Case-based Reasoning for Meta-heuristics Self-Parameterization in a Multi-Agent Scheduling System

Ivo Pereira
GECAD - Knowledge Engineering and Decision Support Group
Institute of Engineering - IPP
Porto, Portugal
iasp@isep.ipp.pt

Ana Madureira
GECAD - Knowledge Engineering and Decision Support Group
Institute of Engineering - IPP
Porto, Portugal
amd@isep.ipp.pt

Paulo de Moura Oliveira
Department of Engineering
University of Trás-os-Montes e Alto Douro
Vila Real, Portugal
oliveira@utad.pt

***Abstract*— A novel agent-based approach to Meta-Heuristics self-configuration is proposed in this work. Meta-heuristics are examples of algorithms where parameters need to be set up as efficient as possible in order to unsure its performance. This paper presents a learning module for self-parameterization of Meta-heuristics (MHs) in a Multi-Agent System (MAS) for resolution of scheduling problems. The learning is based on Case-based Reasoning (CBR) and two different integration approaches are proposed. A computational study is made for comparing the two CBR integration perspectives. In the end, some conclusions are reached and future work outlined.**

Keywords- Case-based Reasoning, Learning, Meta-heuristics, Multi-Agent Systems, Scheduling

I. INTRODUCTION

Scheduling problem can be defined as "the definition of start and completion times and the allocation of resources to each task of a given set, according to various constraints of resources and/or tasks" [2], and several approaches have been developed to its resolution. However, many of those approaches are impractical in real manufacturing environments, which are inherently dynamic where complex constraints exist and a variety of unexpected disruptions occur. In most real-world environments, scheduling is a progressive reactive process where the presence of real-time information requires continuous reconsideration and review of the pre-established plans. Research on scheduling has not fully considered this dynamic issue, mostly focusing on the optimization of static scheduling plans [3].

In spite of all research made so far, the scheduling problem is still known to be NP-complete, even in static environments. This fact presents severe challenges to conventional algorithms and stimulate researchers to explore new directions. Multi-Agent technology has been considered an important approach for developing industrial distributed systems [4][5][6] motivating its use in this work.

Learning is a crucial component of autonomy and pro-activeness which must be a study target of agents and MAS [7]. Panait and Luke [8] described two different approaches of learning in MAS: team learning and concurrent learning. In the first, there is only one apprentice involved, with the objective to learn about the environment to improve a set of agents. In concurrent learning, there are multiple apprentices trying to improve parts from the team.

Since MH parameterization revealed to be a hard task, which requires expertise knowledge about the domain and what techniques and parameters should be used, it is important to make systems capable of autonomously parameterize themselves. To validate this self-parameterization, we use learning about past experience, with Case-based Reasoning (CBR) revealing to be a promising approach. Using CBR, systems can remember previous solve cases and automatically decide which MH and parameters to use for the resolution of new similar problems.

The paper is organized as follows: section II describes the dynamic scheduling problem used in this work. CBR is explained along section III and some applications to scheduling problem resolution are presented. In section IV, the implemented prototype is described detailing the MAS and the different perspectives of CBR integration. The computational study is presented in section V and, finally, in section VI some conclusions and some future work is presented.

II. DYNAMIC SCHEDULING PROBLEM

The scheduling problem treated in this work is named Extended Job-Shop Scheduling Problem (EJSSP) [9] and has some major extensions and differences compared to the classic Job-Shop Scheduling Problem (JSSP).

JSSP have a set of tasks processing in a set of machines, with each task having an ordered list of operations, each one characterized by the respective processing time and machine where is processed [10].

The main elements of EJSSP problem are:

- a set of multi-operation jobs $J_1,\ldots,J_n$ to be scheduled on a set of machines $M_1,\ldots,M_{n;}$
- d_j represents the due date of job J_j;
- t_j represents the initial processing time of job J_j;
- r_j represents the release time of job J_j;

EJSSP problems consist in JSSP problems with additional restrictions, with the objective to better represent reality [9]. Some of those restrictions are:

- Different release and due dates for each task;
- Different priorities for each task;
- Possibility that not every machines are used for all tasks;
- A task can have more than one operation being processed in the same machine;
- Two or more operations of the same task can be processed simultaneously;
- Possibility of existence of alternatives machines, identical or not, for operations' processing.

Scheduling Problem is included in NP-hard combinatorial optimization problems. The methods for their resolution can be divided in exact and approximation algorithms [9]. In the first, it is made an exhaustive solutions space search and it is ensured the optimal solution. In the other hand, approximation algorithms [11], including heuristics and MH, do not guarantee the optimal solution but have the objective to find a good solution in an acceptable period of time. For this reason, they are used in this work together with MAS.

III. CASE-BASED REASONING

Case-Based Reasoning (CBR) is an Artificial Intelligence methodology aiming to solve new problems by using information about previous similar problems resolution [12]. CBR operates under the premise that similar problems can require similar solutions [13].

CBR roots are found in the work of Roger Schank about dynamic memory and how the memory of previous situations can affect problems' resolution and learning processes [14]. There are also references about the study of analogical reasoning [15].

CYRUS system, developed by Janet Kolodner [12], is the first known CBR system. It was based in Schank's dynamic memory model [14] and it was a question-answer system with knowledge about the different travels and meetings of USA ex-Secretary of State, Cyrus Vance. PROTOS, developed by Bruce Porter and his group [16], was another of the first systems to use CBR, dealing with classification problem.

A. CBR Cycle

In CBR, previous solved cases and their solutions are memorized as cases, stored in a repository (casebase), in order to be reused in the future [13]. Instead of defining a set of rules or general lines, a CBR system solves a new problem by reusing similar cases that were previously solved [17].

The CBR cycle (known as the '4 REs' cycle) is described in Figure 1 and consists in four main phases [13][18]:

1) Retrieve the most similar case or cases;
2) Reuse the retrieved information and knowledge;
3) Revise the proposed solution;
4) Retain the revised solution to be useful for future problem solving.

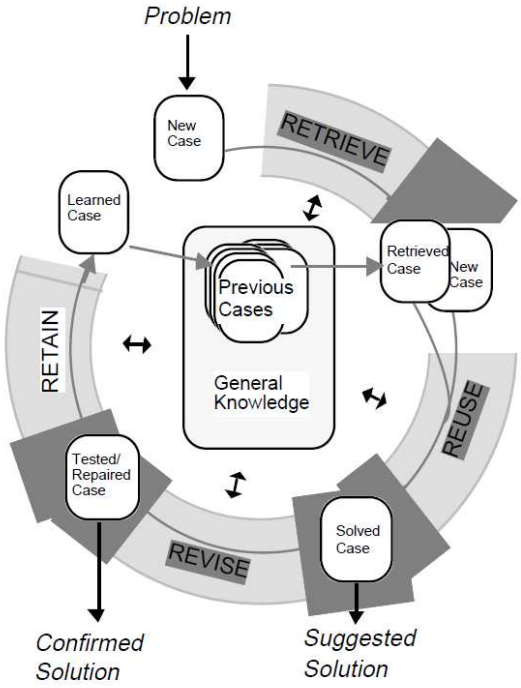

Figure 1. The CBR cycle [18]

The initial description of a problem (new case), is used to retrieve a case from the casebase. In the Reusing phase, the retrieved case is analyzed in order suggest a solution for new case's resolution. In the Revising phase, this suggested solution is tested, for example, by executing it in the system, and repaired if it fails. In the Retaining phase, the useful experience is retained for future use, and the casebase is updated with the new learned case (or by modifying some existing cases).

B. Applications to Scheduling problem resolution

Burke et al. [19] have referred that CBR is an suitable approach for scheduling systems endowed with expertise knowledge, and emphasize a research potential in dynamic scheduling.

In general, CBR applications to scheduling resolution can be classified in three categories [17]: algorithms reuse, operators reuse, and solutions reuse.

In CBR scheduling systems reusing algorithms it is assumed that an effective approach for a specific problem's resolution will also be effective in the resolution of a similar problem. In these systems, a case consists in a representation of the problem and in a known effective algorithm for its resolution. Some applications in algorithms reuse can be found in [20], [21] and [22].

CBR scheduling systems in the second category reuse the operators for the resolution of the new problem [19]. 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 [13]. Some applications to scheduling problem resolution are [13] and [19].

In CBR scheduling systems reusing solutions it is used the whole or part of previous problems' solutions to construct the solution of the new problem. A case contains the description of a problem and its solution, or part of solution. This method was used for the resolution of manufacturing scheduling problems [23][24][25] and university courses timetabling [19][26]. It was also used for constructing MH' initial solutions, as Genetic Algorithms [27][29] and Simulated Annealing [28].

IV. IMPLEMENTED PROTOTYPE

Meta-heuristics are very useful obtaining good solutions in reasonable execution times, and sometimes they even obtain optimal solutions. But, to be possible to achieve optimal or near-optimal solutions, it is required the most appropriate tuning of MH parameters, which has revealed to be a hard task, since it requires some expertise knowledge from the MH to use and from the problem to solve [30].

Given this, the proposed system must adopt and provide self-parameterization of MHs respectively to the problem to be solved through CBR, with the possibility that parameters can change in run-time. According to the current situation being treated, the system must be able to define which MH must be used and also define the respective parameters. It is even possible to change from one MH to another, according to current state and previous information, through learning and experience.

In the proposed MAS, there are agents representing jobs/tasks and agents representing resources/machines. The system is able to find optimal or near optimal solutions through the use of MHs, deal with dynamism (arriving of new jobs, cancelled jobs, changing jobs attributes, etc.), change/adapt the parameters of the algorithm according to the current situation, switch from one MH to another, and perform a coordination between agents through cooperation or negotiation mechanisms.

Since it is impossible to predict each problem to treat, the system should be capable of learning about its experience during lifetime, as humans do. To perform this learning mechanism, it is proposed the use of CBR.

The next three subsections describe the implemented MAS and also the two different perspectives of CBR integration on the MAS.

A. *Multi-Agent Scheduling System*

The developed MAS for Scheduling problem resolution is a hybrid autonomous architecture represented by hierarchical and team models (if using cooperation) or market models (if using negotiation), as defined by Horling and Lesser [31]. It is composed by three kinds of agents as shown in Figure 2.

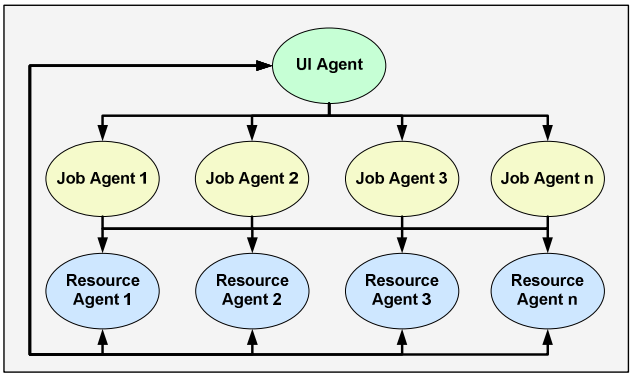

Figure 2. Multi-Agent Scheduling System

In order to allow a consistent communication with the user, a User Interface Agent (UI Agent) was implemented. This agent is responsible for the user interface definition and also for the dynamic generation of the necessary Job and Resource agents, according to the number of tasks/jobs and machines comprising the scheduling problem. It is also responsible for assigning each task to the respective Job Agent and, in the end, applying the repair mechanism and starting the coordination mechanisms.

Job agents process the necessary information about the respective job. They are responsible for the generation of the earliest and latest processing times on the respective job and automatically separate each job's operation for the respective Resource Agent.

Resource agents are responsible for scheduling the operations that require processing in the machine supervised by the agent. These agents implement MH in order to find the best possible single-machine schedules/plans of operations and communicate those solutions to the AgentUI for later feasibility check.

B. *Global CBR perspective*

In this system, CBR can be used in two different approaches. The first one consists in retrieving the most similar case to the new problem, regardless the MH to use. With this, the case containing the MH and respective parameters to use is retrieved. The second approach consists in first choosing the MH and then retrieving the most similar case, with the parameters to use for that MH.

Both approaches represent a Solution Reuse, but the first one is more appropriate to the Scheduling problem resolution, since it is important that the system have the capacity to decide which MH to use and defining the respective parameters, because not all MH are effective for the resolution of every type of problem.

In a global perspective of CBR integration on the system (Figure 3), there is an external module (agent) responsible

for executing the CBR. This module communicates with UI Agent in order to suggest a MH to use and configure the respective parameters. This communication occurs after loading the problem to solve and before communicating with Job and Resource agents.

In this perspective, it is possible to assume that the system performs a team learning approach, since the CBR module performs self-parameterization for the whole system.

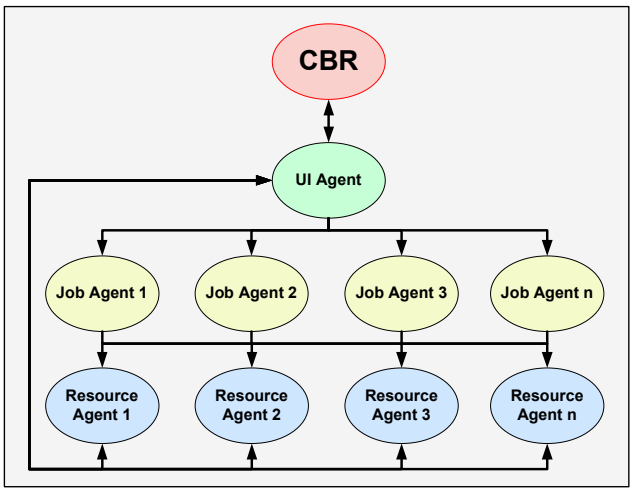

Figure 3. CBR global integration

Like previously described in Figure 1, every new problem, or perturbations occurred, leads to a new case in the system, with the previous most similar cases being retrieved from the casebase. After that, the better case is reused, becoming a suggested solution. After the solution revision, the case is executed in the MAS. This revision is made to be possible to escape from local optimal solutions and stagnation, since it is used some disturbance in the parameters of the proposed solution. After the conclusion of the MAS execution, the case is confirmed as a good solution, being retained on the database as a new learned case, for future use.

C. *Resource agents' CBR perpective*

Another possible approach of CBR integration is in a resource agents' perspective, where a concurrent learning is made, since each resource agent has his own CBR module for performing the self-parameterization of MHs. This perspective is shown in Figure 4.

In this perspective, the problem is quite different, since each resource/machine has to solve his own Single Machine Scheduling Problem, instead of EJSSP.

However, in this concurrent approach, each resource agent, through CBR module, uses his own and independent technique with self-tuned parameters. This is an important different opposing with global perspective, since each resource agent can use a different MH and parameters, and does not know which techniques are used by other agents.

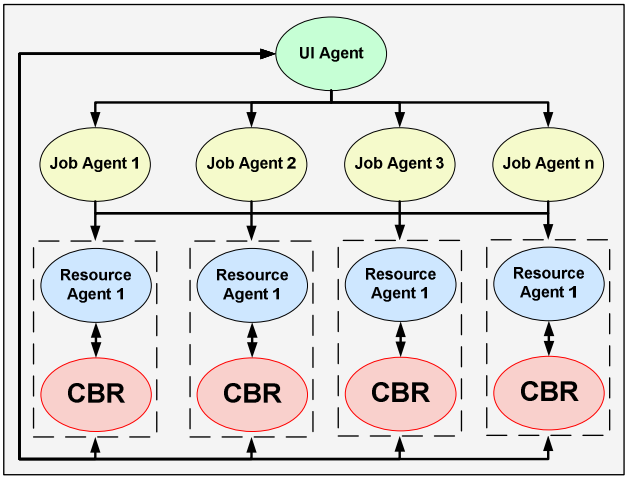

Figure 4. CBR Resource agents' integration

The CBR cycle has the same structure of global perspective, working in the same way, like previously described.

V. COMPUTATIONAL STUDY

The objective of this computational study is to analyze the CBR integration and reach some conclusions about which perspective is better, comparing with previous obtained results (without the integration of CBR) and comparing the two perspectives together.

For the computational study, it were used all instances from OR-Library Job-Shop Scheduling problems [32] (a total of 82 instances), proposed by Adams, Balas and Zawack [10], Fisher and Thompson [34], Lawrence, Applegate and Cook [36], Storer, Wu and Vaccari [37], and Yamada and Nakano [38]. These instances cover problems with 10, 20, 30, and 50 jobs, processed by 5, 10, 15, and 20 machines. All instances were executed five times (before and after CBR integration)

The machine used for the computational study is a HP Z400 Workstation, with the following main characteristics: Intel® Xeon® CPU W3565 @ 3.20 GHz, 6GB RAM, Samsung HD103SJ disk with 1TB, and Windows 7 64-bit.

A. *Global application*

The comparison between previous obtained results (before the introduction of CBR in the system) and global perspective's obtained results is shown in Figure 5.

Analyzing this figure, average obtained results were improved by 22% and best obtained results were improved by 26%. We consider it a very good evolution of the results, since the previous obtained results were consider to be very good.

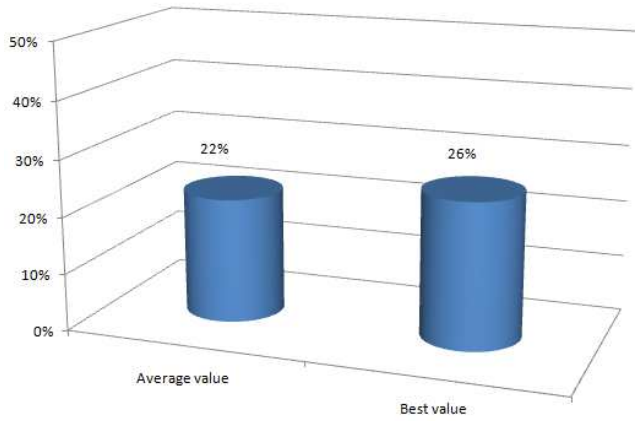

Figure 5. Percentage of obtained results improvement on global perspective

B. *Resource agents' application*

The comparison of obtained results before the introduction of CBR in the system with resource agents perspective is shown in Figure 6.

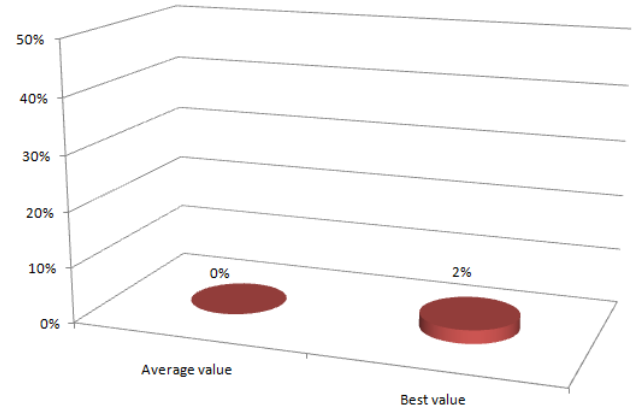

Figure 6. Percentage of obtained results improvement on resource agents' perspective

Analyzing this figure, we conclude that the improvement of results was very poor, since average results were not improved at all and best results were improved only by 2%. It is a sign that this perspective is not adequate for the system.

C. *Global perspectiv vs Resource agents' perspective*

Comparing the two perspectives, it will be possible to know which one has more merit to integrate the system.

Figure 7 compares the average obtained results between perspectives. The average obtained results from global perspective were all better when comparing with resource agents' perspective.

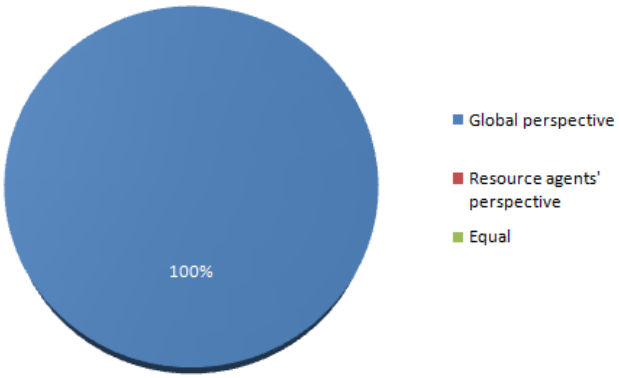

Figure 7. Comparison of average obtained results between perspectives

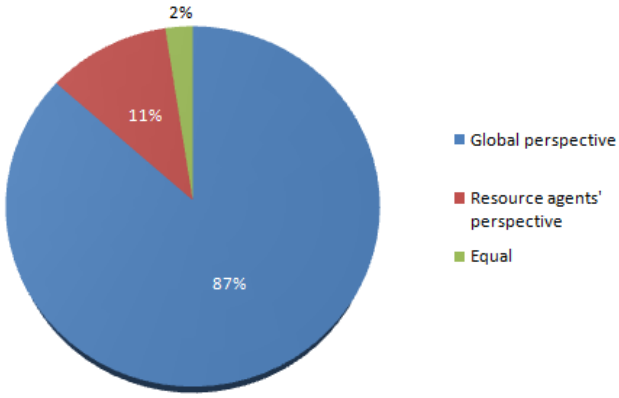

Figure 8. Comparison of best obtained results between perspectives

Comparing the best obtained results (Figure 8), it is possible to conclude that best results from global perspective were 87% better and only 11% were better when executing resource agents' perspective. The obtained results were the same in 2% of instances.

With this obtained results it is possible to conclude that global perspective, where the learning process is based in one apprentice, is better for integrating the system, and, for this system, a team learning approach revealed better results than a concurrent learning approach.

VI. Conclusions and Future Work

The paper addressed the use of CBR to self-parameterize MHs for the resolution of Scheduling problems, integrated on a MAS. With CBR, the system is able to choose and parameterize MH autonomously, for the resolution of static or dynamic Scheduling problems.

Two perspectives of CBR integration were proposed. The first one was based on a team learning approach, with CBR being integrated in a global perspective, choosing and self-parameterizing the same MH for every resource agents . The second perspective was based on a concurrent learning approach, where every resource agent has his own CBR module, choosing and parameterizing MHs for the resolution of his own local SMSP problem.

Analyzing the computational study performed, it was possible to conclude that global perspective revealed better results when comparing with resource agents' perspective. With this, we conclude that, in our system, a team learning approach is better than a concurrent learning.

Future work includes improvements on CBR global perspective and the implementation of other learning techniques, to compare with CBR.

ACKNOWLEDGMENT

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] Madureira, A., Santos, J. and Pereira, I., A Hybrid Intelligent System for Distributed Dynamic Scheduling, Natural Intelligence for Scheduling, Planning and Packing Problems, Studies in Computational Intelligence, eds. Chiong, R. and Dhakal, S., volume 250, 295-324, 2009.

[2] Portmann, M.C., Scheduling Methodology: optimization and compu-search approaches, The planning and scheduling of production systems, Chapman & Hall, 1997.

[3] Ouelhadj, D. and Petrovic, S., A Survey of Dynamic Scheduling in Manufacturing Systems, Journal of Scheduling, 2008.

[4] Madureira, A., Santos, J., Fernandes, N. and Ramos, C., Proposal of a Cooperation Mechanism for Team-Work Based Multi-Agent System in Dynamic Scheduling through Meta-Heuristics, IEEE Intern. Symp. on Assembly and Manufacturing (ISAM07), pp. 233-238, 2007.

[5] Wellner, J. and Dilger, W., Job shop scheduling with multiagents, in Workshop Planen und Konfigurieren, 1999.

[6] Monostori, L., J. Váncza and Kumara, S., Agent based systems for manufacturing, CIRP Annals - Manufacturing Technology, vol. 55, no. 2, pp. 697–20, 2006.

[7] Alonso, E., D'inverno, M., Kudenko, D., Luch, M., and Noble, J., Learning in Multi-agent Systems, The Knowledge Engineering Review, volume 16 (3), pp.277-84, 2001.

[8] Panait, L. and Luke, S., Cooperative Multi-Agent Learning: The State of the Art, Autonomous Agents and Multi-Agent Systems, pp.387-434, 2005.

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

[10] Adams, J., Balas, E. and Zawack, D., The shifting bottleneck procedure for job shop scheduling, Management Science 34, 391-401, 1988.

[11] Gonzalez, T., Handbook of Approximation Algorithms and Metaheuristics, Chapman&Hall/Crc Computer and Information Science Series, 2007.

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

[13] Beddoe, G., Petrovic, S. and Li, J., A hybrid metaheuristic case-based reasoning system for nurse rostering. Journal of Schedling 12, 2, April, 99-119, 2009.

[14] Schank, R., Dynamic memory; a theory of reminding and learning in computers and people. Cambridge University Press, 1982.

[15] Gentner, D., Structure mapping - a theorical framework for analogy. Cognitive Science, 7, 155-170, 1983.

[16] Porter, B. and Bareiss, R., PROTOS: An experiment in knowledge acquisition for heuristic. In Proceedings of the First International Meeting on Advances in Learning (IMAL), Les Arcs, France, 1986.

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

[18] Aamodt, A. and Plaza, E., Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, Artificial Intelligence Communications, 7, 39-52, 1994.

[19] Burke, E.K., MacCarthy, B.L., Petrovic, S. and Qu, R., Knowledge Discovery in a Hyper-Heuristic for Course Timetabling Using Case-Based Reasoning. PATAT 2002, 2002.

[20] Schmidt, G., Case-based reasoning for production scheduling. International Journal of Production Economics, 56-57, 537-546, 1998.

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

[22] Priore, P., de la Fuente, D. and Pino, R., Learning-Based Scheduling of Flexible Manufacturing Systems Using Case-Based Reasoning, Applied Artificial Intelligence 15 (10), pp. 949-963, 2001.

[23] Coello, J.M.A. and Santos, R.C., 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.

[24] MacCarthy, B.L. and Jou, P., Case-based reasoning in scheduling. In MK, K. & CS, W., eds. Proceedings of the Symposium on Advanced Manufacturing Processes, Systems and Techniques (AMPST96), MEP Publications Ltd, 1996.

[25] Dong, Y. and Kitaoka, M., Implementation of a case-based production scheduling system in C language, Computers & Industrial Engineering, Volume 27, Issues 1-4, pp. 265-268, 1994.

[26] Burke, E., MacCarthy, B., Petrovic, S. and Qu, R., Multiple-Retrieval Case-Based Reasoning for Course Timetabling Problems, Journal of Operations Research Society, 57 (2), pp. 148-162, 2006.

[27] Oman, S. and Cunningham, P., Using case retrieval to seed genetic algorithms. International Journal of Computational Intelligence and Applications, 1, 1, 71-82, 2001.

[28] Cunningham, P. and Smyth, B., Case-Based Reasoning in Scheduling: Reusing Solution Components. The International Journal of Production Research, 35, 2947-2961, 1997.

[29] Louis, S. and Xu, Z., Genetic Algorithms for Open Shop Scheduling and Re-Scheduling, ISCA 11th Intl. Conf. on Computers and their Applications, pp. 99-102, 1996.

[30] Madureira, A. and Pereira, I., Self-Optimization for Dynamic Scheduling in Manufacturing Systems, Technological Developments in Networking, Education and Automation, pp. 421-426, 2010.

[31] Horling, B. and Lesser, V., A survey of multi-agent organizational paradigms, Knowl. Eng. Rev., volume 19(4), pp.281-316, 2004.

[32] OR-Library, http://people.brunel.ac.uk/~mastjjb/jeb/info.html

[33] Adams, J., Balas, E. and Zawack, D., The shifting bottleneck procedure for job shop scheduling, Management Science 34, 391-401, 1988.

[34] Fisher, H. and Thompson G.L., Probabilistic learning combinations of local job-shop scheduling rules, Muth, J.F. and Thompson, G.L. (eds.), Industrial Scheduling, Prentice Hall, pp. 225-251, 1963.

[35] Lawrence, S., Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement), Carnegie-Mellon University, Pittsburgh, 1984.

[36] Applegate, D. and Cook, W., A computational study of the job-shop scheduling instance, ORSA Journal on Computing 3, 149-156, 1991.

[37] Storer, R.H., Wu, S.D. and Vaccari, R., New search spaces for sequencing instances with application to job shop scheduling, Management Science 38, 1495-1509, 1992.

[38] Yamada, T. and Nakano, R., A genetic algorithm applicable to large-scale job-shop instances, R. Manner, B. Manderick (eds.), Parallel instance solving from nature 2, North-Holland, 281-290, 1992.