

Scheduling a Cutting and Treatment Stainless Steel Sheet Line with Self-Management Capabilities

Ana Madureira^{1,2}, Ivo Pereira^{1,2}, Nelson Sousa^{1,2}, Paulo Ávila² and João Bastos²

¹ GECAD – Knowledge Engineering and Decision Support Group Porto, Portugal, (email: {amd, iasp, nffs}@isep.ipp.pt)

² Institute of Engineering – Polytechnic of Porto, Porto, Portugal, (email: {psa, jab}@isep.ipp.pt)

Abstract With advancement in computer science and information technology, computing systems are becoming increasingly more complex with an increasing number of heterogeneous components. They are thus becoming more difficult to monitor, manage, and maintain. This process has been well known as labor intensive and error prone. In addition, traditional approaches for system management are difficult to keep up with the rapidly changing environments. There is a need for automatic and efficient approaches to monitor and manage complex computing systems. In this paper, we propose an innovative framework for scheduling system management by combining Autonomic Computing (AC) paradigm, Multi-Agent Systems (MAS) and Nature Inspired Optimization Techniques (NIT). Additionally, we consider the resolution of realistic problems. The scheduling of a Cutting and Treatment Stainless Steel Sheet Line will be evaluated. Results show that proposed approach has advantages when compared with other scheduling systems.

1 Introduction

Manufacturing scheduling can be defined as the allocation, over the time, of jobs to machines, within a short temporal horizon and according to a specific criterion, such as cost or tardiness. It is a complex combinatorial problem, more specifically a non-polynomial (NP) problem: the objective is to find the optimal sequence from the $j! possible schedules, where j is the number of jobs and m the number of machines. Manufacturing scheduling domains are characterized by a great amount of uncertainty that leads to significant system dynamism. Such dynamic scheduling is receiving increased attention amongst both researchers and practitioners.$

Scheduling is an important aspect of automation in manufacturing systems. Most of scheduling domains are characterized by a great amount of uncertainty

that leads to significant system dynamics [Aytg et al. 2005]. Such dynamic scheduling is receiving increased attention amongst both researchers and practitioners [Aytug et al. 2005, Madureira 2003, Madureira et al. 2008, Madureira et al. 2007, Wellner and Dilger 1999]. However, scheduling is still having difficulties in real world situations and, hence, human intervention is required to maintain real-time adaptation and optimization.

Dynamic changes of a problem could arise from new user requirements and the evolution of the external environment. In a more general view, dynamic problem changes can be seen as a set of constraint insertions and cancellations.

For these dynamic optimization problems environments, that are often impossible to avoid in practice, the objective of the optimization algorithm is no longer to simply locate the global optimum solution, but to continuously track the optimum in dynamic environments, or to find a robust solution that operates optimally in the presence of perturbations [Aytug et al. 2005, Madureira et al. 2007]. In spite of all the previous trials, the scheduling problem is still known to be NP-complete, even for static environments. This fact poses serious challenges to conventional algorithms and incites researchers to explore new directions [Madureira 2003, Madureira et al. 2008, Madureira et al. 2007] and Multi-Agent technology has been considered an important approach for developing industrial distributed systems.

This paper addresses the use of Multi-Agent Systems paradigm for supporting dynamic and distributed scheduling in Manufacturing Systems with Autonomic properties, in order to reduce the complexity of managing systems and human interference. Additionally, we consider the resolution of realistic problems. The scheduling of a Cutting and Treatment Stainless Steel Sheet Line will be evaluated.

The remaining sections are organized as follows: in Section 2 Nature Inspired Optimization Techniques are presented. Section 3 describes Multi-Agent Systems paradigm. Section 4 summarizes some aspects and related work on Autonomic Computing. In section 5 the AutoDynAgents system is presented. Section 6 presents the description of the Production Process of the Cutting and Treatment Stainless Steel Sheet Line and the computational study. Finally, the paper presents some conclusions and puts forward some ideas for future work.

2 Nature Inspired Optimization Techniques

Many optimization problems in diverse fields have been solved using different optimization algorithms. Traditional optimization techniques such as linear programming (LP), non-linear programming (NLP), and dynamic programming (DP) have had major roles in solving these problems. However, their drawbacks generate demand for other types of algorithms, such as heuristic optimization approaches (Simulated Annealing, Tabu Search, and Evolutionary Algorithms).

However, there are still some possibilities of devising new heuristic algorithms based on analogies with natural or artificial phenomena or even the development of hybrid approaches.

The interest of the NIT based approaches is that they converge, in general, to satisfactory solutions in an effective and efficient way (computing time and implementation effort). NIT have often been shown to be effective for difficult combinatorial optimization problems appearing in various industrial, economical, and scientific domains [Gonzalez 2007, Madureira et al. 2007, Siarry 2008].

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.

3 Multi-Agent Systems

Multi-agent paradigm is emerging for the development of solutions to very hard distributed computational problems. This paradigm is based either on the activity of "intelligent" agents which perform complex functionalities or on the exploitation of a large number of simple agents that can produce an overall intelligent behavior leading to the solution of alleged almost intractable problems. The multi-agent paradigm is often inspired by biological systems that are based in the Social Systems interactions between agents and subject to negotiations.

Considering the complexity inherent to the manufacturing systems, dynamic scheduling is considered an excellent candidate for the application of agent-based technology.

The main term of Multi-Agent based computing is an Agent. However the definition of the term Agent has not common consent. In the last few years most authors agreed that this definition depends on the domain where agents are used. However there is a general consensus about its two main abstractions:

- An agent is a computational system that is situated in a dynamic environment and is capable of exhibiting autonomous and intelligent behaviour.
- An agent may have an environment that includes other agents. The community of interacting agents, as a whole, operates as a multi-agent system.

Some of most important common properties of computational agents are as follows [Wooldridge 2002]:

- act on behalf of their designer or the user they represent in order to meet a particular purpose.
- are autonomous in the sense that they control both their internal state and behaviour in the environment.
- exhibit some kind of intelligence, from applying fixed rules to reasoning, planning and learning capabilities.
- interact with their environment, and in a community, with other agents.
- are ideally adaptive, i.e., capable of tailoring their behaviour to the changes of the environment without the intervention of their designer.

Additional agent properties, characteristic in particular domains and applications are mobility (when an agent can transport itself to another environment to access remote resources or to meet other agents), genuineness (when it does not falsify its identity), credibility or trustworthiness (when it does not communicate false information wilfully) and sociality (when agents work in open operational environments hosting the execution of a multiplicity of agents, possibly belonging to different stakeholders).

In many implementations of MAS systems for manufacturing scheduling, the agents model the resources of the system and the tasks scheduling is done in a distributed way by means of cooperation and coordination amongst agents [Aytug et al. 2005, Bhat et al. 2006]. There are also approaches that use a single agent for scheduling that defines the schedules that the resource agents will execute [Abdelwahed and Kandasamy 2006, Bustard and Sterritt 2006]. When responding to disturbances, the distributed nature of multi-agent systems can also be a benefit to the rescheduling algorithm by involving only the agents directly affected, without disturbing the rest of the community that can continue with their work.

4 Autonomic Computing

Autonomic Computing is an IBM Grand Challenge proposed in 2001 by Paul Horn, Senior Vice-President of IBM Research [Horn 2001]. Horn argues that the Information Technology (IT) industry 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.

AC proposes a broad new field of research related to the automation of IT management processes, drawing inspiration from the human autonomous nervous system. From its inception, the concept revolves around four self-* properties, in which research efforts may be categorized: Self-Configuring, Self-Healing, Self-Optimizing and Self-Protecting. This number was by no means restricted, but no other proposals seem to have been made thus far.

Although the names of these properties are fairly self-explanatory, there is one inherent and implicit concept of significant importance: 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 to changes, and continuously seek ways to improve efficiency or counter negative environment changes.

Many studies have already been made around this area. These range from Software Engineering concerns to address this new development paradigm [Bustard and Sterritt 2006, Cervenka et al. 2006] all the way down to industry integration [Ganek 2006, IBM 2006].

Important techniques have been tapped into from areas such as Service-Oriented Architectures [Adams et al. 2006, Bhat et al. 2006, Chess et al. 2006] Schwan et al. 2006], Multi-Agent Systems [Kephart and Chess 2003], Grid Computing (Ganek 2006)(Kephart and Chess 2003) and Control Theory [Abdelwahed and Kandasamy 2006, Bhat et al. 2006].

The Organization for the Advancement of Structured Information Standards (OASIS) has also furthered the establishment of standards in the very important area of communication, mostly related to Web Services [Chess et al. 2006]. These have proven to be central in the quest for a communication middleware layer that can effectively abstract away the heterogeneity of underlying IT components.

Planning is a critical component of the Autonomic Computing vision [Kephart and Chess 2001] 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. In the AC vision [Kephart and Chess 2001], four aspects of self-* are distinguished:

- *Self-configuration*: deals with installation, configuration and integration of IT systems. The installation procedures work by gathering information about the host environment, figuring out the dependencies among needed tasks and also optimizing performance measures, and finally executing the tasks to realize the changes. Information about host system is increasingly getting standardized along structured formats but the executable tasks can be adhoc scripts. Humans want to be closely involved in key decisions during execution.
- *Self-optimizing*: deals with improving the performance of running systems by leveraging alternative opportunities. The system would monitor its performance and based on its changing environment, could initiate new changes (e.g., resource re-provisioning).
- *Self-healing*: deals with determination of problematic situations and recovering from them. It requires the system to reason with how activities can be performed, how diagnostic information is produced and how new changes can be affected with minimal cost and maximum benefit. The specification of actions could be known at some level of granularity.

- *Self-protecting*: deals with monitoring the environment for threats and responding to them. It is related to self-optimizing aspect but with the difference that the situation needs time-bound response and lead to cascading effect.

5 AutoDynAgents System

AutoDynAgents 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.

The main purpose of AutoDynAgents is a Multi-Agent System 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 Genetic Algorithms, Tabu Search or other NIT; 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.

Scheduling approach followed by AutoDynAgents system 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 NIT. This considers a specific kind of social interaction that is cooperative problem solving (CPS), where the group of agents work together to achieve a good solution for the problem.

The original scheduling problem defined in [Madureira 2003, Madureira et al. 2007], is decomposed into a series of Single Machine Scheduling Problems (SMSP). The Resource Agents (which has an NIT associated) obtain local solutions and later cooperate in order to overcome inter-agent constraints and achieve a global schedule.

Two possible approaches, to deal with this problem, could be used. In the first, the AutoDynAgents system waits for the solutions obtained by the Resource Agents and then apply a repair mechanism to shift some operations in the generated schedules till a feasible solution is obtained (Repair Approach). In the second, a coordination mechanism is established between related agents in the process, in order to interact with each other to pursuit common objective through cooperation. These coordination mechanism must be prepared to accept agents subjected to dynamism (new jobs arriving, cancelled jobs, changing jobs attributes).

The AutoDynAgents system architecture (Fig. 2.1 and 2.2) is based on six different types of agents.

In order to allow a seamless communication with the user, an *User Interface Agent* was implemented. This agent, apart from being responsible for the user interface, generates the necessary Job Agents dynamically according to the number of tasks that comprise the scheduling problem and assign each task to the respective Task Agent.

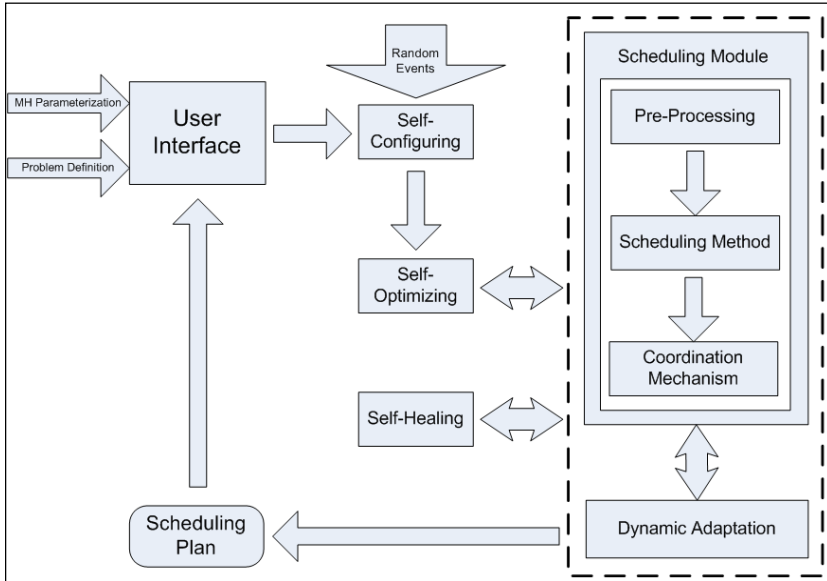


Fig 2.1 AutoDynAgents system architecture

The *Task Agent* will process the necessary information about the job. That is to say that 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 Machine Agent is responsible for solving a specific conflict.

The *Machine Agent* is 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 Task Agent for later feasibility check.

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 a dynamic adaptation. With this agent, the system will be prepared to automatically handle dynamism by adapting the solutions to external perturbations. While, on one hand, partial events only require a redefinition of jobs' attributes and re-evaluation of the objective function, on other hand, total events require changes on the solution's structure and size, carried out by insertion or deletion of operations, and also re-evaluation of the objective function. Therefore,

under total events, 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 dynamic adaptation of population/neighborhood.

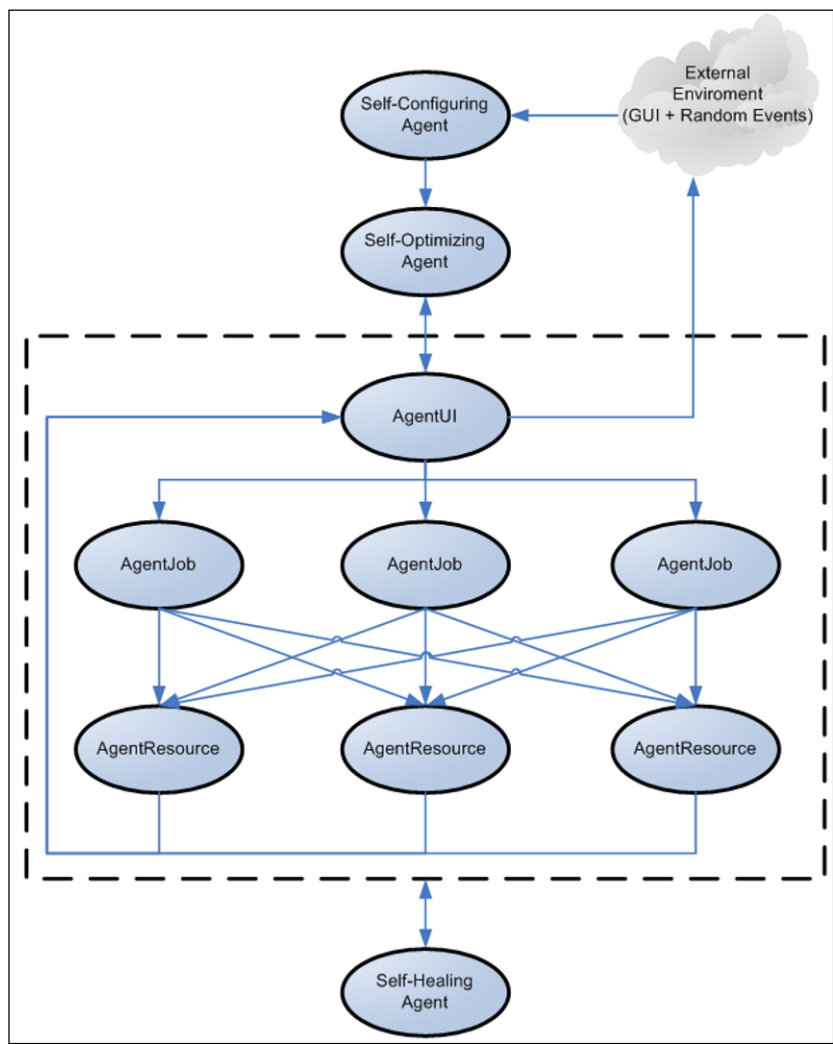


Fig 2.2. AutoDynAgents agents' architecture

The *Self-Optimizing Agent* is responsible for the automatically tuning of the meta-heuristics' parameters, according to the problem. This agent receives the ini-

tial problem, or the changes detected by Self-Configuring Agent, and automatically choose the meta-heuristic to use, and makes its self-parameterization. If some dynamism occurs, parameters may change in run-time. This tuning of parameters is made through learning and experience, since it uses a Case-based Reasoning (CBR) module. Each time a new problem (case) appears, the 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.

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.

Rescheduling is necessary due to two classes of events: Partial events imply variability in jobs/operations attributes such as processing times, due dates or release times; and Total events imply variability in neighborhood/population structure, resulting from new job arrivals, job cancellations, machines breakdown, etc.

6 Case Study: Cutting and Treatment Stainless Steel Sheet Line

The scheduling systems for the Cutting and Treatment Stainless Steel Sheet Line have different objectives and constraints and operate in an environment where there is a substantial quantity of real-time information concerning production failures and customer requests. At this stage, the main objective is the effective and efficient resolution of the Scheduling of Cutting and Treatment Stainless Steel Sheet Line.

At this work the AutoDynAgents has been evaluated only from its scheduling system perspective on a deterministic environment. Autonomic behaviour was not evaluated at this stage.

6.1 Description of the Production Process

The production process under study is a Job-Shop like scheduling problem, which pretends the processing the cut of plates of stainless steel with or without superficial treatment based on gridding. The number of different final products is very large, i.e., each order is normally different from another, because the enterprise

works majority with product specifications (dimensions and type of superficial treatment) by order. By this reason, the enterprise scheduling problem is sequence dependent of setup times and its optimization is very important for the system competitiveness, due to the setup time, which varies from 7 minutes to 18 minutes.

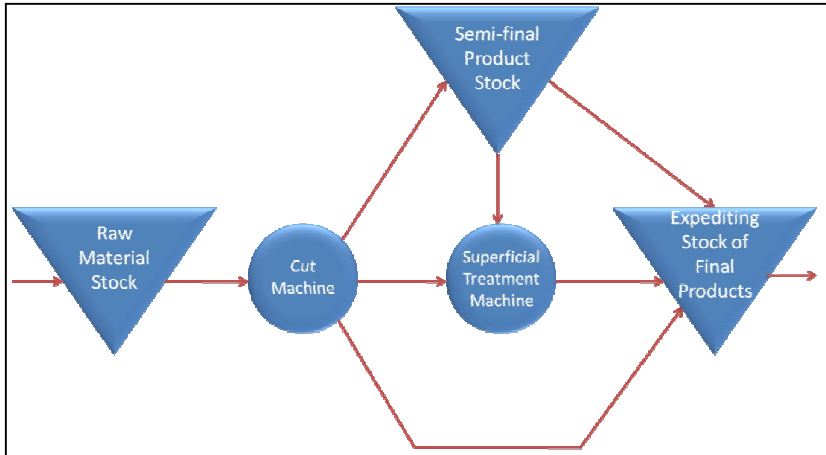


Fig 2.3. Production Process Flow

The production system is constituted by two machines (Fig. 2.3): the cut machine, automatic, and the superficial treatment machine, semi-automatic; and by three storage zones: raw material stock, semi-final product stock, and expediting stock of final products. The routing of the products, showed in the figure below that represents the process chart, can be of four different ways. The principal one is the horizontal, passing through the two machines. The second one begins in the raw material stock, pass through the cut machine and finish in the expediting stock of final products. The third one begins in the raw material stock, pass through the cut machine, then goes through the semi-final product stock and finish in the expediting stock of final products. The fourth one begins in the semi-final product stock, pass through the superficial treatment machine and finish in the expediting stock of final products.

6.2 Scheduling Problem Description

The production orders, more than 10 per day in average, are obtained through the current planning system (ERP) and comprehend the following information: product type, quantity, and production date. After that, the production orders, just allocated to the routing process, are transferred to the scheduling tool, the *Izaro Grey*

from the *Softi9* enterprise, that sequence the orders attending the minimization of the *makespan*.

The organization scheduling problem is performed by the *Izaro Grey* based on the following assumptions coherent with its reality:

- All the n jobs are independent and available for processing at the initial time, according to the ERP results for a time horizon of nine days;
- Are previously known the delivery date for all the jobs;
- The allocation of the orders to the machines it is already predefined, by that, the scheduling problem resume to the Sequencing/Dispatching problem;
- Job setup times are sequence dependent and deterministic;
- For all the jobs, the processing times at each operation are known and deterministic;
- Pre-emption is not allowed;
- Do not exist precedence restrictions between operations of different jobs;
- One machine can process only one job at a time and one job can be processed by only one machine at any time;
- Each job have a linear production process;
- Each job have equal priority order;
- The production scheduling is static during a period of one to three days;
- The scheduling objective is to minimize the *makespan*;
- The production work station has sufficient capacity to store and manage the work-in-process inventory generated during the execution of the complete set of jobs. That is, we assume infinite capacity at each stage;
- All the three storage zones referred in the routing of the products have sufficient capacity;
- Travel time between consecutive stages is negligible.

According to the literature [Pinedo 2008], the scheduling problem of this enterprise is denoted by $J_m || C_{max}$, that designates a job shop problem with m machines, in our case two. There is no recirculation, the job visits each machine at most once and the objective is the minimization of the *makespan*.

6.3 Simulation Plans and Computational Results

In order to analyse the performance of the AutoDynAgents with the scheduling systems, Izaro APS2, which is used at the enterprise and Lekin3, we selected a pe-

² <http://www.softi9.pt/images/download/doc32.pdf>

³ <http://www.stern.nyu.edu/om/software/lekin/index.htm>

riod of nine days of production orders (period used by the enterprise to scheduling performing). Considering that Lekin has capacity restrictions in the total number of production orders, we had considered two simulation plans: Plan 1 – Izaro versus AutoDynAgents for all the 90 production orders; Plan 2 – Lekin versus AutoDynAgents for the first 62 production orders.

Table 2.1 Simulation Plans

Simulation Plans	Scheduling System		Number of Production Orders
Plan 1	Izaro	AutoDynAgents	90
Plan 2	Lekin	AutoDynAgents	62

Considering that scheduling systems under analysis are based in stochastic based algorithms, we have run ten simulations for each plan and choose the best solution for each scheduling system. The final results obtained for the each plans are showed in [tables 2.2](#) and [2.3](#).

Table 2.2 Simulation Results for the Plan 1

Scheduling Tools	Izaro	AutoDynAgents
Makespan (min)	4070	3950
Maximum Lateness (min)	591	0
Number of Late Jobs	8	0
Σ Lateness (min)	3018	0
Total Production Time (min)	4165	4305

Table 2.3 Simulation Results for the Plan 2

Scheduling Tools	Lekin	AutoDynAgents
Makespan (min)	2664	2549
Maximum Lateness (min)	0	51
Number of Late Jobs	0	1
Σ Lateness (min)	0	51
Total Production Time (min)	2618	2689

When comparing the obtained results by Izaro and AutoDynAgents systems ([Table 2.2](#)) we can conclude about the advantage in effectiveness obtained by AutoDynAgents in almost optimization measures, namely on the minimization of *makespan*, maximum *lateness*, number of late jobs and the sum of lateness. In simulation plan 2 ([Table 2.3](#)), AutoDynAgents system outperforms Lekin system when analyzed *makespan* optimization criteria, but with a degradation of lateness related optimization criteria. It is possible to conclude about the general advantage in effectiveness of AutoDynAgents over Izaro APS and Lekin scheduling systems.

7 Conclusions

We believe that a new contribution for the resolution of more realistic scheduling problems was described in this paper. A novel approach to scheduling resolution by combining Autonomic Computing, Multi-Agent Systems and Nature Inspired Optimization Techniques was proposed. The use of Multi-Agent Systems paradigm for supporting dynamic and distributed scheduling in Manufacturing Systems with Autonomic properties, in order to reduce the complexity of managing systems and human interference was supported. Additionally, we consider the resolution of realistic problems: the scheduling of a Cutting and Treatment Stainless Steel Sheet Line was evaluated.

The experimental results showed the performance of proposed scheduling on the several plans simulations advantage over the others scheduling systems under considerations.

Work still to be done includes the exhaustive testing of the proposed system and negotiation mechanisms under dynamic environments subject to several random perturbations.

Acknowledgments The authors would like to acknowledge FCT, FEDER, POCTI, POCI for their support to R&D Projects and GECAD - Knowledge Engineering and Decision Support Group Unit.

References

- Abdelwahed S, Kandasamy N (2006) A Control-Based Approach to Autonomic Performance Management in Computing Systems, in *Autonomic Computing – Concepts, Infrastructure, and Applications*
- Adams R, Brett P, Iyer S, Milojicic D, Rafaeli S, Talwar V (2006) Scalable Management – Technologies for Management of Large-Scale, Distributed Systems, in *Autonomic Computing – Concepts, Infrastructure, and Applications*
- Aytug H, Lawley MA, McKay K, Mohan S, Uzsoy R (2005) Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research*, Volume 16 (1), 86-110
- Bhat V, Parashar M, Kandasamy N (2006) Autonomic Data Streaming for High-Performance Scientific Applications, in *Autonomic Computing – Concepts, Infrastructure, and Applications*
- Bustard D, Sterritt R (2006) A Requirements Engineering Perspective on Autonomic Systems Development, in *Autonomic Computing – Concepts, Infrastructure, and Applications*.
- Cervenka R, Greenwood D, Trencansky I (2006) The AML Approach to Modeling Autonomic Systems, Whitestein Technologies, Presented at the International Conference on Autonomic and Autonomous Systems (ICAS), July 19-21, 2006, Silicon Valley, USA
- Chess D, Hanson J, Kephart J, Whalley I, White S (2006) Dynamic Collaboration in Autonomic Computing, in *Autonomic Computing – Concepts, Infrastructure, and Applications*
- Ganek A (2006) Overview of Autonomic Computing: Origins, Evolution, Direction, in *Autonomic Computing – Concepts, Infrastructure, and Applications*

- Gonzalez T (2007) Handbook of Approximation Algorithms and Metaheuristics. Chapman&Hall/Crc Computer and Information Science Series
- Horn P (2001) Senior Vice-President, IBM Research. Autonomic Computing: IBM's Perspective on the State of Information Technology, IBM Research, October 2001
- IBM (2006) An Architectural Blueprint for Autonomic Computing, White paper by IBM, June 2006
- Kephart J, Chess D (2003) The Vision of Autonomic Computing, Computer, vol. 36, pp. 41-50
- Madureira A (2003) Meta-Heuristics Application to Scheduling in Dynamic Environments of Discrete Manufacturing. PhD Dissertation. University of Minho, Portugal. (in portuguese)
- Madureira A, Santos F, Pereira I (2008) Self-Managing Agents for Dynamic Scheduling in Manufacturing, GECCO'2008 (Genetic and Evolut. Comput. Conference 2008, Atlanta, Georgia (EUA))
- Madureira A, Santos J, Fernandes N, Ramos C (2007) 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), Ann Arbor (USA), pp. 233-238, ISBN: 1-4244-0563-7
- Pinedo M (2008) Planning and Scheduling in Manufacturing and Services, Springer Series in Operations Research and Financial Engineering, Springer
- Schwan K, Cooper B, Eisenhauer G, Gavrilovska A, Wolf M, Abbasi H, Agarwala S, Cai Z, Kumar V, Lofstead J, Mansour M, Seshasayee B, Widener P (2006) AutoFlow: Autonomic Information Flows for Critical Information Systems, in Autonomic Computing – Concepts, Infrastructure, and Applications
- Siarry P (2008) Advances in Metaheuristics for Hard Optimization. Springer-Verlag
- Wellner J and Dilger W (1999) Job shop scheduling with multiagents, in Workshop Planen und Konfigurieren
- Wooldridge M (2002) An Introduction to Multiagent Systems, John Wiley and Sons

Computational Intelligence for Engineering Systems

Emergent Applications

Madureira, A.; Ferreira, J.; Vale, Z. (Eds.)

2011, VIII, 196 p., Hardcover

ISBN: 978-94-007-0092-5