5 A HOLONIC APPROACH TO DYNAMIC MANUFACTURING SCHEDULING

Paulo Leitão[1], Francisco Restivo[2]
[1]*Polytechnic Institute of Bragança,*
Quinta Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal
pleitao@ipb.pt
[2]*Faculty of Engineering of University of Porto,*
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
fjr@fe.up.pt

Manufacturing scheduling is a complex combinatorial problem, particularly in distributed and dynamic environments. This paper presents a holonic approach to manufacturing scheduling, which in opposite to traditional approaches, distributes the scheduling functions over several entities, combining their calculation power and local optimization. In this scheduling and control approach, the scheduling mechanism evolves dynamically to combine optimized scheduling, achieved by central entities, and distributed scheduling, improving its responsiveness and robustness.

1. INTRODUCTION

Manufacturing scheduling can be defined as the allocation, over the time, of jobs to machines, within a shorter temporal horizon and respecting 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 $j!^m$ possible scheduling sequences, where j is the number of jobs and m the number of machines. The manufacturing scheduling problem becomes even more complex when it takes place in an open, distributed and dynamic environment.

The scheduling problem has been widely studied, mainly due to its highly combinatorial aspects, its dynamic nature and its applicability in manufacturing systems [1]. Examples of such methods are heuristics, linear programming, constraint satisfaction techniques, Lagrangian relaxation, neighbourhood search techniques (e.g. simulation annealing or taboo search) and genetic algorithms.

Manufacturing scheduling is traditionally elaborated in a centralized manner using one of referred methods, often calculated off-line and considering that it is a static and deterministic problem. However, in an industrial manufacturing system the things rarely go as expected, mainly because: i) new tasks arrive continuously to the system, while scheduled tasks are cancelled, ii) certain resources become unavailable and additional resources are introduced, iii) unexpected events occur in the system, such as machine failures, operator absence, rush orders or unavailability of raw-materials, and iv) scheduled tasks may take more or less time than expected.

Please use the following format when citing this chapter:

Leitão, P., Restivo, F., 2006, in IFIP International Federation for Information Processing, Volume 220, Information Technology for Balanced Manufacturing Systems, ed. Shen, W., (Boston: Springer), pp. 37–46.

In such dynamic environments, the optimised schedule produced by the front office can quickly become unacceptable, requiring the dynamic re-scheduling, as fast as possible, and done in a short amount of time, to avoid the risk of degradation of the production productivity. Traditional methods don't fulfil the real dynamic re-scheduling needs mainly because they are inflexible and slow.

Agent-based and holonic manufacturing approaches suggest the implementation of distributed scheduling, since the scheduling algorithm maybe easily distributed over a number of entities which combine their calculation power and their local knowledge to optimize the global performance [2]. Unlike traditional manufacturing scheduling approaches, using centralised scheduler, in agent-based manufacturing scheduling systems, each agent can locally handle the schedule of its machine, operator, robot or station. The major advantages of the distributed scheduling are the improvement of reaction to disturbances and the parallel computation.

Some of the distributed scheduling approaches use traditional algorithms embedded in distributed entities, while others are based on emergent behaviour, like market-based and net protocol algorithms. Among others, Sousa and Ramos [3] proposes a dynamic scheduling system supported by a holonic approach, using forward and backward influence in the negotiation leading to the task allocation, to handle the temporal constraints and to solve conflicts, and Gou et al. [4] presents a holonic manufacturing scheduling approach using Lagrangian relaxation, where capacity constraints of a scheduling problem can be relaxed and replaced by a penalty cost. Markus et al. [5] proposes a market model to solve dynamic order processing and scheduling problems, and Sugimura et al. [6] models the manufacturing operations using an object-oriented approach and proposes a real time scheduling mechanism for assembly lines. Hino and Moriwaki [7] introduces a recursive propagation technique based on sending messages regarding schedule changes to agents responsible for subsequent tasks, and Logie et al. [8] extends this concept by limiting the focus of the agents to tasks within a specified time window.

This paper describes a holonic approach to the dynamic manufacturing scheduling, introduced by ADACOR (ADAptive holonic COntrol aRchitecture for distributed manufacturing systems) [9], which combines distributed scheduling, where holons negotiate the resource allocation using free market based techniques, thus achieving fast re-scheduling, with coordination entities, responsible for the introduction of global optimization. The holonic dynamic scheduling is supported by decision-making capabilities embedded in each distributed holon and cooperation mechanisms that support the evolution of the dynamic scheduling.

The rest of the paper is organized as follows: Section 2 presents the main principles of the proposed holonic scheduling architecture, focusing on the distributed components, dynamic adaptation model and distributed scheduling. Section 3 describes the cooperation mechanisms to support the dynamic re-scheduling and Section 4 describes the prototype implementation. Finally, Section 5 rounds up the paper with conclusions.

2. HOLONIC DYNAMIC SCHEDULING ARCHITECTURE

The idea beyond our scheduling approach is that a global optimized schedule should be generated whenever possible, and a fast re-scheduling should be used in case of disturbances, because, in this case, this is preferable to waiting a significant amount of time for an optimized schedule, which is likely to be not optimized again soon.

2.1 Holonic Manufacturing Components

The proposed approach is designed upon a community of distributed and autonomous control units, the holons, representing the manufacturing components. Three types of holons are identified to handle the scheduling and control at shop floor level [9], as illustrated in the Figure 1:

- *Task holons* that represent production orders launched to the shop floor to execute products, each one containing information about the production of the product, and about the progress of the production order execution.
- *Operational holons* that represent physical resources or operators available at shop floor, each one with a set of skills and knowledge.
- *Supervisor holons* that represent the logical coordination of a group of operational and/or supervisor holons, providing co-ordination and optimization services to the holons under their supervision, and thus introducing hierarchy in an otherwise decentralized system.

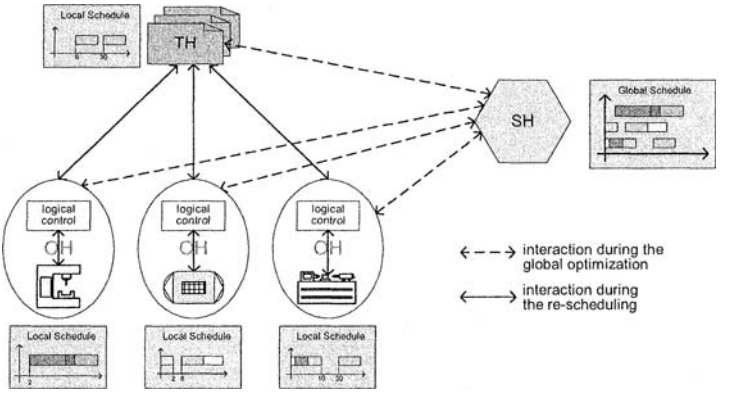

Figure 1 - Interaction between Distributed Manufacturing Components

Each holonic control unit has its own decision-making capability, performing, among others, control and scheduling functions. These embedded mechanisms are dependent of the holon's type, its behaviour and objectives.

The scheduling mechanism embedded in supervisor holons deals with the multiple machines and multiple jobs scheduling problem. Depending on the number of machines and jobs, the schedule produced by the top level supervisor holon may take a large computational effort and time, but it is an optimal plan since the supervisor has a global view of the system.

Each operational holon is responsible for its own schedule, built dynamically from its local knowledge, and using a scheduling mechanism addressing the problem of multiple jobs for a single machine. It achieves optimal local schedule plans, but due to the lack of global information, may not lead to an optimal global schedule.

The motivation of ADACOR holons to execute the manufacturing actions is regulated by a credits system. Each task holon when is launched receives a fund (π) to execute a production order and a penalty value for the delay. The task holons are responsible for managing the fund received to produce their production orders, without exceeding the initial fund or paying the delay penalty.

During the interaction to allocate the operations, task holons try to pay as less as possible and the operational holons try to receive as more as possible. After the negotiation, each task holon accepts to pay a price of ξ credits to the operational holon that will execute a certain operation and to receive a penalty of ξ credits from the operational holon if it does not fulfil the contracted due date.

Table 1 summarises the evolution of the credits of task and operational holons during their life cycles.

Table 1: Evolution of Credits during the Holon Life Cycle

Phase	Task Holon	Operational Holon
Resource allocation process.	Contracts the execution by ξ and the penalty by φ.	Contracts the execution by ξ and the penalty by φ.
Finish of an operation with success.	Pays the value ξ ($\pi \leftarrow \pi - \xi$).	Increases its credits by ξ ($\mu \leftarrow \mu + \xi$).
End of an operation with delay.	Pays the value ξ and receives the value φ ($\pi \leftarrow \pi - \xi + \varphi$).	Decreases its credits by φ and increases by ξ ($\mu \leftarrow \mu + \xi - \varphi$).
Operation cancelled (delay, failure, etc.)	Receives the value φ ($\pi \leftarrow \pi + \varphi$).	Decreases its credits by φ ($\mu \leftarrow \mu - \varphi$).

The global performance of operational holons in terms of credits is given by the sum of rewards received minus the penalties paid for the delays. These rewards and penalties reflect the *reputation* of the holon.

2.2 Dynamic Scheduling Model

The interaction process leading to the achievement of the manufacturing schedule has the following constraints:

- A part cannot be started until its preceding part(s) is finished.
- An operation cannot be started until its preceding operations are finished.
- Each machine can only process one operation at time t.
- A resource R_j possessing the set of skills S_j, has abilities to execute the operation o_{ik}, having a list of requirements $B_{ik} = \{B_{ikz} | z \in I\}$, if

$$B_{ik} \subseteq S_j \Leftrightarrow \forall x \in B_{ik} \Rightarrow x \in S_j$$

  i.e., the resource has abilities to execute an operation if it fulfils all the requirements presented by the operation.

The dynamic scheduling model is the result of the dynamic interaction between task, operational and supervisor holons, combining the problem solving at the individual holon level and the coordination-negotiation schema at the system level, to produce a global manufacturing scheduling, which is simultaneously centralized in normal situations or locally produced in the presence of a disturbance.

The self-organization capability is the key concept to support this adaptive production control and scheduling mechanism. The adaptation is achieved by the self-organization of each ADACOR holon, contributing to the dynamic adaptation of the whole system. The self-organization is regulated by the *autonomy factor*, which fixes the level of autonomy of each holon, and evolves dynamically in order to adapt the holon behaviour to the changes in the environment where it is placed. The evolution is governed by a decision mechanism, and the overall efficiency of the self-organization is dependent on how the learning mechanisms are implemented, and on how new knowledge influences its parameters.

Figure 2 illustrates a small example about how the adaptive dynamic scheduling approach works. In normal operation, i.e. without the occurrence of unexpected disturbances, the holons are running in a hierarchical structure, with supervisor holons coordinating several operational and/or supervisor holons, and operational holons having low autonomy factor. Periodically, regulated by their internal clocks, supervisor holons generate manufacturing scheduling plans globally optimized.

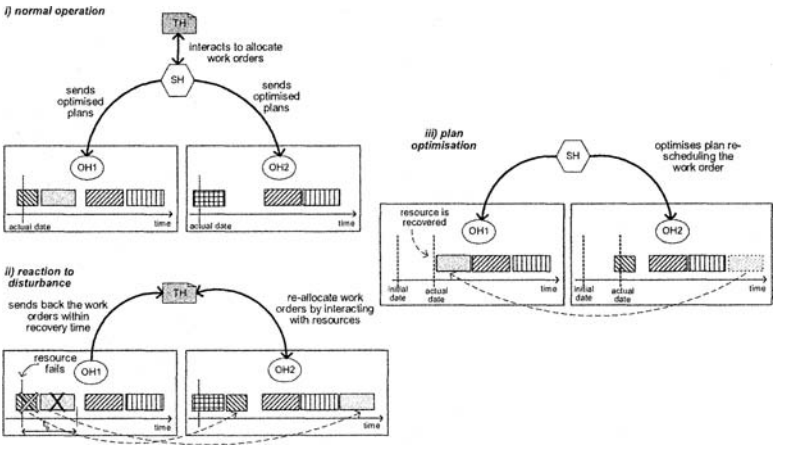

Figure 2 – Dynamic Holonic Scheduling

The optimised schedule plans are offered, as advices, to the holons under their coordination domains, which have the capability to accept or reject them. Normally, they follow the schedule advices proposed by the supervisor holons since they have low level of autonomy.

In turbulent scenarios or once an unexpected disturbance (e.g. a machine failure or deviation from plan) is detected, the system is forced to evolve to a heterarchical structure, characterized by totally decentralized decision-making mechanisms. In fact, the autonomy factor of each ADACOR holon is increased dynamically according to a function that takes in consideration its current value, the estimated time to recover from the disturbance, and the level of impact of the disturbance.

This transient state, which should be as short as possible, operates without the presence of coordination levels, the manufacturing scheduling being achieved in a distributed manner. The distributed scheduling results solely from the interaction between task and operational holons. The cooperation strategy built into each holon is therefore the key to the success of this approach.

After the transient phase, the system returns to a hierarchical structure.

2.3 Distributed Resource Allocation Schema

In the distributed resource allocation schema the computational complexity is related to find an optimal determination problem in combinatorial auctions. The distributed scheduling mechanism, introduced in this work, uses a resource allocation mechanism based on a multi-round Contract Net Protocol (CNP) [10], extending the original CNP schema with capability of apply the contract net schema several times,

and capability to contract partial quantities.

In presence of operation announcements, each operational holon decides, based on its skills and capacity, its availability to execute the operation. In case of availability, the operational holon then calculates the price to be proposed to the task holon. The price may be calculated according to the following function,

$$p_{jik} = C_s + C_p \times d_{ik} + C_b \times \left(2 - e^{-\sigma \times \beta} \times (1 - \gamma)\right)$$

which models the market laws, increasing or decreasing the final price in function of the actual load of the resource (reflected by the parameter β) and of the actual bid acceptance rate (reflected by the γ parameter, with $0 \leq \gamma \leq 1$). The holon uses the knowledge learned from the previous bids to adjust the final price: reducing the γ parameter if the acceptance rate is low or increasing it in the opposite case.

The task holon evaluates the proposals sent by operational holons to allocate the operation to the best bid. The decision procedure takes into account, among others, the proposed price, the location of the resource and the degree of confidence about the holon. The confidence degree reflects the trust that the task holon has in a operational holon, and is based on the knowledge learned from previous interactions. In case of an inconclusive evaluation, the task holon can start another iterative negotiation, re-formulating the bid parameters, for example the due date.

3. COOPERATION MECHANISMS FOR RE-SCHEDULING

The dynamic scheduling algorithm must respond dynamically and promptly to emergent and unexpected disturbances. An important design factor is the *size* of the disturbance that will activate the re-scheduling mechanisms described. Re-scheduling mechanisms can the be divided in: periodic re-scheduling, which considers all disturbances (usually many small disturbances) at once, generating new optimized schedules periodically, and event-based re-scheduling, which is more suited for bigger and single disturbances, like machine breakdowns or rush orders.

In the next sections, the several types of re-scheduling mechanisms (i.e. event-based and periodic) will be described.

3.1 Re-scheduling for Cancellation of Orders

The cancellation of a production order can be considered as a simple disturbance at shop floor level that only requires the local re-scheduling, in order to optimise the schedule. After generating a new schedule, the operational holon notifies the supervisor holon about its new schedule, allowing the synchronisation of both agendas and the optimization of the global schedule. It must be noticed that this type of disturbance may open free spaces in the agenda, allowing to execute earlier some operations that were eventually delayed.

3.2 Re-scheduling for Machine Breakdowns

In the case of occurrence of a machine failure, the operational holon determines the state of the machine and of the part after the failure, and estimates how long the downtime will be. The diagnostic can lead to different scenarios: the machine can become immediately available or stay out of service for a more delayed repair intervention, and the part may have been destroyed or not. If the part is not destroyed and the machine is ready to re-execute the operation, no action has to be

performed; however, other scheduled operation(s) may become delayed, being then treated as a delay disturbance.

In the other cases, both operational and task holons have specific tasks to perform. The operational holon:

- In case of destruction of the part, removes the proper operations from its agenda, and notifies the task and supervisor holons about the occurrence.
- In case of machine breakdown, notifies the task holon about its impossibility to execute the operations in the scheduled dates.

The operational holon also executes a re-scheduling to optimise the plan.

The task holon can take two different actions:

- If the part is destroyed, re-allocates from the beginning all operations belonging to the production order.
- If the machine became unavailable, re-schedules the operations taking in consideration the information from previous resource allocation processes. The achieved allocation can lead to delays in the posterior operations, requiring an adjustment of the temporal window to execute each operation.

In both cases, the re-scheduling is performed using the distributed resource allocation schema.

3.3 Re-scheduling for Delays

An operation delay can occur after a disturbance, when the operational holon can not fulfil the scheduled due date of an operation. In this situation, the operational holon notifies the task holon about the delay, proposing a new date. The decision about the acceptance of the operation delay is dependent of the actual state of the operation. If the operation is already in execution, this notification is seeing as a warning of delay, being necessary to re-schedule all the posterior operations affected by the delay. If the operation is waiting for the execution, the task holon can try to find alternative resources to allocate the operation by asking other operational holons about their capacity to execute the operation.

Based in the proposals sent by the operational holons and in the estimated delay, the task holon decides if accepts the proposal for the estimated delay or changes the allocation to another operational holon. In this last case, the operational holon removes the operation from its local schedule, and triggers a local scheduling optimization. Additionally, the task holon re-schedules the posterior operations, adjusting their scheduled start and due dates.

3.4 Re-scheduling for New Rush Orders

A rush production order is an order, usually of high priority, that arrives to the system and must be processed immediately, since it has a near due date. As the schedule plans are elaborated periodically by the supervisor holons, the treatment of these kinds of orders causes a disturbance in the system.

In this situation, the rush task holon interacts directly with the operational holons to allocate it, using the distributed resource allocation mechanism. The problem appears if the rush production order has to be executed in a time window already occupied by other tasks, which requires a special negotiation to relax the other operations and to introduce the new ones.

Since each order has associated a priority, operational holons take this

information in consideration. In the case that the rush order has maximal priority, i.e. that must be executed as soon as the current operation is completed, the operational holon tries to re-schedule, relaxing the operations that have minimal priority, i.e. those operations that can be delayed beyond the due date, to find capacity to execute the rush operation.

In case of impossibility to find capacity to allocate the rush production order, the task holon needs to negotiate with the task holons that have operations allocated to the resources occupying the requested time window to execute the rush order, trying to convince them to release some time window. This mechanism is based in the trade of credits units for rewards and penalties. The task holon can use the penalty value, that it will pay if does not fulfil the due date, to manage the problem of finding a time window to execute the rush operations.

In the negotiation process, the rush task holon interacts with other task holons, requesting the desired time window and offering a reward. Each one of the other task holons analyse the offer and in case that the reward covers the penalty that the task holon may to pay for the delay, it accepts; otherwise, it rejects the proposal. In case that all task holons reject the reward, the rush task holon must increase the reward value and make another offer. The task holon should repeat this procedure until one task holon accepts the offer or the offered reward value reaches the maximum value, which is equal to the penalty to be paid in case of delay.

In case that one task holon accepts the offer, it will notify the operational holons to decrease the priority to free the time window. In parallel, the rush task holon announces again the production order to the operational holons.

3.5 Re-scheduling for Optimization

After the execution of event-based re-scheduling, performed in a distributed manner, it is necessary to synchronise and optimise the elaborated schedule. The synchronization is required because supervisor holons don't know what kind of schedule was achieved during the distributed re-scheduling. For this purpose, lower-level operational holons notify the supervisor holon about its new schedule plan.

Since the current schedule was achieved in a fast but not optimised way, supervisor holons starts the optimization of the re-schedule plan achieved using the distributed scheduling schema. The elaboration of this optimized re-scheduling is performed in background and does not consider the operations included in a safe time window, as illustrated in the Figure 3.

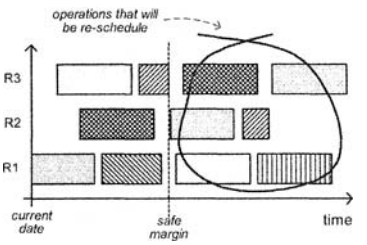

Figure 3 – Optimized Re-Scheduling after the Distributed Scheduling

This safe time window guarantees that the current schedule plan can be executed in the factory plant during the elaboration of the optimized re-schedule and is

defined according the estimated time required to optimize the schedule plan.

4. PROTOTYPE IMPLEMENTATION

The proposed holonic manufacturing scheduling approach was implemented in a prototype, using agent technology, namely the JADE (Java Agent Development Framework) framework. All three types of designed holons were implemented being the communication between them performed encoding messages according the FIPA-ACL (Agent Communication Language) communication language.

The decision component embedded in an ADACOR holon uses a rule-based system, applying declarative knowledge expressed in terms of rules, to regulate the holon's behaviour. For this purpose, it is used the JESS (Java Expert System Shell) rule-oriented programming infrastructure. The decision component also uses procedural knowledge, embodied in procedures that are triggered as actions by some rules. The scheduling algorithm is an example of this type of knowledge.

In the prototype, the scheduling mechanism embedded in the supervisor and operational holon uses simple algorithms that guarantees rapid and reliable scheduling. As the ADACOR architecture is built upon functional blocks, similar to Lego® components, these scheduling algorithms can be easily modified in the future, by plugging more powerful scheduling algorithms. In a similar way, it was developed the required mechanisms to implement the distributed scheduling. For this purpose, it was implemented the mechanisms for the propagation of re-organization using ant-based techniques and the factor of autonomy.

Figure 4 illustrates the system prototype for the flexible manufacturing system from CIM Centre of Porto, described in [11]. It shows graphically the optimized schedule elaborated by a supervisor holon and the local schedule performed by an operational holon.

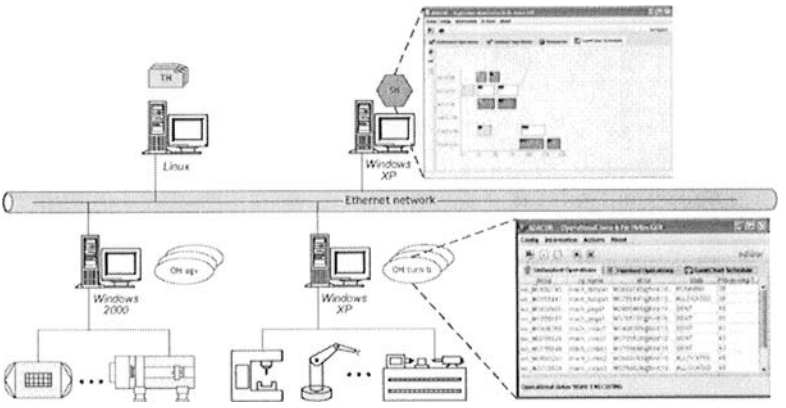

Figure 4 – The Central and Local Schedules in the Distributed ADACOR Entities

The prototype operation, showed in a first instance, the correctness and applicability of ADACOR control system, and particularity the holonic scheduling. It was also proved that the proposed holonic scheduling approach presents fast responsiveness and better flexibility, scalability and robustness, particularly for

unexpected situations. A set of experimental results were presented in [11], where this approach was evaluated and compared with other two different control approaches, namely hierarchical-like and heterarchical-like ones. The results showed that the proposed approach has potential to improve the system performance, mainly combining agility and global production optimization. The adaptive and holonic control and scheduling approach plays a crucial role to support this performance.

5. CONCLUSIONS

Manufacturing scheduling is traditionally elaborated in a centralized manner and doesn't consider the dynamic re-scheduling. This paper presented a holonic approach to dynamic manufacturing scheduling addressing the improvement of the fast re-scheduling maintaining the global optimization. The architecture is based in the following main foundations:

- Distributed approach, with decision-making distributed by a community of autonomous entities, each one having partial knowledge about the problem.
- In normal operation the scheduling is achieved in a central manner, using coordination entities to achieve optimization, and in turbulent operation, the scheduling is elaborated in a distributed manner aiming fast re-scheduling.
- The dynamic adaptive mechanism allows the evolution of the overall system in order to combine centralised and distributed scheduling strategies.

At this stage, the objective is not to have complex scheduling algorithms but to achieve fast re-scheduling combined with global optimization, using simple local scheduling algorithms embedded in the holons. In further work, the embedded local scheduling mechanisms will be improved in order to achieve high quality scheduling in a timely fashion.

6. REFERENCES

1. Shen, W. and Norrie, D., An Agent-based Approach Dynamic Manufacturing Scheduling. Workshop Notes of the Agent-based Manufacturing Workshop at Autonomous Agents, 1998.
2. Bongaerts, L., Integration of Scheduling and Control in Holonic Manufacturing Systems. PhD thesis, Katholieke Universiteit Leuven, Belgium, 1998.
3. Sousa, P. and Ramos, C., A Distributed Architecture and Negotiation Protocol for Scheduling in Manufacturing Systems. Computers in Industry, 38(2), 1999, pp. 103–113.
4. Gou, L., Luh, P. and Kyoya, Y., Holonic Manufacturing Scheduling: Architecture, Cooperation Mechanism and Implementation. Computers in Industry, 37, 1998, pp. 213-231.
5. Markus, A., Vancza, T. K., and Monostori, L. A Market Approach to Holonic Manufacturing. Annals of CIRP, 45, 1996, pp. 433–436.
6. Sugimura N., Hiroi M., Moriwaki T. and Hozumi K., A Study on Holonic Scheduling for Manufacturing System of Composite Parts. In Japan/USA Symp. on Flexible Manufacturing, 1996.
7. Hino, R. and Moriwaki, T. Decentralized Scheduling in Agent Manufacturing System. Proceedings of the 2nd International Workshop on Intelligent Manufacturing Systems, Belgium, 1999, pp. 41-47.
8. Logie, S., Sabaz, D. and Gruver, W. A. Sliding Window Distributed Combinatorial Scheduling using JADE. Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics Netherlands, 2004, pp. 1984-1989.
9. Leitão, P. and Restivo, F., ADACOR: A Holonic Architecture for Agile and Adaptive Manufacturing Control. Computers in Industry, 57 (2), 2006, Elsevier, pp. 121-130.
10. Smith, R., Contract Net Protocol: High-Level Communication and Control in a Distributed Solver. IEEE Transactions on Computers, C-29(12), 1980, pp. 1104-1113.
11. Leitão, P. and Restivo, F., Experimental Validation of ADACOR Holonic Control System. In V. Marík, R. Brennan and M. Pechoucek (eds.), Holonic and Multi-Agent Systems for Manufacturing, LNAI 3593, Springer-Verlag, 2005, pp. 121-132.