International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 3 (2011) pp.072-079

Intelligent Bio-Inspired System for Manufacturing Scheduling under Uncertainties

Ana Madureira[1] **and Ivo Pereira**[1]

[1]GECAD - Knowledge Engineering and Decision Support Research Group,
Institute of Engineering - Polytechnic of Porto (ISEP/IPP), Portugal
{*amd,iasp*}*@isep.ipp.pt*

Abstract: **A novel approach to scheduling resolution by combining Autonomic Computing (AC), Multi-Agent Systems (MAS), Case-based Reasoning (CBR), and Bio-Inspired Optimization Techniques (BIT) will be described. AC has emerged as a paradigm aiming at incorporating applications with a management structure similar to the central nervous system. The main intentions are to improve resource utilization and service quality. In this paper we envisage the use of MAS paradigm for supporting dynamic and distributed scheduling in Manufacturing Systems with AC properties, in order to reduce the complexity of managing manufacturing systems and human interference. The proposed CBR based Intelligent Scheduling System was evaluated under different dynamic manufacturing scenarios.**

Keywords: Multi-Agent Systems, Autonomic Computing, Case-based Reasoning, Bio-Inspired Optimization Techniques

I. Introduction

Self-organization can be defined as the process by which systems tend to reach a particular objective with minimal human interference. The mechanisms dictating its behavior are internal to the system. This field of research has received much attention in the Autonomic Computing paradigm [1].

AC intends to address complexity by using technology to manage technology. The term *autonomic* is derived from human biology. The autonomic nervous system monitors human heartbeat, checks blood sugar level and keeps body temperature without any conscious effort on humans part. Considering the analogy with human behaviours, self-managing autonomic capabilities anticipate Information Technology (IT) systems' requirements and solve problems with minimal human intervention. As a result, IT professionals can focus on jobs with higher value to the business.

However, there is an important distinction between autonomic activity in the human body and autonomic activities in IT systems. Many of the decisions made by autonomic capabilities in the body are involuntary. In contrast, self-managing autonomic capabilities in computer systems perform tasks that IT professionals choose to delegate on technology according to business or security policies.

Learning and adaptation represent key components in the design of modern effective and generally apply optimization, autonomic and intelligent approaches.

We propose to use a self-organized architecture for a Co-operative Scheduling System considering that it must be able to perform scheduling in highly dynamic environments where there is incomplete information and changes often occur; modify previously formed schedules considering recent dynamic information, minimizing the disruption of earlier schedules and still aiming for the most effective possible use of resources and achievement of goals and provide flexibility to robustly react to any perturbation in an efficient way.

In this work we intend to take advantage from the potentialities and combination of different efficient approaches from Hybrid Problem Solving perspective.

The remaining sections are organized as follows: Section II describes the background, with an overview about AC, BIT and CBR paradigms. A description of scheduling problem under consideration is made in Section III. Section IV presents a Bio-Inspired Scheduling System Prototype, with the details about system's implementation described in Section V. In Section VI it is presented a discussion about the obtained results about system's performance under dynamic perturbations. Finally, Section VII presents some conclusions and puts forward some future work.

II. Background

A. Autonomic Computing

Autonomic Computing is an IBM Grand Challenge proposed in 2001 by Paul Horn, Senior Vice-President of IBM Research. Horn argues that IT industry's focus on constant expansion will soon reach its breaking point: massive data centres are built in organic, ad hoc ways, resulting in a heterogeneous composition whose maintenance costs in terms of qualified staff, time and capital will soon exceed corporate capabilities [1].

AC proposes a broad new field of research related to the automation of IT management procedures, drawing inspiration from the human autonomous nervous system. AC concept revolves around four self-* properties, in which research efforts may be categorized: Self-Configuring, Self-Healing, Self-Optimizing and Self-Protecting. The names of these properties are self-explanatory, there is one inherent and implicit concept of significant importance relative to proactiveness. This is what separates this area of research from some of the functionalities which are already being integrated with

existing software systems.
Software systems managing IT resources without human supervision, called Autonomic Managers, are expected to continuously and autonomously respond and adapt to changes, and continuously seek ways to improve efficiency.
Many studies have already been made around this area. From Software Engineering concerns to address this new development paradigm [2, 3] all the way down to industry integration [4, 5].
Important techniques have been proposed from areas such as Service-Oriented Architectures [6, 7], Multi-Agent Systems [1], Grid Computing [4] and Control Theory [6, 8].
Planning is a critical component of the Autonomic Computing vision [1] where in the behavior of system elements are monitored and analyzed, and the performance is used to plan and execute suitable actions to take or keep the system in desirable states.

B. Bio-Inspired Optimization Techniques

Biological and natural processes have been sources of inspiration for computer science and information technology.
The interest of the BIT, also named Meta-heuristics, is that they converge, in general, to satisfactory solutions in an effective and efficient way (computing time and implementation effort). BIT have often been shown to be effective for difficult combinatorial optimization problems appearing in several industrial, economical, and scientific domains [9, 10].
Prominent examples are Genetic Algorithms, Simulated Annealing, Tabu Search, Scatter Search, Memetic Algorithms, Ant Colony Systems, and Particle Swarm Optimization.
When considering and understanding solutions followed by nature it is possible to use this acquired knowledge on the resolution of complex problems on different domains. From this knowledge application, on a creative way, arise new computing science areas - Bionic Computing.
The complexity of current computer systems has led the software engineering, distributed systems and management communities to look for inspiration in diverse fields, e.g. robotics, artificial intelligence or biology, to find new ways of designing and managing systems. Hybridization of different approaches seems to be a promising research field of computational intelligence focusing on the development of the next generation of intelligent systems.

C. Case-based Reasoning

CBR is an Artificial Intelligence methodology which aims to solve new problems by using information about the obtained solutions to previous similar problems [11] in learning based scenarios through experience.
In CBR, previous solved cases and its solutions are memorized as cases, saved in a repository, the casebase, so they can be reused in the future [12]. Instead of defining a set of rules or general lines, a CBR system solves a new problem by reusing similar cases previously solved.
CBR consists in a cycle (figure 1), usually described as the '4 Rs' [12]:

1. **Retrieve** the most similar case or cases;
2. **Reuse** the retrieved information and knowledge;

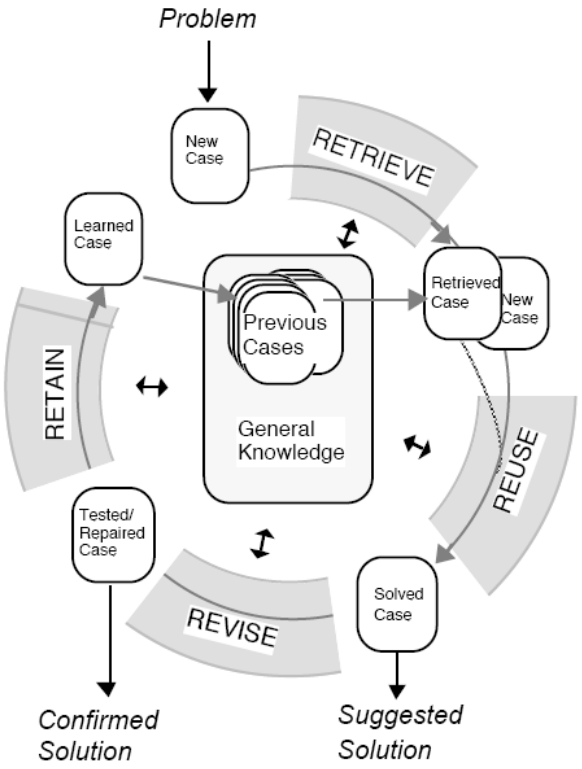

Figure. 1: CBR cycle (extracted from [12])

3. **Revise** the proposed solution;
4. **Retain** the revised solution to be useful for future problem solving.

A new case is a initial description of a problem, which is used to retrieve a case from the casebase. This retrieved case is then combined with the new case, in the Reuse phase, which gives a suggested solution to solve the new case. In the Revising phase, the suggested solution is tested and repaired if fails. In the Retaining phase, the useful experience is retained for future reuse, and the casebase is updated with the new learned case, or by modifying some existing cases.
The Retrieving phase starts with a partial description of the problem, and usually ends when it finds the most similar previous case. It is frequently defined a similarity measure to calculate the similarity between previous cases and the new case [13]. The correct definition of similarity measures for real world problems is one of the greatest challenges of research in CBR.
There are two ways of reusing the previous cases (Reusing phase): solution reuse and method reuse. In the first, the past solution is not directly copied to the new case, but there is some knowledge allowing the previous solution to be transformed into the solution of the new case. In the second, it is observed how the problem was solved, in the retrieved case, which has information about the used method for the resolution of the problem, including an explanation about the operators used, sub-objectives considered, alternatives, failures, etc. The retrieved method is reused to the new problem in the new context.
The objective of Revise phase is to evaluate the retrieved solution. If this solution is well succeeded it is possible to learn about the success, otherwise the solution is repaired using

some problem domain's expertise knowledge. The evaluating task executes the proposed solution and evaluates the result. This is usually a step outside the CBR, once it needs the problem execution in an application.
Retain phase consists in a process of integrating the useful information about the resolution of the new case in the case-base. It is important to know what case information is useful to retain, how to retain it, how to index the case for a future retrieve in similar cases, and how to integrate the new case in the memory structure.
Burke et al. [13] have referred that CBR is an appropriate approach for scheduling systems with expertise knowledge and empathize a research potential in dynamic scheduling. In general, CBR applications in scheduling domain can be classified in three categories [14]: algorithms reuse, operators reuse, and solutions reuse.
CBR based scheduling systems in the first category assume that it is probable that an effective approach for a specific problem's resolution will also be effective in the resolution of a similar problem. In these kind of systems, a case consists in a representation of the problem and in a known effective algorithm for its resolution. Schmidt [15] designed a CBR structure to choose the most appropriate method for scheduling problems' resolution in production machines. Schirmer [16] implemented a CBR system for selection of scheduling algorithms with the objective to solve project scheduling problems. Schirmer experimentally showed that some scheduling algorithms work better than others, in some instances of problems.
CBR scheduling systems in the second category reuse the operators for the resolution of the new problem [13]. A case describes a context in which a useful scheduling problem is used for repairing/adapting a scheduling plan to improve its quality, in terms of constraints' satisfaction [17]. Burke et al. [13] have proposed a case-based hyper-heuristic to solve timetabling problems. Beddoe et al. [17] have developed a CBR system to solve nurse scheduling problems.
Finally, in CBR scheduling systems of third category, it is used the whole or part of previous solutions of problems to construct the solution of the new case. A case contains the description of the problem and its solution, or part of the solution. This method was used for the resolution of manufacturing scheduling problems [18, 19] and university courses timetabling [13]. It was also used for constructing Meta-heuristics' initial solutions, as Genetic Algorithms [20] and Simulated Annealing [21].
There is also some research in hyper-heuristic area which can be defined as "heuristics that choose heuristics" or as "algorithms that choose the most adequate algorithm for a specific situation" [13]. Examples of CBR hyper-heuristics in scheduling problems are [13], [22] and [23].

III. Scheduling Problem Definition

Most real-world multi-operation scheduling problems can be described as dynamic and extended versions of the classic Job-Shop scheduling combinatorial optimization problem. In practice, scheduling environment tends to be dynamic, i.e. new jobs arrive at unpredictable intervals, machines breakdown, jobs can be cancelled and due dates and processing times can change frequently.

The problem, focused in this work, which we call Extended Job-Shop Scheduling Problem (EJSSP), has major extensions and differences in relation to the classic Job-Shop Scheduling Problem. In this work, we define a job as a manufacturing order for a final item, which can be Simple or Complex [24]. The main elements of the EJSSP problem [24, 25] could be modeled as: a set of multi-operation jobs $J_1, ..., J_n$ has to be scheduled on a set of machines $M_1, ..., M_n$. d_j is the due date of job J_j. t_j is the initial processing time of job J_j. r_j is the release time of job J_j. The existence of operations on the same job, on different parts and components, processed simultaneously on different machines, followed by components assembly operations (multi-level jobs).
Furthermore EJSSP should meet the following restricted conditions:

1. The existence of different job release dates r_j and due dates d_j.
2. The possibility of job priorities definition, reflecting the importance of satisfying their due dates, being similar to the weight assigned to jobs in scheduling theory.
3. Precedence constraints among operations of the different jobs.
4. New jobs can arrive at unpredictable intervals. Jobs can be cancelled. Changes in task attributes can occur: processing times, due dates, release dates and priorities.
5. Each operation O_{ijkl} must be processed on one machine of the set M_i, where p_{ijkl} is the processing time of operation O_{ijkl} on machine M_i.
6. The existence of operations on the same job, on different parts and components, processed simultaneously on different machines, followed by components assembly operations (multi-level jobs).
7. A machine can process more than one operation of the same job (recirculation).
8. The existence of alternative machines, identical or not.

IV. Bio-Inspired Scheduling System Prototype

Bio-Inspired Scheduling System Prototype (BioSched) is an Autonomic Scheduling System in which communities of agents model a real manufacturing system subject to perturbations. Agents must be able to learn and manage their internal behavior and their relationships with other autonomic agents, by cooperative negotiation in accordance with business policies defined by user manager.
BioSched system is a MAS where each agent represents a resource (Resource Agents) in a Manufacturing System. Each Resource Agent must be able: to find an optimal or near optimal local solution through BIT algorithms; to deal with system dynamism (new jobs arriving, cancelled jobs, changing jobs attributes, etc); to change/adapt the parameters of the basic algorithm according to the current situation; to switch from one Meta-Heuristic algorithm to another and to cooperate with other agents.

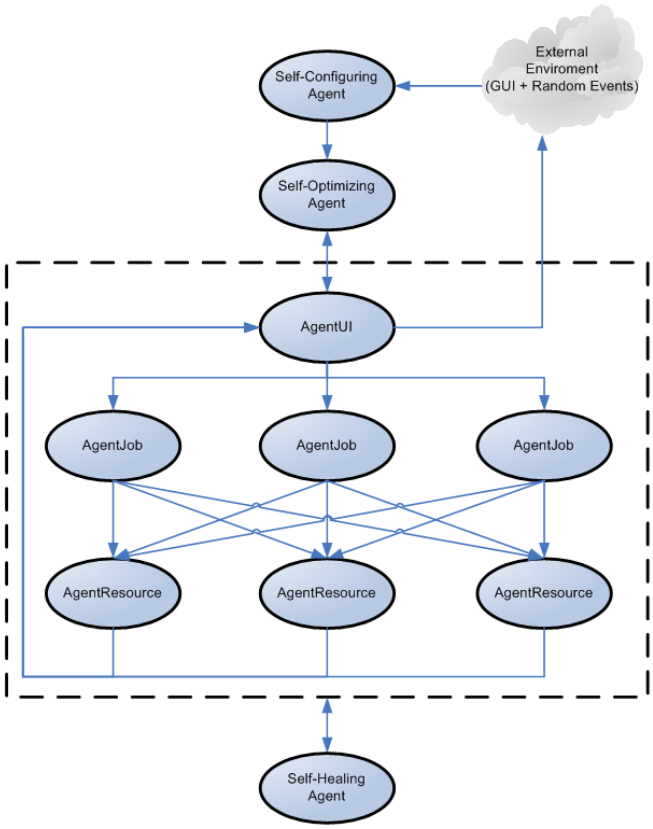

Figure. 2: BioSched system's architecture

Scheduling approach followed in this proposal is rather different from the ones found in the literature; as we try to implement a system where each Resource Agent is responsible for optimizing the scheduling of operations for one machine through a BIT. This considers a specific kind of social interaction that is cooperative problem solving, where the group of agents work together to achieve a good solution for the problem.

The original Scheduling problem defined in Section III is decomposed into a series of Single Machine Scheduling Problems (SMSP). The Resource Agents, which have a BIT associated, obtain local solutions and later cooperate in order to overcome inter-agent constraints and achieve a global schedule [25].

The system waits for the solutions obtained by the Resource Agents and then applies a repair mechanism to shift some operations in the generated schedules till a feasible solution is obtained. This cooperation mechanism must be prepared to accept agents subjected to dynamism (new jobs arriving, cancelled jobs, changing jobs attributes).

BioSched system's architecture is based on six different types of agents (figure 2).

In order to allow a seamless communication with the user, an **User Interface Agent (AgentUI)** is implemented. This agent, apart from being responsible for the user interface, will dynamically generate the necessary Job Agents according to the number of jobs that comprise the scheduling problem and assign each job to the respective Job Agent.

The **Job Agents** will process the necessary information about the respective job. This agent will be responsible for the generation of the earliest and latest processing times, the verification of feasible schedules and identification of constraint conflicts on each job and the decision on which Resource Agent is responsible for solving a specific conflict.

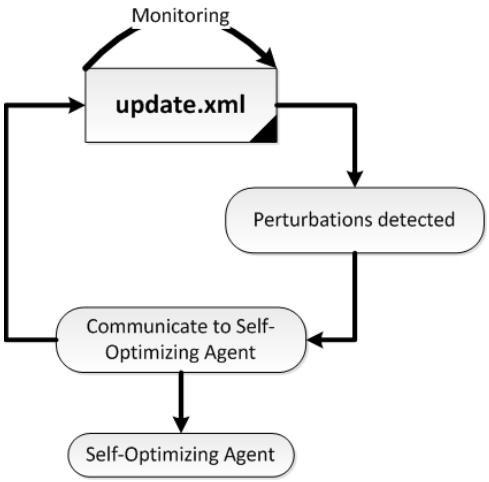

Figure. 3: Self-Configuring module

The **Resource Agents** are responsible for the scheduling of the operations that require processing in the machine supervised by the agent. This agent will implement meta-heuristic and local search procedures in order to find best possible operation schedules and will communicate those solutions to the Job Agent for later feasibility check.

Additionally, we consider **Self-* agents**, namely Self-Configuring, Self-Optimizing and Self-Healing, which will be described in the following subsections. Self-protecting was not considered in this work.

A. Self-Configuring module

The function of Self-Configuring is a control loop that collects details from the system and acts accordingly (figure 3). Self-Configuring enable systems to adapt to changing conditions by adapting their own configurations, allowing the addition and removal of resources without service disruption.

Respectively to the Self-*Agents, the Self-Configuring Agent is responsible for monitoring the system in order to detect changes occurred in the schedule, allowing the system to perform dynamic adaptation. With this agent, the system will be prepared to automatically handle dynamism by adapting the solutions to external perturbations.

Dynamic Adaptation is necessary due to two classes of external events: **Partial events** imply variability in jobs/operations attributes such as processing times, due dates or release times; and **Global events** imply variability in neighborhood/population structure, resulting from new job arrivals, job cancellations, machines breakdown, etc. While, on one hand, partial events only require redefining job attributes and re-evaluation of the objective function, global events require a change on solution structure and size, carried out by inserting or deleting operations, and also re-evaluation of the objective function. Therefore, under a total event, the modification of the current solution is imperative, through job arrival integration mechanisms (when a new job arrives to be processed), job elimination mechanisms (when a job is cancelled) and regeneration mechanisms in order to ensure a

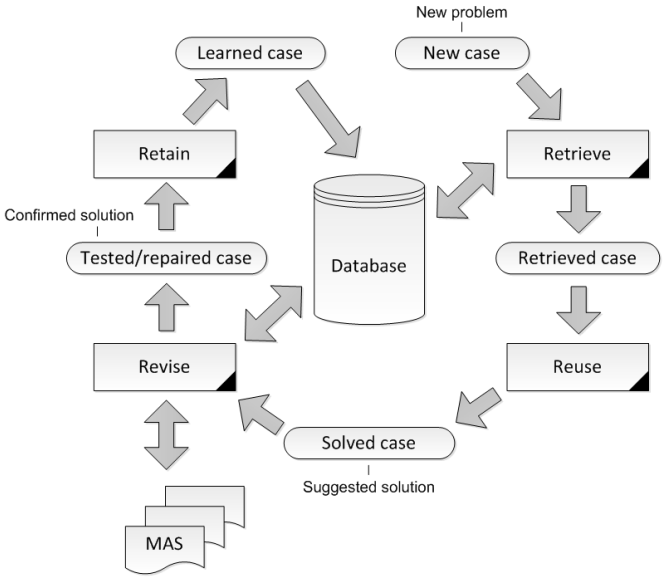

Figure. 4: Self-Optimizing module

dynamic adaptation of population/neighborhood.

B. Self-Optimizing module

Self-Optimizing module (figure 4) gives the system the ability of monitoring its state and performance and proactively tune itself to respond to environmental stimuli. Each Resource Agent adopts and provides self-parameterization of the solving method in accordance with the problem being solved (parameters can change in run-time). Each Resource Agent must be able to define which BIT will be used and define initial parameters of BIT according to the current situation or even to commute from one algorithm to the other according to current state and previous learning. Resource Agents self-optimize through learning and past experience.
The Self-Optimizing Agent is responsible for automatically tuning the BIT parameters, according to the problem's characteristics and dimension. This agent receives the initial problem, or the changes detected by Self-Configuring Agent, and automatically select the BIT to use, and makes its self-parameterization. If some perturbation occurs in the system, parameters may change in run-time. This tuning of parameters is made through learning and experience, since it uses a CBR module. Each time a new problem (case) appears, CBR uses past experience in order to specify the meta-heuristic and respective parameters for that case. When the new case is solved, it is stored for later use.

C. Self-Healing module

Finally, the Self-Healing Agent gives the capacity to the system for diagnosing deviations from normal conditions and proactively takes actions to normalize them and avoid service disruptions. This agent monitors other agents in order to provide overall self-healing capabilities. Since agents may crash for some reason, self-healing provides one or more agents backup registries in order to grant storage for the reactivation of lost or stuck scheduling agents with meaningful results, thus enabling the system to restart from a previous checkpoint as opposed to a complete reset. With this agent, the system becomes stable, even if some deadlocks or crashes occur.

V. Implementation and Computational Study

To evaluate the proposed architecture we have developed a prototype, named BioSched system, whose main objective is supporting dynamic and distributed scheduling for Manufacturing Systems with autonomic properties.
The system is developed in JADE (Java Agent DEvelopment Framework) and Java. JADE is structured in order to simplify and minimize the development of multi-agent systems while ensuring standard compliance through a comprehensive set of system services and agents in compliance with the FIPA specifications. More information about JADE can be found in http://jade.tilab.com/.
Initially BioSched's behaviour will be analysed and illustrated under different dynamic scenarios on the resolution of an instance of Lawrence [26] problems. Then the computational study will be extended in order to evaluate the global BioSched's performance.

A. Ant Colony Optimization

The ACO algorithm (Table 1) is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. This algorithm is a member of ant colony algorithms family, in swarm intelligence methods, and it constitutes some Meta-Heuristic optimizations. Initially proposed by Marco Dorigo [27] in his PhD thesis. The first ACO algorithm is known as Ant System was aiming to search for an optimal path in a graph; based on the foraging behaviour of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on several aspects of the behaviour of ants [28]. After initialization, the Meta-Heuristic iterates over three phases: at each iteration, a number of solutions are constructed by the ants; these solutions could be then improved through a local search (this step is optional), and finally the pheromone is updated through evaporation and by increasing the pheromone levels associated with a chosen set of good solutions.
The good results of ACO algorithms has made them appealing for applications in industrial settings [28].

B. Parameterization

In order to evaluate the performance of the developed system, we have considered for experimental case study the Lawrence's La03 benchmark problem [26] which will be subject to dynamic perturbations. The expected partial and global events to test dynamism are shown in Table 2.

Table 1: Ant Colony Optmization Algorithm

Initialization: Set ACO parameters. Initialize pheromone trails
While termination criteria not met do
    Construct Ant Solutions
    Apply Localsearch (optional)
    Update Pheromones
EndWhile

Table 2: Global and Partial Events

Event	Time instant	Event type	Job	Value
Partial	75	Due date update	T1	630
	175	Priority update	T3	3
	200	Due date update	T4	600
	300	Priority update	T8	5
	375	Due date update	T10	650
Global	50	Job cancelation	T2	
	100	Arrival of new job	T11	= T1
	125	Job cancelation	T9	
	350	Arrival of new job	T12	= T5
	450	Arrival of new job	T13	= T10

In order to correctly evaluate the system behavior, we used the system with and without the CBR module. The CBR module is responsible to select a Meta-heuristic and self-parameterize it, with minimal human interference. To get results without CBR we have used ACO, with the following parameters: stopping criteria of 100, evaporation rate of 80%, 25 ants, alpha and beta equal to 1.
The optimization measure used in this computational study was the minimization of Weighted Tardiness (WT).

VI. Results Discussion

The obtained results are related from BioSched simulation with and without CBR module, under four dynamic scenarios: without dynamism, in the presence of partial events, in the presence of global events, and in the presence of both partial and global events (mixed events).
In figure 5 it is possible to analyze the obtained results by WT minimization. BioSched system with the CBR module encountered advantages in the presence of partial and mixed events. In the presence of partial events, it was obtained the 3040 value with CBR, versus the 3514 value without CBR. With mixed events, it was obtained the 6728 value with CBR opposed to the 8606 value without CBR.
It is important to clarify the results in the presence of global events. The significant deviation obtained is because, sometimes, jobs' cancelation is not already possible, since those jobs are already being processed, in the time instant of the global event occurrence. It was what happened in the presence of global events, with CBR. The two jobs where not cancelled, resulting in a deterioration of results (figure 5).
Another evaluation criterion under consideration was the system's utilization rate. These results are presented in figure 6. This is a maximization criteria, so higher the value, better it is. Analysing this results we can reach almost to the same conclusions, concerning the improvement of system's performance by better use of resources.
From these obtained results we can understand that the system's performance with CBR improved while performing real world events (partial and global events), which is the real objective of the system. This appoints to the advantage of given learning capabilities.
We also analyse the computational execution time on the different scenarios. These results are presented in figure 7. At first sight, it is possible to notice the significant difference between the system without CBR and with CBR in the presence of global and mixed events. This can be explained because the CBR module used a more adequate parameterization.

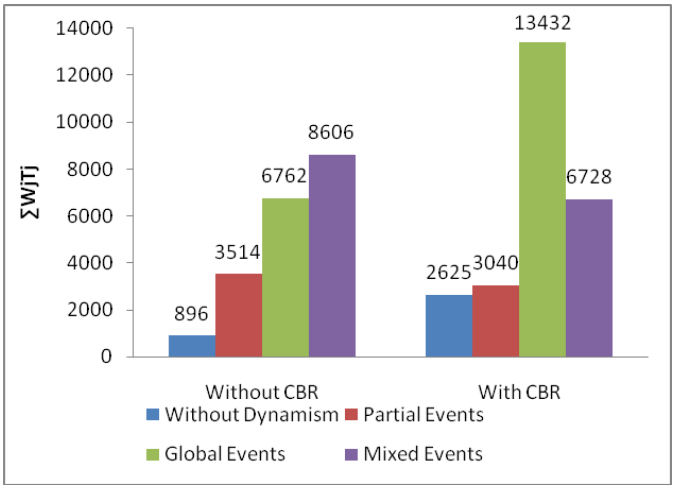

Figure. 5: WT results

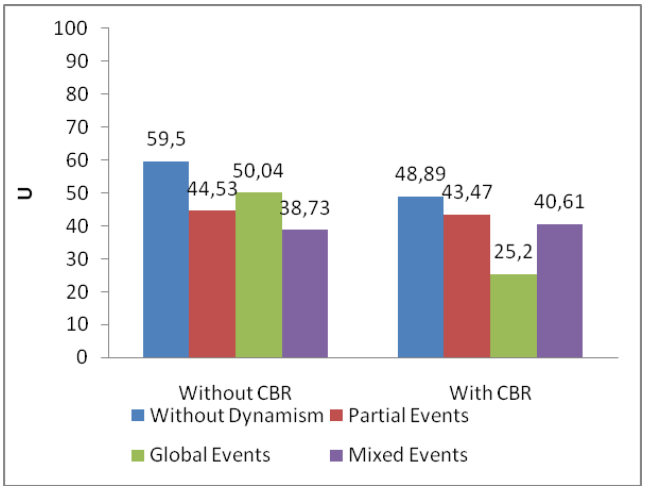

Figure. 6: Results of system's utilization rate

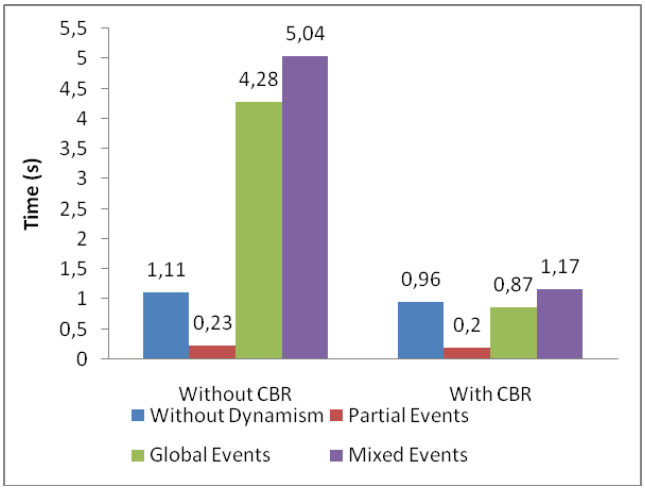

Figure. 7: Results of execution time (in seconds)

Considering all the obtained results (WT minimization, system's utilization rate and execution time) it is important to highlight the advantages of using CBR in the presence of mixed events. It achieved better results in an efficient manner, what can be very important in real world scenarios, where changes can often occur and fast responses are necessary.
The computational study was extended for 10 instances of academic problems selected from OR-Library [29] and it was possible to reach almost the same conclusions.

VII. Conclusion

This paper presented a vision on the use of self-organization methods as a promising concept to automate scheduling optimization. Manufacturing scheduling is traditionally elaborated in a centralized manner and does not consider dynamic adaptation to external perturbations. This paper envisage the proposal of a Self-organized architecture for a Cooperative Scheduling System considering that it must be able to perform scheduling in highly dynamic environments where changes often occur; modify previously formed schedules considering recent dynamic information, minimizing the disruption of earlier schedules and still aiming for the most effective possible use of resources and achievement of goals and provide flexibility to react robustly to any disruption in an efficient and timely manner.
The obtained results showed that the proposed Autonomic Scheduling System has potential to improve the system performance mainly by combining robustness, efficiency, effectiveness and autonomy.
Work still to be done includes the exhaustive testing of the proposed scheduling system and proposed mechanisms under dynamic environments subject to several random perturbations.

Acknowledgments

The authors would like to acknowledge FCT, FEDER, POCTI, POSI, POCI, POSC, and COMPETE for their support to R&D Projects and GECAD.

References

[1] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[2] D. Bustard and R. Sterritt, "A requirements engineering perspective on autonomic systems development," *Autonomic Computing - Concepts, Infrastructure, and Applications*, pp. 19–33, 2006.

[3] R. Cervenka, D. Greenwood, and I. Trencansky, "The aml approach to modeling autonomic systems," in *ICAS '06: Proceedings of the International Conference on Autonomic and Autonomous Systems*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 29–34.

[4] A. Ganek, "Overview of autonomic computing: Origins, evolution, direction," *Autonomic Computing - Concepts, Infrastructure, and Applications*, pp. 3–18, 2006.

[5] IBM, "An architectural blueprint for autonomic computing," *White paper by IBM*, 2006.

[6] V. Bhat, M. Parashar, and N. Kandasamy, "Autonomic data streaming for high-performance scientific applications," *Autonomic Computing - Concepts, Infrastructure, and Applications*, pp. 413–433, 2006.

[7] D. Chess, J. Hanson, J. Kephart, I. Whalley, and S. White, "Dynamic collaboration in autonomic computing," *Autonomic Computing - Concepts, Infrastructure, and Applications*, pp. 253–274, 2006.

[8] S. Abdelwahed and N. Kandasamy, "A control-based approach to autonomic performance management in computing systems," *Autonomic Computing - Concepts, Infrastructure, and Applications*, pp. 149–167, 2006.

[9] P. Siarry and Z. Michalewicz, *Advances in Metaheuristics for Hard Optimization*, ser. Natural Computing Series. Springer, 2008.

[10] F. Xhafa and A. Abraham, *Metaheuristics for Scheduling in Industrial and Manufacturing Applications*. Springer, 2008, vol. 128.

[11] J. Kolodner, *Case-Based Reasoning*. Morgan Kaufmann Publishers Inc, 1993.

[12] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *Artificial Intelligence Communications*, vol. 7, pp. 39–52, 1994.

[13] E. Burke, B. MacCarthy, S. Petrovic, and R. Qu, "Knowledge discovery in a hyper-heuristic for course timetabling using case-based reasoning," in *PATAT 2002*, 2002, pp. 276–287.

[14] S. Petrovic, Y. Yang, and M. Dror, "Case-based selection of initialisation heuristics for metaheuristic examination timetabling," *Expert Syst. Appl. 33*, pp. 772–785, 2007.

[15] G. Schmidt, "Case-based reasoning for production scheduling," *International Journal of Production Economics*, vol. 56–57, pp. 537–546, 1998.

[16] A. Schirmer, "Case-based reasoning and improved adaptive search for project scheduling," *Naval Research Logistics 47*, pp. 201–222, 2000.

[17] G. Beddoe, S. Petrovic, and J. Li, "A hybrid metaheuristic case-based reasoning system for nurse rostering," *Journal of Schedling 12*, pp. 99–119, 2009.

[18] J. M. A. Coello and R. C. Santos, "Integrating cbr and heuristic search for learning and reusing solutions in real-time task scheduling," in *Proceedings of 3rd International Conference on Case-Based Reasoning*. Germany: Springer-Verlag, 1999, pp. 89–103.

[19] B. L. MacCarthy and P. Jou, "Case-based reasoning in scheduling," in *Proceedings of the Symposium on Advanced Manufacturing Processes, Systems and Techniques (AMPST96)*. MEP Publications Ltd, 1996, pp. 211–218.

[20] S. Oman and P. Cunningham, "Using case retrieval to seed genetic algorithms," *International Journal of Computational Intelligence and Applications*, vol. 1, pp. 71–82, 2001.

[21] P. Cunningham and B. Smyth, "Case-based reasoning in scheduling: Reusing solution components," *The International Journal of Production Research*, vol. 35, pp. 2947–2961, 1997.

[22] H. L. Fang, P. Ross, and D. Corne, "A promising hybrid ga/heuristic approach for open-shop scheduling problems," in *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI'94)*. John Wiley and Sons, Ltd, 1994, pp. 201–222.

[23] S. Grolimund and J.-G. Ganascia, "Driving tabu search with case-based reasoning," *European Journal of Operational Research*, pp. 326–338, 1997.

[24] A. Madureira, "Meta-heuristics application to scheduling in dynamic environments of discrete manufacturing," Ph.D. dissertation, University of Minho, Braga, Portugal, 2003, (in portuguese).

[25] A. Madureira, J. Santos, N. Gomes, and C. Ramos, "Proposal of a cooperation mechanism for teamwork based multi-agent system in dynamic scheduling through meta-heuristics," in *Proceedings 2007 IEEE International Symposium on Assembly and Manufacturing (ISAM07)*, Ann Arbor, Michigan, USA, 2007, pp. 233–238.

[26] S. Lawrence, "Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques," Ph.D. dissertation, Pittsburgh, Pennsylvania: Graduate School of Industrial Administration, Carnegie-Mellon University, 1984.

[27] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Politecnico di Milano, Italy (in Italian), 1992.

[28] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization - artificial ants as a computational intelligence technique," *IEEE Computational Intelligence Magazine*, pp. 28–39, 2006.

[29] OR-Library. [Online]. Available: http://people.brunel.ac.uk/ mastjjb/jeb/info.html

Author Biographies

Ana Madureira was born in Lourenço Marques, Mozambique, in 1969. She got his BSc degree in Computer Engineering with a specialization in manufacturing, in 1993, from Institute of Engineering - Polytechnic of Porto (ISEP/IPP), the MSc degree in Electrical and Computer Engineering, in 1996, from the University of Porto, and the PhD degree in Production and Systems, in 2003, from University of Minho, Portugal. Currently she is Coordinator Professor at the Institute of Engineering - Polytechnic of Porto (ISEP/IPP). She is senior member of IEEE and PhD researcher of the GECAD - Knowledge Engineering and Decision Support Research Group where she is supervisor of 3 R&D projects, and supervised 6 MSc thesis and 3 PhD thesis works. In the last few years, she was author and co-author of over sixty scientific papers in conference proceedings, journals and books. His main teaching and scientific work areas of interest are Industrial Management, Planning and Scheduling, Dynamic and Reactive Scheduling, Meta-Heuristics, Genetic Algorithms, Evolutionary Computation, Decision Support Systems and Intelligent systems.

Ivo Pereira was born in Foz do Douro, Porto, in 1984. He got his BSc degre in Computer Engineering in 2007, from Institue of Engineering - Polytechnic of Porto (ISEP/IPP), the MSc degree in Computer Enginerring, with a specialization in Knowledge and Decision Technologies, in 2009, also from Institue of Engineering - Polytechnic of Porto, and currently he is PhD student of Electrical and Computer Engineering, in University of Trás-os-Montes e Alto Douro (UTAD). He is researcher of the GECAD - Knowledge Engineering and Decision Support Research Group, where participated in two R&D projects. In the last few years, he was author and co-author of over twenty scientific papers in conference proceedings, journals and books. His main scientific work areas of interest are Dynamic and Reactive Scheduling, Meta-Heuristics, Evolutionary Computation, Multi-Agent Systems, Learning, Decision Support Systems and Intelligent systems.