agent only accepts an incoming solution if it is better than

Algorithm 2: COOPERATIVE-MODEL$_n$

```
1  for i ∈ ℕ_n^+ do
2  |  S_i ← GENERATEINITIALPOPULATION();
3  end for
4  cycles ← 1;
5  while cycles ⩽ Θ do
6  |  for i ∈ ℕ_n^+ do
7  |  |  S_i ← a_i(S_i); // Population update
8  |  end for
9  |  for (i, j) ∈ T_R do
10 |  |  if KTNS(BEST(S_i)) < KTNS(BEST(S_j)) then
11 |  |  |  S_j ← S_j ∪ {BEST(S_i)} \ {WORST(S_j)}; // new solution accepted
12 |  |  end if
13 |  end for
14 |  cycles ← cycles + 1;
15 end while
16 return arg min{KTNS(BEST(S_i)) | i ∈ ℕ_n^+};
```

the best solution contained in its corresponding pool of candidates. Note also that other acceptance criteria are possible; in fact we also tested empirically a model in which an agent accepted an incoming solution only if this was better than the *worst* solution in its candidate pool but the algorithms devised from this model performed worse. Observe that, for a maximum number of evaluations $E_{\max}$ and for a specific number of cycles Θ, each cycle in our cooperative algorithms spends $E_{cycle} = E_{\max}/\Theta$ evaluations, and the specific search method of any agent takes E_{cycle}/n evaluations at most.

Three strategies based on different interaction topologies are considered (Figure 1 shows a graphical representation of these topologies):

- RING: $\mathbf{T}_R = \{(i, i(n)+1) \mid i \in \mathbb{N}_n^+ \text{ and } i(n) \text{ denotes } i \text{ modulo } n\}$. Thus, there exists a circular list of agents in which each node only sends (resp. receives) information to its successor (resp. from its predecessor).
- BROADCAST: $\mathbf{T}_R = \mathbb{N}_n^+ \times \mathbb{N}_n^+$, i.e., a *go with the winners*-like topology in which the best overall solution at each synchronization point is transmitted to all agents. This means all agents execute intensification over the same local region of the search space at the beginning of each cycle.
- RANDOM: $\mathbf{T}_R$ is composed by n pairs (i, j) that are randomly sampled from $\mathbb{N}_n^+ \times \mathbb{N}_n^+$. This sampling is done each time communication takes place, and hence any two agents might eventually communicate in any step.

The topologies mentioned above can be regarded as a simple way of coordinating the functioning of the different agents. It is possible to conceive dynamic topologies which change in time in response to the evolution of the search process, trying to adapt the whole model to the needs of the former. In some sense, the adaptive mechanisms described in [56] for determining the choice of memes to be applied can be cast upon this model in terms of plugging/unplugging particular agents depending upon some measure of the state of the search, e.g., genetic diversity [57], fitness diversity [58–61], etc. As an

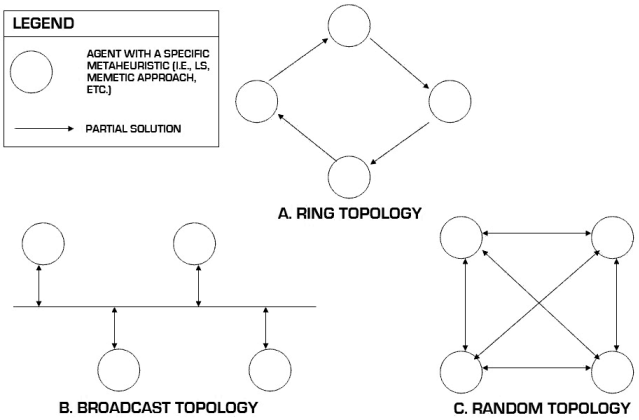

Fig. 1 Considered topologies in the memetic cooperative model. Agents are represented by circles and arrows show the channel (and flow) of communication among them.

aside remark, ideas from co-evolutionary MAs [14] naturally adapt to these spatially-structured models, allowing to devise a rich variety of pairing strategies between solutions and memes.

In addition to these strategies we have considered a so-called Ring SDI model, based on an interesting proposal described in [62] and that we also considered in [50]. SDI stands for Search, Diversification and Intensification, and hence the SDI architecture consists of three agents dedicated to different purposes: the first one to local search, the second one to diversification and the third one to intensification.

4 Computational Results

As far as we know, no standard data instance exists for this problem (at least publicly available) so that we have selected a wide set of problem instances that were attacked in [38,44,47,48]; more specifically, 16 instances were chosen with values for the number of jobs, number of tools, and machine capacity ranging in [10,50], [9,60] and [4,30] respectively. Table 1 shows the different problem instances chosen for the experimental evaluation where a specific instance with n jobs, m tools and machine capacity C is labeled as $C\zeta_n^m$.

Five different datasets[1] (i.e., incident matrixes or relations among tools and jobs) were generated randomly per instance. Each dataset was generated with the restriction, already imposed in previous works such as [44], that no job is *covered* by any other job in the sense that $\forall i, j \in \mathbb{N}_n^+, i \neq j, T^{(J_i)} \not\subseteq T^{(J_j)}$. The reason to enforce this constraint is to avoid the simplification of the problem by preprocessing techniques as done for instance in [38] and [48].

The experiments have been performed using a wide set of different algorithms that were devised from the beam search (BS) presented in [48], the two

[1] All datasets are available at *http://www.unet.edu.ve/∼jedgar/ToSP/ToSP.htm*

Table 1 Problem Instances considered in the experimental evaluation. The minimum and maximum of tools required for all the jobs is indicated in second and third rows respectively. Fourth row shows the work from which the problem instance was obtained.

	$4\zeta_{10}^{10}$	$4\zeta_{10}^{9}$	$6\zeta_{10}^{15}$	$6\zeta_{15}^{12}$	$6\zeta_{15}^{20}$	$8\zeta_{20}^{15}$	$8\zeta_{20}^{16}$	$10\zeta_{20}^{20}$
Min.	9	9	11	4	6	6	7	9
Max.	24	24	30	10	15	15	20	20
Source	[47],[44]	[38],[48]	[48]	[38],[48]	[44]	[47]	[38],[48]	[38],[48]

	$10\zeta_{30}^{25}$	$15\zeta_{30}^{40}$	$15\zeta_{40}^{30}$	$20\zeta_{40}^{60}$	$24\zeta_{20}^{30}$	$24\zeta_{20}^{36}$	$25\zeta_{50}^{40}$	$30\zeta_{20}^{40}$
Min.	4	6	6	7	9	9	9	11
Max.	10	15	15	20	24	24	20	30
Source	[47]	[44]	[47]	[44]	[38],[48]	[38],[48]	[47]	[48]

LS methods described in Section 3.1, the MA scheme shown in Section 3.2 and the cooperative scheme shown in Section 3.3. From these a wide number of algorithms were devised and tested. For instance, in the case of BS, five different values $\beta \in \mathbb{N}_5^+$ were considered for the beam width. Regarding LS methods, we consider the TS and HC as described previously. We take into account also LS versions in which a partial exploration of the neighborhood was done by obtaining a fixed-size random sample; in particular, the size of this sample has been chosen to be αn, i.e., proportional to the number of jobs (the value $\alpha = 4$ has been used). The notation HCP and HCF (resp. TSP and TSF) was used to indicate the HC variant (resp. TS variant) in which the neighborhood was partially or fully explored respectively. Also, in the case of HC, the search is restarted from a different initial point if stagnation takes place before consuming the allotted number of evaluations. Regarding TS, the tabu tenure is 5, and the number of iterations on each neighborhood for performing strategic oscillation is 3. In both cases, this corresponds to the setting used in [47]. The memetic approaches (MAHCP and MATSP) consist of a combination of a GA with HCP/TSP where HCP/TSP was always applied to each offspring generated after the mutation step (i.e., $p_{LS} = 1.0$). The choice of parameter values of the MA scheme shown in Algorithm 1 was done after an extensive phase of experimentation with many different values. The best combinations of the values were finally selected; the parameters were identical to those used in [49], i.e., $\mu = 30$, $p_X = 1.0$, and $p_M = 1/n$ where n is the number of jobs. Due to the high number of possible combinations for the cooperative model, we only display the results obtained by the algorithms that employed LS techniques with partial neighborhood exploration since the results shown in [50] (as well as preliminary experiments conducted on the memetic schemas presented here) suggest these algorithms perform better that those based on LS with a full neighborhood exploration schema.

Regarding the cooperative memetic model, we have considered a topology with 3 agents and have tested 36 different scenarios of the cooperative model: in our experiments we use the following notation $\boldsymbol{\Theta}\mathbf{T}(\mathbf{a_1}, \mathbf{a_2}, \mathbf{a_3})$ for the cooperative memetic versions where $\Theta \in \{4, 5\}$ represents the number of cycles, $T \in \{$Broadcast (Br), Random (Ra), Ring (Ri)$\}$ the topology of the model,

and $\mathbf{a}_i \in$ {MATSP, MAHCP,HCP, TSP} the optimization methods loaded in agent i (for $1 \leq i \leq 3$). In addition we include in our experiments the Ring SDI model where the GA described in Section 3.2 was used for diversification, HCP as local searcher, and the Kick Operator (KO) [62] –a first-ascent HC on the swap neighborhood– for intensification; this model, denoted here as **Θ**Ri(GA,HCP,KO), was considered because in [50] it was illustrated that its performance was comparable to that shown by the MAHCP approach as no significant statistical difference between these two approaches was perceived; moreover the Ring SDI model provided the best results among the cooperative proposals shown in [50]. Precisely, for this reason the remaining memetic models proposed in [50] are not illustrated here.

All algorithms were run 10 times (per instance and dataset) for a maximum of $E_{\max} = \varphi n(m - C)$ evaluations[2] per run (with $\varphi > 0$). Preliminary experiments on the value of φ proved that $\varphi = 100$ is an appropriate value that allows to keep an acceptable relation between solution quality and computational cost. Regarding the BS algorithm, because of its deterministic nature, just one execution per dataset (and per value of beam width) was run and the algorithm was allowed to be executed until exhaustion (i.e., until completing the search). Tables 4-8 in the appendix show the computational results obtained by the different algorithms in each of the instances considered in the experiments.

Given the ample number of algorithms and instances considered, we have used a rank-based approach to analyze globally the performance of each technique. To do so, we have computed the rank r_j^i of each algorithm j on each instance i (rank 1 for the best, and rank k for the worst, where $k = 49$ is the number of algorithms; in case of ties, an average rank is awarded). The distribution of these ranks is shown in Fig. 2. As it can be seen several cooperative MAs outperform the MAHCP approach, the previous incumbent for this problem.

Next, we have used two well-known non-parametric statistical tests [63] to compare ranks, namely Friedman test [64] and Iman-Davenport test [65]. The results are shown in Table 2. As seen in the first row, the statistic values obtained are clearly higher than the critical values, and therefore the null hypothesis, namely that all algorithms are equivalent, can be rejected. Since there are algorithms with markedly poor performance, we have repeated the test with the top 16 algorithms (i.e., those that occupy the sixteen lower positions at the bottom of Fig. 2). Again, it can be seen that the statistical test is passed, thus indicating significant differences in their ranks at the standard $\alpha = 0.05$ level.

Subsequently, we have focused in these top 16 algorithms, and performed Holm's test [66] in order to determine whether there exists significant differences with respect to a control algorithm (in this case 4Ra(MAHCP,MAHCP,MAHCP), the algorithm with the best mean rank

[2] Observe that the number of evaluations increases with the number of tools (assumed to be directly related with problem difficulty) and decreases when the magazine capacity increases (that, in some sense, it is also inversely related to the problem difficulty).

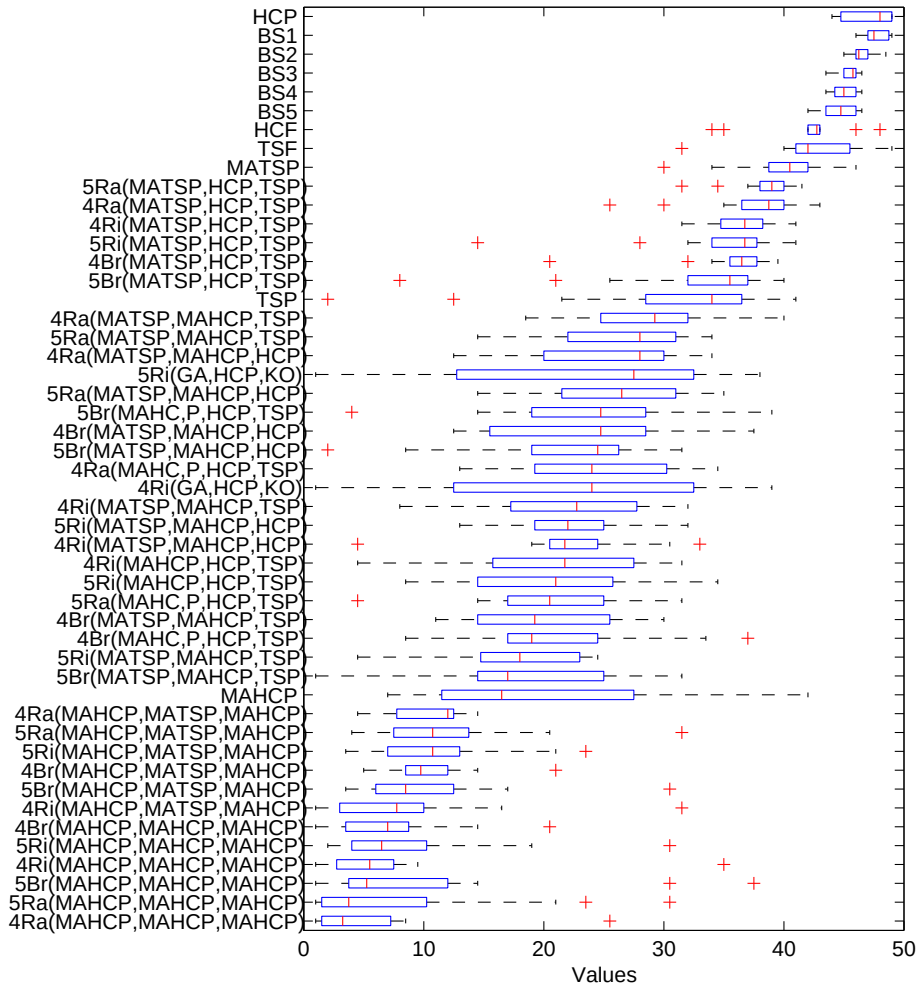

Fig. 2 Rank distribution of each algorithm across all instances. As usual, each box comprises the second and third quartiles of the distribution, the median is marked with a vertical line, whiskers span 1.5 times the inter-quartile range, and outliers are indicated with a plus sign.

acording to Fig. 2). The results are shown in Table 3 where we can notice that there is no statistical difference between the six first cooperative MAs, but there does exist a statistical difference with respect to MAHCP, the previous best-known algorithm to solve the ToSP. This seems to fundament the validity of our cooperative memetic model.

In order to understand the sensitivity of the cooperative memetic model we conducted next an experimental analysis to ascertain adequate values for the main parameters of the model, namely the number of agents and the number

Table 2 Results of Friedman and Iman-Davenport tests.

	Friedman value	critical χ^2 value	Iman-Davenport value	critical F_F value
all	603.67	65.17	55.10	1.38
top 16	66.88	24.99	13.01	1.71

Table 3 Results of Holm's test using 4Ra(MAHCP,MAHCP,MAHCP) as control algorithm.

i	algorithm	z-statistic	p-value	α/i
1	5Ra(MAHCP,MAHCP,MAHCP)	0.8540	0.1966	0.05
2	4Ri(MAHCP,MAHCP,MAHCP)	0.8726	0.1914	0.025
3	4Br(MAHCP,MAHCP,MAHCP)	1.1882	0.1174	0.0167
4	4Ri(MAHCP,MATSP,MAHCP)	1.5966	0.0552	0.0125
5	5Ri(MAHCP,MAHCP,MAHCP)	1.7266	0.0421	0.01
6	5Br(MAHCP,MAHCP,MAHCP)	1.7266	0.0421	0.0083
7	5Br(MAHCP,MATSP,MAHCP)	2.7848	0.0027	0.0071
8	5Ri(MAHCP,MATSP,MAHCP)	3.0447	0.0012	0.0063
9	4Br(MAHCP,MATSP,MAHCP)	3.1190	0.0009	0.0056
10	5Ra(MAHCP,MATSP,MAHCP)	3.2489	0.0006	0.005
11	4Ra(MAHCP,MATSP,MAHCP)	3.2675	0.0005	0.0045
12	5Br(MATSP,MAHCP,TSP)	5.1426	< 0.0001	0.0042
13	5Ri(MATSP,MAHCP,TSP)	5.2540	< 0.0001	0.0038
14	MAHCP	5.4396	< 0.0001	0.0036
15	4Br(MATSP,MAHCP,TSP)	5.8852	< 0.0001	0.0033

of cycles of communication among them. To do so, we experiment with 5 of the largest instances considered here for ToSP, i.e., $10\zeta_{30}^{25}$, $15\zeta_{30}^{40}$, $15\zeta_{40}^{30}$, $20\zeta_{40}^{60}$ and $25\zeta_{50}^{40}$ and 5 datasets per instance. According to Figure 2, there is a model whose instances systematically show the best performance; this cooperative memetic model corresponds with the schema ΘT(MAHCP,MAHCP,MAHCP), that is to say, a 3-agent topology in which each agent is endowed with the previous incumbent for the problem; moreover according to Table 3 there is no statistical difference if we consider 4 or 5 cycles. Thus to determine an appropriate value for the number of agents we have considered the algorithms $5\mathrm{T}(\mathrm{a}_1,\ldots,\mathrm{a}_n)$ where T is one of the topologies considered here, the number of cycles Θ was set to 5, n (i.e., the number of agents) ranges in $[2,6]$, and each agent a_i is endowed with the previous incumbent algorithm MAHCP. The results for the three algorithms devised from this model (i.e., one for each topology) are shown in Figure 3 in which the y-axis represents the average error (i.e., the average difference to the best solution known for each dataset). In average (see the right bottom graphic) the 3-agent model seems to be the most adequate for this problem independently of the topology, although the 2-agent model behaves better in the broadcast and random topologies. In any case, the model clearly degrades for values of n above 3.

There is thus a trend that indicates the memetic model performs better with a lower number of agents, and its performance degrades when the number of agents increases. This might indicate that the higher the number of agents, the less intensified the exploitation process developed in each of the agent is.

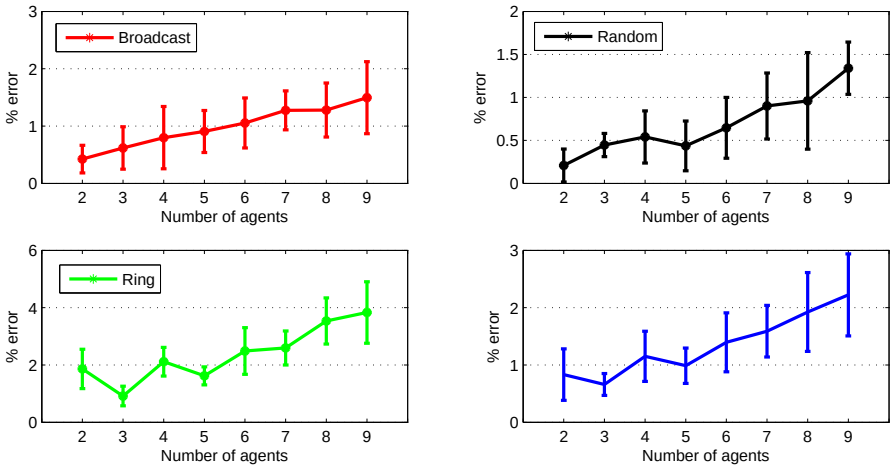

Fig. 3 Influence of the number of agents in the best cooperative model: Average error obtained by considering the instances $10\zeta_{30}^{25}$, $15\zeta_{30}^{40}$, $15\zeta_{40}^{30}$, $20\zeta_{40}^{60}$ and $25\zeta_{50}^{40}$ and 5 datasets per instance. Upper graphics and left bottom graphic correspond to the results obtained using the algorithms $5\mathrm{T}(\mathrm{a}_1,\ldots,\mathrm{a}_n)$ being $T \in \{$BROADCAST (Br), RANDOM (Ra), RING (Ri)$\}$ the topology of the model, n the number of agents, and a_i = MAHCP the optimization method loaded in each agent i (for $1 \leq i \leq n$). Right bottom graphic shows the average result considering the three algorithms.

This basically means that the effort carried out during the optimization is too scattered, so that the search mechanism in each agent has not enough resources (i.e., time or number of evaluations) to advance properly in its own search process.

Once we have analyzed how the number of agents affects performance, we have conducted a similar experiment with respect to the number of cycles and have studied how this influences the results; to do so, the best cooperative model according to the analysis previously conducted on the number of agents, that is to say, the 3-agent model ΘT(MAHCP,MAHCP,MAHCP) has been used. For each topology value $T \in \{$BROADCAST (Br), RANDOM (Ra), RING (Ri)$\}$, ten different instances of this model have been considered, corresponding to different values of $\Theta \in \mathbb{N}_{10}^{+}$. Figure 4 depicts the results; again the average error with respect to best value found considering the 5 instances mentioned previously and 5 datasets per instance is shown. In general the model performs better for a low number of cycles and 3 cycles fixes the threshold over which the model starts to degrade. The reason that we find to explain this degradation is similar to the one given for the number of agents, that is to say, a higher number of cycles clearly disfavors the intensification that each agent produces in an autonomous way and thus its contribution to the whole system is lower. Also, in average, decreasing the number of cycles might have a negative effect because the exploration process is also affected (i.e., agents are activated a lower number of times) and, as a consequence, the whole model

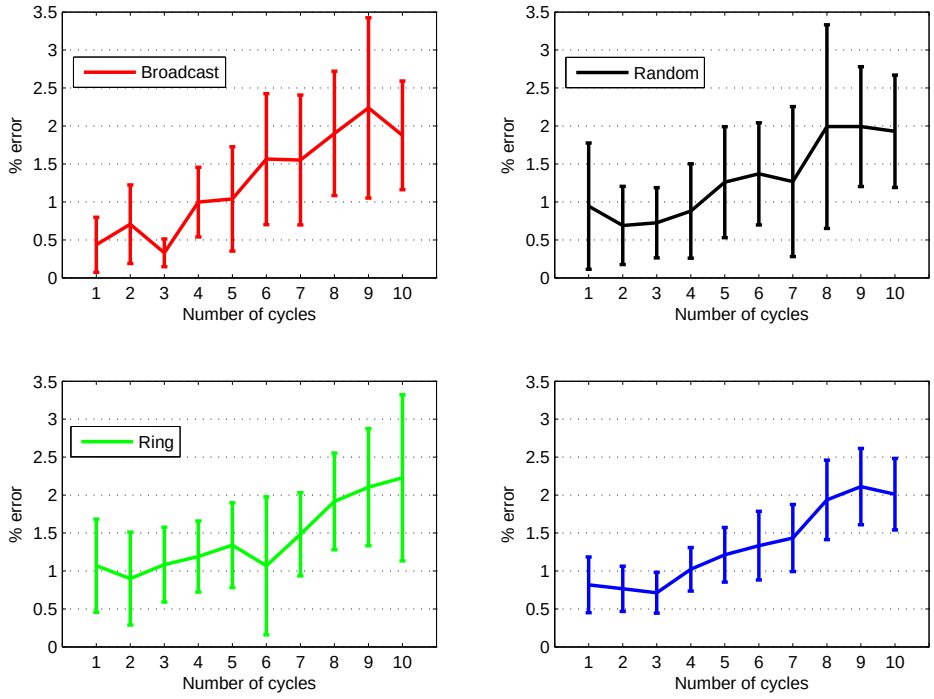

Fig. 4 Influence of the number of cycles in the best cooperative model: Average error obtained by considering the instances $10\zeta_{30}^{25}$, $15\zeta_{30}^{40}$, $15\zeta_{40}^{30}$, $20\zeta_{40}^{60}$ and $25\zeta_{50}^{40}$ and 5 datasets per instance. Upper graphics and left bottom graphic correspond to the results obtained using the algorithms ΘT(MAHCP,MAHCP,MAHCP) for different values of $\Theta \in [1, 10]$ and being $T \in \{$Broadcast (Br), Random (Ra), Ring (Ri)$\}$ the topology of the model, Right bottom graphic shows the average result considering the three algorithms associated to the different topologies.

is less cooperative in the sense that the flow of information between agents decreases. Indeed, this behavior is analogous to that observed in distributed evolutionary algorithms with varying communication frequency [67] – see also [36].

5 Conclusions

In this work we have described a generic schema for the cooperation of MAs. This schema consists basically of an architecture where n agents (possibly memetic themselves) interchange periods of communication with isolated executions. These resulting model is aimed at combining synergistically the capabilities of different meta-heuristics (most notably, MAs), diversifying the search and providing a new improved balance with intensification processes.

Also, as it was argued in the introduction section, and according to the definition given in [9], the spatial structure of our cooperative model is by itself a memetic computing structure and thus it might be catalogued as a MA (a superset of parallel MAs actually).

To demonstrate the feasibility and effectiveness of our proposal we have tackled a hard combinatorial optimization problem, namely the tool switching problem, by considering a high number of cooperative MAs that were produced as instances (specifically adjusted to cope with this problem) of our generic schema. To this end, we have considered up to 36 memetic cooperative models, differing in the particular combination of meta-heuristics assigned to the agents and their connection topology. Fully memetic models –i.e., cooperative models in which each agent is endowed with a (possibly different) MA– are shown to provide the best results, significantly outperforming a standalone MA which was the previous best-known algorithm for this problem. A sensitivity analysis conducted to ascertain the different trade-offs involved in the cooperative model indicates that a relatively low number of agents with moderate isolation provides the best results.

Note that our proposed schema is generic and can be adapted to handle distinct combinatorial optimization problems; in fact, many different instances –as shown in the optimization of the tool switching problem– can be devised from the schema to solve a specific optimization problem. The nature of these instances can range from completely heterogeneous models (i.e., those in which the algorithm loaded in each of the agents is different one from each others) to fully homogeneous models (i.e., those in which all the agents support the same algorithm). In fact, one of the lessons that one can learn from the analysis conducted here is that it is good to use heterogeneous agents (the composite models are shown to perform better than the individual agents); it is likely to find an appropriate combination of them and an adequate balance in the general model between exploration and exploitation, and this can surely work even if all or most of agents are homogeneous, provided these are extremely good as individual solvers (as it is the case of MAHCP for the TosP). Actually, there exists in this sense some connections with techniques such as VNS, hyperheuristics, and multimeme/adaptive MAs, namely the need for integrating several approaches or selectively using a mixture of algorithms in order to boost search capabilities. The model presented here is a general complementary approach to any of those mentioned before, and can be actually used in conjunction with them, thus enlarging the arsenal of techniques available for this purpose.

Future work will be directed along two major directions: considering additional techniques in the arsenal available to each of the agents in the model, and moving to different problem domains. In both cases one of the main goals is determining the extent to which the performance trends observed are also present in these new scenarios. As a second further step, we plan to consider more sophisticated collaborative models in which high-level features such as the intervening techniques or the underlying topology may change dynamically, either in response to some predefined agent-internal logic, under the

control of a global model manager, or as the result of a self-adaptive process [68,69].

Acknowledgements Thanks are due to the anonymous reviewers for useful suggestions which have allowed us to improve the focus and readability of the paper. The second and third authors were partially supported by Spanish MICINN project under contract TIN2008-05941 (NEMESIS project) and by project TIC-6083 of Junta de Andalucía.

References

1. Hart, W.E., Belew, R.K.: Optimizing an arbitrary function is hard for the genetic algorithm. In Belew, R.K., Booker, L.B., eds.: 4^{th} International Conference on Genetic Algorithms, San Mateo CA, Morgan Kaufmann (1991) 190–195
2. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation **1(1)** (1997) 67–82
3. Bonissone, P., Subbu, R., Eklund, N., Kiehl, T.: Evolutionary algorithms + domain knowledge = real-world evolutionary computation. IEEE Transactions on Evolutionary Computation **10**(3) (2006) 256–280
4. Hart, W., Krasnogor, N., Smith, J.: Recent Advances in Memetic Algorithms. Volume 166 of Studies in Fuzziness and Soft Computing. Springer-Verlag, Berlin Heidelberg (2005)
5. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: Model, taxonomy, and design issues. IEEE Transactions on Evolutionary Computation **9**(5) (2005) 474–488
6. Moscato, P.: On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Technical Report Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA (1989)
7. Moscato, P., Cotta, C.: A gentle introduction to memetic algorithms. In Glover, F., Kochenberger, G., eds.: Handbook of Metaheuristics. Kluwer Academic Publishers, Boston MA (2003) 105–144
8. Moscato, P.: Memetic algorithms: A short introduction. In Corne, D., Dorigo, M., Glover, F., eds.: New Ideas in Optimization. McGraw-Hill (1999) 219–234
9. Ong, Y.S., Lim, M.H., Che, X.: Memetic computation – past, present & future. IEEE Computational Intelligence Magazine **5**(2) (2010) 24–36
10. Krasnogor, N., Blackburne, B., Burke, E., Hirst, J.: Multimeme algorithms for protein structure prediction. In Merelo, J., et al., eds.: Parallel Problem Solving From Nature VII. Volume 2439 of Lecture Notes in Computer Science. Springer-Verlag, Berlin (2002) 769–778
11. Smith, J.: Co-evolution of memetic algorithms: Initial investigations. In Merelo, J., et al., eds.: Parallel Problem Solving From Nature VII. Volume 2439 of Lecture Notes in Computer Science. Springer-Verlag, Berlin (2002) 537–548
12. Krasnogor, N.: Self generating metaheuristics in bioinformatics: The proteins structure comparison case. Genetic Programming and Evolvable Machines **5**(2) (2004) 181–201
13. Krasnogor, N., Gustafson, S.: A study on the use of "self-generation" in memetic algorithms. Natural Computing **3**(1) (2004) 53–76
14. Smith, J.: Coevolving memetic algorithms: A review and progress report. IEEE Transactions on Systems, Man, and Cybernetics, Part B **37**(1) (2007) 6–17
15. Smith, J.E.: Credit assignment in adaptive memetic algorithms. In Lipson, H., ed.: GECCO '07: Proceedings of the 9th annual conference on Genetic and Evolutionary Computation Conference, ACM Press (2007) 1412–1419
16. Dawkins, R.: The Selfish Gene. Clarendon Press, Oxford (1976)
17. Radcliffe, N.: The algebra of genetic algorithms. Annals of Mathematics and Artificial Intelligence **10** (1994) 339–384

18. Radcliffe, N., Surry, P.: Formal Memetic Algorithms. In Fogarty, T., ed.: Evolutionary Computing: AISB Workshop. Volume 865 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (1994) 1–16
19. Toulouse, M., Crainic, T.G., Sanso, B., Thulasiraman, K.: Self-organization in cooperative tabu search algorithms. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics. Volume 3. (1998) 2379–2384
20. Toulouse, M., Thulasiraman, K., Glover, F.: Multi-level cooperative search: A new paradigm for combinatorial optimization and an application to graph partitioning. In Amestoy, P., Berger, P., Dayd, M., Ruiz, D., Duff, I., Frayss, V., Giraud, L., eds.: Euro-Par'99 Parallel Processing. Volume 1685 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (1999) 533–542
21. Crainic, T.G., Gendreau, M.: Cooperative parallel tabu search for capacitated network design. Journal of Heuristics **8**(6) (2002) 601–627
22. Crainic, T.G., Gendreau, M., Hansen, P., Mladenović, N.: Cooperative parallel variable neighborhood search for the p-median. Journal of Heuristics **10** (2004) 293–314
23. LeBouthillier, A., Crainic, T.G.: A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. Computers & Operations Research **32**(7) (2005) 1685 – 1708
24. Pelta, D., Cruz, C., Sancho-Royo, A., Verdegay, J.: Using memory and fuzzy rules in a co-operative multi-thread strategy for optimization. Information Sciences **176** (2006) 1849–1868
25. Cruz, C., Pelta, D.: Soft computing and cooperative strategies for optimization. Applied Soft Computing **9**(1) (2009) 30 – 38
26. Lu, S., Sun, C.: Coevolutionary quantum-behaved particle swarm optimization with hybrid cooperative search. In: Computational Intelligence and Industrial Application, 2008. PACIIA '08. Pacific-Asia Workshop on. Volume 1. (2008) 109 –113
27. Milano, M., Roli, A.: Magma: a multiagent architecture for metaheuristics. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **34**(2) (2004) 925 – 941
28. Malek, R.: Collaboration of metaheuristic algorithms through a multi-agent system. In Mark, V., Strasser, T., Zoitl, A., eds.: Holonic and Multi-Agent Systems for Manufacturing. Volume 5696 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2009) 72–81
29. Sbihi, A.: A cooperative local search-based algorithm for the multiple-scenario max-min knapsack problem. European Journal of Operational Research **202**(2) (2010) 339 – 346
30. Cowling, P., Kendall, G., Soubeiga, E.: A hyperheuristic approach to schedule a sales submit. In Burke, E., Erben, W., eds.: PATAT 2000. Volume 2079 of Lecture Notes in Computer Science., Berlin Heidelberg, Springer-Verlag (2008) 176–190
31. Chakhlevitch, K., Cowling, P.: Hyperheuristics: Recent developments. In Cotta, C., Sevaux, M., Sörensen, K., eds.: Adaptive and Multilevel Metaheuristics. Volume 136 of Studies in Computational Intelligence. Springer-Verlag, Berlin Heidelberg (2008) 3–29
32. Bradwell, R., Brown, K.N.: Parallel asynchronous memetic algorithms. In: Evolutionary Computation and Parallel Processing Workshop – GECCO 1999, Orlando FL (1999)
33. Talukdar, S.N.: Collaboration rules for autonomous software agents. Decision Support Systems **24**(3-4) (1999) 269 – 278
34. Mühlenbein, H.: Evolution in Time and Space – The Parallel Genetic Algorithm. In Rawlins, G.J., ed.: Foundations of Genetic Algorithms, Morgan Kaufmann Publishers (1991) 316–337
35. Cotta, C., Mendes, A., Garcia, V., França, P., Moscato, P.: Applying memetic algorithms to the analysis of microarray data. In Raidl, G., et al., eds.: Applications of Evolutionary Computing. Volume 2611 of Lecture Notes in Computer Science. Springer-Verlag, Berlin (2003) 22–32
36. Tang, J., Lim, M., Ong, Y., Er, M.: Study of migration topology in island model parallel hybrid-ga for large scale quadratic assignment problems. In: Control, Automation, Robotics and Vision Conference, 2004. Volume 3. (2004) 2286 – 2291
37. Belady, L.: A study of replacement algorithms for virtual storage computers. IBM Systems Journal **5** (1966) 78–101
38. Bard, J.F.: A heuristic for minimizing the number of tool switches on a flexible machine. IIE Transactions **20**(4) (1988) 382–391

39. Tang, C., Denardo, E.: Models arising from a flexible manufacturing machine, part I: minimization of the number of tool switches. Operations Research **36**(5) (1988) 767–777
40. Shirazi, R., Frizelle, G.: Minimizing the number of tool switches on a flexible machine: an empirical study. International Journal of Production Research **39**(15) (2001) 3547–3560
41. Laporte, G., Salazar-González, J., Semet, F.: Exact algorithms for the job sequencing and tool switching problem. IIE Transactions **36**(1) (2004) 37–45
42. Oerlemans, A.: Production planning for flexible manufacturing systems. Ph.d. dissertation, University of Limburg, Maastricht, Limburg, Netherlands (1992)
43. Crama, Y., Kolen, A., Oerlemans, A., Spieksma, F.: Minimizing the number of tool switches on a flexible machine. International Journal of Flexible Manufacturing Systems **6** (1994) 33–54
44. Hertz, A., Laporte, G., Mittaz, M., Stecke, K.: Heuristics for minimizing tool switches when scheduling part types on a flexible machine. IIE Transactions **30** (1998) 689–694
45. Djellab, H., Djellab, K., Gourgand, M.: A new heuristic based on a hypergraph representation for the tool switching problem. International Journal of Production Economics **64**(1-3) (2000) 165–176
46. Hertz, A., Widmer, M.: An improved tabu search approach for solving the job shop scheduling problem with tooling constraints. Discrete Applied Mathematics **65** (1993) 319–345
47. Al-Fawzan, M., Al-Sultan, K.: A tabu search based algorithm for minimizing the number of tool switches on a flexible machine. Computers & Industrial Engineering **44**(1) (2003) 35–47
48. Zhou, B.H., Xi, L.F., Cao, Y.S.: A beam-search-based algorithm for the tool switching problem on a flexible machine. The International Journal of Advanced Manufacturing Technology **25**(9-10) (2005) 876–882
49. Amaya, J.E., Cotta, C., Fernández, A.J.: A memetic algorithm for the tool switching problem. In Blesa, M., Blum, C., Cotta, C., Fernández, A., Gallardo, J., Roli, A., Sampels, M., eds.: Hybrid Metaheuristics 2008. Volume 5296 of Lecture Notes in Computer Science., Berlin Heidelberg, Springer-Verlag (2008) 190–202
50. Amaya, J.E., Cotta, C., Fernández, A.J.: Hybrid cooperation models for the tool switching problem. In González, J., Pelta, D., Cruz, C., Terrazas, G., Krasnogor, N., eds.: Nature Inspired Cooperative Strategies for Optimization (NICSO 2010). Volume 284 of Studies in Computational Intelligence. Springer-Verlag, Berlin Heidelberg (2010) 39–52
51. Tzur, M., Altman, A.: Minimization of tool switches for a flexible manufacturing machine with slot assignment of different tool sizes. IIE Transactions **36**(2) (2004) 95–110
52. Mladenovic, N., Hansen, P.: Variable neighborhood search. Computers & Operations Research **24**(11) (1997) 1097 – 1100
53. Larrañaga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., Dizdarevic, S.: Genetic algorithms for the travelling salesman problem: A review of representations and operators. Articial Intelligence Review **13** (1999) 129–170
54. Cotta, C., Troya, J.: Genetic forma recombination in permutation flowshop problems. Evolutionary Computation **6**(1) (1998) 25–44
55. Ong, Y.S., Keane, A.: Meta-lamarckian learning in memetic algorithms. IEEE Transactions on Evolutionary Computation **8**(2) (2004) 99–110
56. Ong, Y.S., Lim, M.H., Zhu, N., Wong, K.: Classification of adaptive memetic algorithms: a comparative study. IEEE Transactions on Systems, Man, and Cybernetics, Part B **36**(1) (2006) 141–152
57. Tang, J., Lim, M., Ong, Y.: Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. Soft Computing **11** (2007) 873–888
58. Neri, F., Toivanen, J., Mäkinen, R.A.E.: An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for hiv. Applied Intelligence **27**(3) (2007) 219–235
59. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for online and offline control design of pmsm drives. IEEE Transactions on Systems, Man, and Cybernetics, Part B **37**(1) (2007) 28–41
60. Tirronen, V., Neri, F., Kärkkäinen, T., Majava, K., Rossi, T.: An enhanced memetic differential evolution in filter design for defect detection in paper production. Evolutionary Computation **16**(4) (2008) 529–555

61. Caponio, A., Neri, F., Tirronen, V.: Super-fit control adaptation in memetic differential evolution frameworks. Soft Comput. **13**(8-9) (2009) 811–831
62. Talbi, E.G., Bachelet, V.: Cosearch: A parallel cooperative metaheuristic. Journal of Mathematical Modelling and Algorithms **5**(1) (2006) 5–22
63. Lehmann, E., D'Abrera, H.: Nonparametrics: statistical methods based on ranks. Prentice-Hall, Englewood Cliffs, NJ (1998)
64. Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association **32**(200) (1937) 675–701
65. Iman, R., Davenport, J.: Approximations of the critical region of the Friedman statistic. Communications in Statistics **9** (1980) 571–595
66. Holm, S.: A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics **6** (1979) 65–70
67. Alba, E., Troya, J.M.: Influence of the migration policy in parallel distributed gas with structured and panmictic populations. Applied Intelligence **12** (2000) 163–181
68. Krause, W., Sollacher, R., Greiner, M.: Self-⋆ topology control in wireless multihop ad hoc communciations networks. In Babaoglu, O., et al., eds.: Self-star Properties in Complex Information Systems. Volume 3460 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg (2005) 49–62
69. Smith, J.E.: Self-adaptation in evolutionary algorithms for combinatorial optimisation. In Cotta, C., Sevaux, M., Sörensen, K., eds.: Adaptive and Multilevel Metaheuristics. Volume 136 of Studies in Computational Intelligence. Springer (2008) 31–57

A Tables of Results

Table 4 Computational results: Beam search, LS methods, and MA versions

		B=1	B=2	B=3	B=4	B=5	HCF	HCP	TSF	TSP	MAHCP	MATSP
$4\zeta_{10}^{9}$	mean	8.4	8.4	8.4	8.4	8.4	8.4	8.4	8.12	8.08	8.1	8.14
	σ	0.49	0.49	0.49	0.49	0.49	1.11	1.0	0.77	0.74	0.75	0.87
$4\zeta_{10}^{10}$	mean	10.0	9.8	9.6	9.6	9.6	9.34	9.6	9.06	8.8	8.94	9.08
	σ	2.1	1.83	2.06	2.06	2.06	1.56	1.57	1.58	1.61	1.62	1.66
$6\zeta_{10}^{15}$	mean	15.2	14.8	14.8	14.8	14.8	14.38	14.7	13.82	13.68	13.89	13.86
	σ	1.47	1.47	1.47	1.47	1.47	2.22	2.25	2.0	2.1	1.99	2.03
$6\zeta_{15}^{12}$	mean	18.2	17.6	17.6	17.4	17.4	18.32	20.1	17.08	16.46	16.26	16.9
	σ	0.75	1.02	1.02	1.2	1.2	1.71	2.05	2.09	1.93	1.79	2.07
$6\zeta_{15}^{20}$	mean	26.2	25.8	25.2	25.2	25.2	24.76	26.54	23.4	23.02	23.18	23.14
	σ	2.32	2.14	1.6	1.6	1.6	2.35	2.4	1.97	2.0	1.96	2.01
$8\zeta_{20}^{15}$	mean	27.0	26.0	25.6	25.2	25.2	24.46	28.9	24.3	23.62	22.86	24.46
	σ	3.95	4.05	4.27	4.12	4.12	3.29	4.17	3.79	3.63	3.41	3.62
$8\zeta_{20}^{16}$	mean	29.4	29.4	29.4	29.4	29.4	28.76	33.78	28.76	27.92	27.24	28.66
	σ	1.62	1.62	1.62	1.62	1.62	1.49	2.48	2.07	2.13	2.22	2.08
$10\zeta_{20}^{20}$	mean	34.2	33.6	33.4	33.4	33.4	34.4	37.46	31.78	30.72	30.53	31.38
	σ	3.19	2.87	2.8	2.8	2.8	1.64	2.88	2.46	2.5	2.49	2.33
$10\zeta_{30}^{25}$	mean	73.6	70.8	70.8	70.8	70.6	69.6	85.46	85.34	67.72	64.32	68.24
	σ	1.02	1.47	1.47	1.47	1.5	1.02	2.97	12.72	1.52	2.4	1.84
$15\zeta_{30}^{40}$	mean	111.6	110.0	109.2	107.8	107.8	103.0	121.2	104.28	101.72	99.7	102.34
	σ	15.19	13.55	13.41	13.26	13.26	14.03	14.61	13.44	13.07	12.82	12.46
$15\zeta_{40}^{30}$	mean	105.2	103.2	102.8	102.8	102.4	100.34	127.54	130.5	101.9	95.86	103.3
	σ	8.52	9.58	9.74	9.74	9.97	8.84	9.51	17.39	8.14	7.52	8.26
$20\zeta_{40}^{60}$	mean	221.8	220.0	218.8	218.6	218.6	214.42	248.7	255.8	213.74	206.3	212.2
	σ	7.03	6.16	5.71	5.82	5.82	9.88	10.17	39.84	8.38	8.81	8.73
$24\zeta_{20}^{30}$	mean	33.0	32.6	32.4	32.4	32.4	25.8	30.24	25.72	25.04	24.78	25.76
	σ	4.43	4.5	4.63	4.63	4.63	2.71	3.69	3.33	3.02	3.29	3.55
$24\zeta_{20}^{36}$	mean	54.0	54.0	53.8	53.8	53.4	48.48	53.24	47.04	45.9	44.87	46.71
	σ	8.2	8.2	8.33	8.33	7.61	8.69	9.33	8.84	8.98	7.55	9.4
$25\zeta_{50}^{40}$	mean	167.2	164.0	162.8	162.8	161.8	150.88	191.02	196.28	153.58	144.18	157.44
	σ	12.91	12.55	12.34	12.34	12.16	15.29	13.36	31.63	12.89	11.94	12.07
$30\zeta_{20}^{40}$	mean	52.2	50.2	50.2	50.2	50.2	44.4	49.12	42.82	42.12	41.3	42.06
	σ	6.24	7.03	7.03	7.03	7.03	5.78	5.29	4.68	4.34	4.41	5.17

Table 5 Computational results: Broadcast topology

$4\zeta_{10}^{9}$	mean	7.94	7.98	8.0	8.04	7.94	8.04	8.0	8.02	7.98	7.94	7.96	7.92	4Br(MAHCP,HCP,TSP)
	σ	0.79	0.81	0.82	0.85	0.73	0.82	0.75	0.79	0.73	0.79	0.77	0.8	
$4\zeta_{10}^{10}$	mean	8.8	8.86	8.8	8.94	8.74	8.74	8.94	8.92	8.8	8.88	8.88	8.74	5Br(MAHCP,HCP,TSP)
	σ	1.74	1.66	1.7	1.64	1.66	1.61	1.65	1.65	1.71	1.68	1.54	1.63	
$6\zeta_{10}^{15}$	mean	13.84	13.8	13.74	13.74	13.76	13.72	13.84	13.76	13.8	13.74	13.74	13.82	4Br(MAHCP,MAHCP,MAHCP)
	σ	2.07	2.1	2.04	2.05	2.09	2.05	2.06	2.11	2.05	2.15	2.05	2.03	
$6\zeta_{15}^{12}$	mean	16.22	16.38	16.0	15.9	16.1	15.94	16.86	16.86	16.54	16.28	16.28	16.22	5Br(MAHCP,MAHCP,MAHCP)
	σ	1.89	1.81	1.82	1.86	1.82	1.96	2.14	1.79	2.06	1.91	2.02	1.89	
$6\zeta_{15}^{20}$	mean	23.04	23.26	22.74	22.94	22.78	22.78	23.16	23.38	23.22	23.06	22.96	23.08	4Br(MAHCP,MATSP,MAHCP)
	σ	2.04	1.93	1.96	1.99	1.92	1.91	2.03	1.95	1.88	2.02	1.95	2.07	
$8\zeta_{20}^{15}$	mean	23.2	23.16	22.68	22.74	22.76	23.0	23.9	24.16	23.52	23.52	23.18	23.22	5Br(MAHCP,MATSP,MAHCP)
	σ	3.6	3.57	3.33	3.35	3.39	3.44	3.77	3.65	3.3	3.49	3.33	3.55	
$8\zeta_{20}^{16}$	mean	27.26	27.26	26.54	26.78	26.86	27.12	28.34	28.14	27.32	27.26	27.28	27.36	4Br(MATSP,HCP,TSP)
	σ	2.31	2.02	2.05	2.08	2.14	2.02	2.32	2.37	2.15	2.14	2.27	2.18	
$10\zeta_{20}^{20}$	mean	30.84	30.78	30.02	29.9	30.22	30.16	31.22	31.42	30.72	30.7	30.36	30.5	5Br(MATSP,HCP,TSP)
	σ	2.36	2.5	2.45	2.66	2.31	2.38	2.51	2.37	2.58	2.53	2.52	2.35	
$10\zeta_{30}^{25}$	mean	65.36	65.5	64.36	63.7	64.52	64.58	68.22	68.18	65.96	65.86	65.06	65.2	4Br(MATSP,MAHCP,HCP)
	σ	2.32	2.31	1.94	2.34	2.06	1.89	2.43	2.42	2.44	2.07	2.85	1.96	
$15\zeta_{30}^{40}$	mean	99.84	100.54	98.4	98.7	99.58	99.2	101.58	101.24	100.0	100.26	100.14	100.98	5Br(MATSP,MAHCP,HCP)
	σ	13.07	12.6	12.44	12.61	12.35	12.8	13.17	12.32	12.6	12.37	12.61	12.79	
$15\zeta_{40}^{30}$	mean	97.42	97.46	95.56	95.1	96.28	95.1	101.04	100.12	96.86	97.62	97.14	97.7	4Br(MATSP,MAHCP,TSP)
	σ	7.54	7.0	7.38	7.71	7.23	6.95	8.17	7.89	7.54	8.07	8.22	7.92	
$20\zeta_{40}^{60}$	mean	206.88	207.51	204.92	204.86	205.92	206.2	211.42	210.78	206.22	206.84	207.9	206.64	5Br(MATSP,MAHCP,TSP)
	σ	7.98	8.58	8.16	7.98	9.41	8.41	8.96	8.88	9.02	8.23	7.71	7.72	
$24\zeta_{20}^{30}$	mean	24.86	24.74	24.2	24.46	24.44	24.34	25.68	25.6	24.98	24.9	24.98	24.62	
	σ	3.22	3.01	3.27	3.34	3.18	3.05	3.43	3.38	3.5	3.14	3.15	3.35	
$24\zeta_{20}^{36}$	mean	45.18	44.4	44.58	44.66	44.94	45.0	46.02	44.65	45.28	44.13	45.46	43.7	
	σ	8.53	8.34	8.22	8.1	8.53	8.1	8.87	8.48	8.06	8.45	8.38	8.18	
$25\zeta_{50}^{40}$	mean	145.72	147.1	143.48	143.16	144.02	143.38	151.46	152.35	145.72	147.05	145.84	145.83	
	σ	12.24	11.51	12.3	11.84	12.23	12.31	13.03	12.73	11.19	12.27	12.68	13.24	
$30\zeta_{20}^{40}$	mean	41.56	41.7	40.9	40.92	41.0	40.86	42.36	42.28	41.42	41.56	41.28	41.36	
	σ	4.25	4.66	4.01	4.62	4.85	4.25	4.64	4.82	4.56	4.53	4.51	4.07	

Table 6 Computational results: random topology

		4Ra(MAHCP,HCP,TSP)	5Ra(MAHCP,HCP,TSP)	4Ra(MAHCP,MAHCP,MAHCP)	5Ra(MAHCP,MAHCP,MAHCP)	4Ra(MAHCP,MATSP,MAHCP)	5Ra(MAHCP,MATSP,MAHCP)	4Ra(MATSP,HCP,TSP)	5Ra(MATSP,HCP,TSP)	4Ra(MATSP,MAHCP,HCP)	5Ra(MATSP,MAHCP,HCP)	4Ra(MATSP,MAHCP,TSP)	5Ra(MATSP,MAHCP,TSP)
$4\varsigma_{10}^{9}$	mean	8.0	7.98	8.02	8.04	7.92	8.0	8.02	8.08	8.02	8.06	8.06	8.0
	σ	0.69	0.76	0.84	0.87	0.74	0.82	0.81	0.77	0.73	0.83	0.79	0.75
$4\varsigma_{10}^{10}$	mean	8.9	8.9	8.76	8.86	8.8	8.8	8.94	8.92	8.8	8.82	8.96	8.86
	σ	1.7	1.77	1.61	1.66	1.65	1.6	1.75	1.72	1.66	1.63	1.67	1.71
$6\varsigma_{10}^{15}$	mean	13.82	13.7	13.72	13.76	13.72	13.82	13.86	13.82	13.74	13.74	13.78	13.74
	σ	2.11	2.08	2.13	2.04	2.13	1.97	2.04	2.03	2.12	2.06	2.01	2.06
$6\varsigma_{15}^{12}$	mean	16.48	16.24	15.8	16.08	16.12	16.04	16.94	16.9	16.62	16.36	16.4	16.26
	σ	1.88	1.97	1.71	1.79	1.95	2.03	2.23	2.16	1.81	1.95	1.7	1.71
$6\varsigma_{15}^{20}$	mean	23.18	22.98	22.64	22.56	22.76	22.92	23.14	23.4	23.0	23.0	23.04	23.04
	σ	2.04	2.11	1.92	1.89	1.87	1.85	1.93	2.13	1.91	1.85	1.96	2.0
$8\varsigma_{20}^{15}$	mean	23.34	23.16	22.54	22.7	22.96	22.92	24.16	24.28	23.38	23.48	23.4	23.54
	σ	3.6	3.56	3.44	3.42	3.44	3.52	3.75	3.53	3.5	3.61	3.5	3.5
$8\varsigma_{20}^{16}$	mean	27.28	27.3	26.86	26.48	27.1	26.74	28.24	28.44	27.36	27.18	27.3	27.38
	σ	2.26	2.06	2.06	2.27	2.15	2.13	2.46	2.51	2.1	2.43	2.04	2.12
$10\varsigma_{20}^{20}$	mean	30.38	30.8	29.78	29.9	29.94	30.16	31.44	31.42	30.62	30.84	30.64	30.76
	σ	2.45	2.19	2.52	2.34	2.63	2.34	2.5	2.43	2.28	2.45	2.36	2.35
$10\varsigma_{30}^{25}$	mean	66.02	65.6	64.02	63.94	65.18	64.92	68.78	68.56	65.54	66.28	66.56	65.98
	σ	2.63	2.22	2.03	1.7	2.22	1.98	1.95	1.73	2.49	2.32	2.45	2.56
$15\varsigma_{30}^{40}$	mean	99.54	99.8	98.24	98.64	99.12	99.14	102.08	102.86	100.8	100.18	101.75	100.98
	σ	12.92	12.59	13.06	12.4	13.13	12.99	13.42	13.15	12.48	13.03	13.02	13.14
$15\varsigma_{40}^{30}$	mean	97.6	97.4	94.86	94.54	96.24	95.26	101.48	101.06	98.64	97.8	98.5	97.74
	σ	7.76	8.18	7.41	7.93	7.76	7.93	8.5	8.23	8.57	7.84	8.06	7.65
$20\varsigma_{40}^{60}$	mean	206.56	208.29	203.76	204.12	206.1	205.66	212.44	211.64	206.74	207.44	207.9	207.78
	σ	7.5	9.86	7.71	8.31	8.55	8.27	7.52	8.87	8.91	8.09	8.02	9.68
$24\varsigma_{20}^{30}$	mean	24.92	24.72	24.42	24.36	24.56	24.66	25.82	25.68	24.98	25.06	24.94	24.78
	σ	3.16	3.19	3.29	3.27	3.01	3.5	3.49	3.39	3.35	3.47	3.37	3.35
$24\varsigma_{20}^{36}$	mean	45.16	45.38	44.44	44.42	44.94	44.98	46.62	46.28	45.56	45.42	45.42	45.86
	σ	8.49	8.32	7.87	8.1	8.21	7.98	8.94	9.01	8.57	8.26	8.12	8.32
$25\varsigma_{50}^{40}$	mean	146.92	146.43	143.26	143.08	144.42	143.92	151.18	151.5	147.16	148.32	148.78	148.28
	σ	12.61	11.39	12.79	11.75	12.42	12.76	13.39	12.04	13.06	12.2	13.49	11.82
$30\varsigma_{20}^{40}$	mean	41.44	41.48	40.74	41.04	40.88	40.8	42.34	42.38	41.62	41.52	41.4	41.72
	σ	4.59	4.61	4.39	4.36	4.63	4.4	4.9	4.56	4.41	4.81	4.43	4.6

Table 7 Computational results: ring topology

$4\varsigma_{10}^{9}$	mean	7.92	7.94	8.06	8.04	7.9	7.98	8.08	7.98	8.04	8.04	8.0	7.98	7.9	7.92	4Ri(MAHCP,HCP,TSP)
	σ	0.82	0.79	0.76	0.85	0.81	0.79	0.8	0.79	0.75	0.82	0.8	0.73	0.73	0.69	
$4\varsigma_{10}^{10}$	mean	8.86	8.8	8.72	8.84	8.78	8.86	8.9	8.88	8.84	8.84	8.9	8.86	8.72	8.7	5Ri(MAHCP,HCP,TSP)
	σ	1.65	1.69	1.69	1.59	1.72	1.71	1.71	1.77	1.71	1.72	1.63	1.69	1.65	1.62	4Ri(MAHCP,MAHCP,MAHCP)
$6\varsigma_{10}^{15}$	mean	13.82	13.76	13.72	13.7	13.82	13.76	13.88	13.84	13.7	13.78	13.82	13.7	13.66	13.74	5Ri(MAHCP,MAHCP,MAHCP)
	σ	2.12	2.1	2.11	2.08	2.01	2.11	2.12	2.01	2.18	2.07	2.03	2.06	2.12	2.07	4Ri(MAHCP,MATSP,MAHCP)
$6\varsigma_{15}^{12}$	mean	16.32	16.32	15.92	15.82	15.92	16.12	16.8	17.06	16.26	16.22	16.4	16.28	16.16	16.1	5Ri(MAHCP,MATSP,MAHCP)
	σ	1.87	2.07	1.83	1.71	1.86	1.82	2.13	2.15	2.04	1.79	1.91	1.77	1.9	2.02	4Ri(MATSP,HCP,TSP)
$6\varsigma_{15}^{20}$	mean	23.14	23.18	22.72	22.86	22.98	22.84	23.18	23.22	23.02	23.0	22.92	22.82	22.96	23.1	5Ri(MATSP,HCP,TSP)
	σ	1.99	1.99	1.99	1.83	1.92	2.04	2.02	2.11	2.02	2.03	1.99	2.17	2.14	2.08	4Ri(MATSP,MAHCP,HCP)
$8\varsigma_{20}^{15}$	mean	22.96	23.18	22.88	22.92	22.66	22.9	24.22	23.8	23.26	23.3	23.32	23.02	23.06	23.2	5Ri(MATSP,MAHCP,HCP)
	σ	3.41	3.49	3.52	3.38	3.22	3.37	4.05	3.67	3.49	3.76	3.52	3.72	3.34	3.42	4Ri(MATSP,MAHCP,TSP)
$8\varsigma_{20}^{16}$	mean	27.18	27.34	26.84	26.7	26.96	26.78	28.4	28.4	27.46	27.3	27.14	27.08	27.44	27.28	5Ri(MATSP,MAHCP,TSP)
	σ	2.44	2.55	2.12	2.17	2.07	1.96	2.78	2.08	2.21	2.22	2.14	2.12	2.05	2.3	4Ri(GA,HCP,KO)
$10\varsigma_{20}^{20}$	mean	30.6	30.46	29.7	30.12	30.18	30.26	31.34	31.34	30.58	30.68	30.82	30.48	30.72	30.74	5Ri(GA,HCP,KO)
	σ	2.45	2.32	2.24	2.54	2.16	2.38	2.46	2.84	2.32	2.6	2.5	2.74	2.41	2.38	
$10\varsigma_{30}^{25}$	mean	65.34	65.76	64.24	64.4	64.2	64.3	68.18	67.5	65.8	65.56	65.36	65.52	67.32	68.1	
	σ	2.32	2.01	1.8	1.84	2.23	1.93	2.21	2.47	2.22	2.26	2.08	2.86	2.69	2.76	
$15\varsigma_{30}^{40}$	mean	99.82	100.48	99.06	98.84	99.1	98.64	101.36	101.3	100.04	100.0	99.08	100.06	102.28	102.2	
	σ	13.19	12.67	12.55	12.61	12.39	12.41	13.03	12.83	12.82	12.76	13.09	12.77	12.37	12.78	
$15\varsigma_{40}^{30}$	mean	97.92	97.72	95.2	95.56	95.08	95.46	100.6	100.4	98.08	97.86	97.3	97.1	99.16	100.24	
	σ	8.14	8.49	8.03	7.64	8.09	7.62	8.25	7.82	7.86	8.61	7.94	7.73	7.64	8.07	
$20\varsigma_{40}^{60}$	mean	207.74	206.22	205.0	204.68	205.78	206.0	210.14	211.72	207.06	207.22	207.42	207.38	211.08	210.44	
	σ	8.27	8.92	7.77	8.34	7.82	7.92	7.82	8.69	8.78	7.83	8.7	9.89	9.34	9.17	
$24\varsigma_{20}^{30}$	mean	24.7	24.7	24.22	24.38	24.42	24.34	25.64	25.34	24.84	25.02	24.82	24.84	24.46	24.46	
	σ	3.23	3.01	3.31	3.03	3.48	3.1	3.3	3.44	3.18	3.15	3.1	3.13	3.51	3.13	
$24\varsigma_{20}^{36}$	mean	45.42	45.44	44.38	44.84	44.82	44.92	46.18	46.1	45.24	45.26	45.28	45.2	45.84	45.64	
	σ	8.6	8.22	8.44	8.25	8.46	8.02	8.66	8.33	8.08	8.56	8.54	8.49	7.91	8.55	
$25\varsigma_{50}^{40}$	mean	144.78	146.26	144.02	143.36	143.22	144.72	151.46	151.62	146.38	146.02	146.6	145.48	146.34	146.98	
	σ	12.62	12.35	12.45	12.07	11.46	11.67	12.42	13.11	12.45	12.06	12.06	11.72	12.28	12.12	
$30\varsigma_{20}^{40}$	mean	41.62	41.24	40.64	40.74	40.6	41.04	42.36	42.32	41.46	41.08	41.4	41.52	41.8	41.58	
	σ	4.96	4.5	4.56	4.66	4.34	4.66	4.71	4.62	4.6	4.37	4.66	4.69	4.79	4.5	

Table 8 Computational results: MAHCP and meta-cooperative models $\Theta T(MAHCP, MATSP, MAHCP)$ and $\Theta T(MAHCP, MAHCP, MAHCP)$

		MAHCP	4Br(MAHCP,MAHCP,MAHCP)	5Br(MAHCP,MAHCP,MAHCP)	4Br(MAHCP,MATSP,MAHCP)	5Br(MAHCP,MATSP,MAHCP)	4Ra(MAHCP,MAHCP,MAHCP)	5Ra(MAHCP,MAHCP,MAHCP)	4Ra(MAHCP,MATSP,MAHCP)	5Ra(MAHCP,MATSP,MAHCP)	4Ri(MAHCP,MAHCP,MAHCP)	5Ri(MAHCP,MAHCP,MAHCP)	4Ri(MAHCP,MATSP,MAHCP)	5Ri(MAHCP,MATSP,MAHCP)
$4\zeta_{10}^{9}$	mean	8.1	8.0	8.04	7.94	8.04	8.02	8.04	7.92	8.0	8.06	8.04	7.9	7.98
	σ	0.75	0.82	0.85	0.73	0.82	0.84	0.87	0.74	0.82	0.76	0.85	0.81	0.79
$4\zeta_{10}^{10}$	mean	8.94	8.8	8.94	8.74	8.74	8.76	8.86	8.8	8.8	8.72	8.84	8.78	8.86
	σ	1.62	1.7	1.64	1.66	1.61	1.61	1.66	1.65	1.6	1.69	1.59	1.72	1.71
$6\zeta_{10}^{15}$	mean	13.89	13.74	13.74	13.76	13.72	13.72	13.76	13.72	13.82	13.72	13.7	13.82	13.76
	σ	1.99	2.04	2.05	2.09	2.05	2.13	2.04	2.13	1.97	2.11	2.08	2.01	2.11
$6\zeta_{15}^{12}$	mean	16.26	16.0	15.9	16.1	15.94	15.8	16.08	16.12	16.04	15.92	15.82	15.92	16.12
	σ	1.79	1.82	1.86	1.82	1.96	1.71	1.79	1.95	2.03	1.83	1.71	1.86	1.82
$6\zeta_{15}^{20}$	mean	23.18	22.74	22.94	22.78	22.78	22.64	22.56	22.76	22.92	22.72	22.86	22.98	22.84
	σ	1.96	1.96	1.99	1.92	1.91	1.92	1.89	1.87	1.85	1.99	1.83	1.92	2.04
$8\zeta_{20}^{15}$	mean	22.86	22.68	22.74	22.76	23.0	22.54	22.7	22.96	22.92	22.88	22.92	22.66	22.9
	σ	3.41	3.33	3.35	3.39	3.44	3.44	3.42	3.44	3.52	3.52	3.38	3.22	3.37
$8\zeta_{20}^{16}$	mean	27.24	26.54	26.78	26.86	27.12	26.86	26.48	27.1	26.74	26.84	26.7	26.96	26.78
	σ	2.22	2.05	2.08	2.14	2.02	2.06	2.27	2.15	2.13	2.12	2.17	2.07	1.96
$10\zeta_{20}^{20}$	mean	30.53	30.02	29.9	30.22	30.16	29.78	29.9	29.94	30.16	29.7	30.12	30.18	30.26
	σ	2.49	2.45	2.66	2.31	2.38	2.52	2.34	2.63	2.34	2.24	2.54	2.16	2.38
$10\zeta_{30}^{25}$	mean	64.32	64.36	63.7	64.52	64.58	64.02	63.94	65.18	64.92	64.24	64.4	64.2	64.3
	σ	2.4	1.94	2.34	2.06	1.89	2.03	1.7	2.22	1.98	1.8	1.84	2.23	1.93
$15\zeta_{30}^{40}$	mean	99.7	98.4	98.7	99.58	99.2	98.24	98.64	99.12	99.14	99.06	98.84	99.1	98.64
	σ	12.82	12.44	12.61	12.35	12.8	13.06	12.4	13.13	12.99	12.55	12.61	12.39	12.41
$15\zeta_{40}^{30}$	mean	95.86	95.56	95.1	96.28	95.1	94.86	94.54	96.24	95.26	95.2	95.56	95.08	95.46
	σ	7.52	7.38	7.71	7.23	6.95	7.41	7.93	7.76	7.93	8.03	7.64	8.09	7.62
$20\zeta_{40}^{60}$	mean	206.3	204.92	204.86	205.92	206.2	203.76	204.12	206.1	205.66	205.0	204.68	205.78	206.0
	σ	8.81	8.16	7.98	9.41	8.41	7.71	8.31	8.55	8.27	7.77	8.34	7.82	7.92
$24\zeta_{20}^{30}$	mean	24.78	24.2	24.46	24.44	24.34	24.42	24.36	24.56	24.66	24.22	24.38	24.42	24.34
	σ	3.29	3.27	3.34	3.18	3.05	3.29	3.27	3.01	3.5	3.31	3.03	3.48	3.1
$24\zeta_{20}^{36}$	mean	44.87	44.58	44.66	44.94	45.0	44.44	44.42	44.94	44.98	44.38	44.84	44.82	44.92
	σ	7.55	8.22	8.1	8.53	8.1	7.87	8.1	8.21	7.98	8.44	8.25	8.46	8.02
$25\zeta_{50}^{40}$	mean	144.18	143.48	143.16	144.02	143.38	143.26	143.08	144.42	143.92	144.02	143.36	143.22	144.72
	σ	11.94	12.3	11.84	12.23	12.31	12.79	11.75	12.42	12.76	12.45	12.07	11.46	11.67
$30\zeta_{20}^{40}$	mean	41.3	40.9	40.92	41.0	40.86	40.74	41.04	40.88	40.8	40.64	40.74	40.6	41.04
	σ	4.41	4.01	4.62	4.85	4.25	4.39	4.36	4.63	4.4	4.56	4.66	4.34	4.66