MASDScheGATS: A Prototype System for Dynamic Scheduling

Ana Madureira, Joaquim Santos, Ivo Pereira
Computer Science Department
Institute of Engineering - Polytechnic of Porto
GECAD – Knowledge Engineering and Decision Support Research Group
Porto, Portugal

Abstract

A manufacturing system has a natural dynamic nature observed through several kinds of random occurrences and perturbations on working conditions and requirements over time. For this kind of environment it is important the ability to efficient and effectively adapt, on a continuous basis, existing schedules according to the referred disturbances, keeping performance levels. The application of Meta-Heuristics and Multi-Agent Systems to the resolution of this class of real world scheduling problems seems really promising.
This paper presents a prototype for MASDScheGATS (Multi-Agent System for Distributed Manufacturing Scheduling with Genetic Algorithms and Tabu Search).

1. Introduction

In the last decade several trends were observed in manufacturing and society in general, namely: market globalization; increasing product/services customization, technology complexity, quality requisites and number of competitors. Stability, certainty and predictability gave place to change, uncertainty and unpredictability.

Traditionally scheduling resolution requires the intervention of highly skilled human problem-solvers. This is a very hard and challenging domain because current systems are becoming more and more complex, distributed, interconnected and subject to rapidly changing and even more disturbances. 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 optimal solution, but to continuously track the optimum, or to find a robust solution that operates optimally in the presence of perturbations [1][8].

Hybridization is a promising research field of computational intelligence focusing on combinations of multiple approaches to develop the next generation of intelligent systems. Recently, hybrid intelligent systems are getting popular due to their capabilities in handling several real world complexities involving imprecision, uncertainty and vagueness.

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.

Considering the complexity inherent to the manufacturing systems, dynamic scheduling is considered an excellent candidate for the application of agent-based technology. In many implementations of Multi-Agent System (MAS) for manufacturing scheduling, the agents model the resources of the system and the scheduling of tasks is done in a distributed way by means of cooperation and coordination amongst agents [6][10]. There are also approaches that use a single agent for scheduling that defines the schedules that the resource agents will execute [1], [2]. 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.

Meta-Heuristics form a class of powerful and practical solution techniques for tackling complex, large-scale combinatorial problems producing efficiently high-quality solutions. From the literature we can conclude that they are adequate for static problems. However, real scheduling problems are quite dynamic, considering the arrival of new orders, orders being cancelled, machine delays or faults, etc. Scheduling problem in dynamic environments have been investigated by a number of authors especially in the evolutionary community, see for example [4][9].

In this paper we will model a Manufacturing System by means of Multi-Agent Systems and Meta-Heuristics technologies, where each agent may represent a processing entity (machine). The objective of the system is to deal with the complex problem of Dynamic Scheduling in Manufacturing Systems. Our approach shows that a good global solution for a scheduling problem may emerge from a community of machine agents solving locally their schedules while cooperating with other machine agents that share some relations between the operations/jobs. Meta-Heuristics (Tabu Search or Genetic Algorithms) can be adapted to deal with dynamic problems, reusing and changing solutions/populations in accordance with the dynamism of the Manufacturing System.

Coordination Mechanisms are used to guarantee the feasibility of schedules. Notice that joining problems that were locally solved will not guarantee the feasibility of schedules (e.g. precedence relations could not be guaranteed). The cooperation mechanism will be established between machine agents involved in the execution of operations (jobs) with precedence relations in order to deal with the feasibility of the generated schedules in run-time.

Considering that the inherent nature of current manufacturing systems is distributed we will address the complex dynamic scheduling problems in a distributed way using the Multi-Agent paradigm. The proposed architecture is based on Team-Work characteristics due to its philosophy of cooperation.

Team-oriented programming suggests a number of different approaches to the definition of agent teams and their coordination in order to achieve common goals. Some MAS organizational aspects [3] are evaluated in order to define the proposed cooperation mechanism.

The remaining sections are organized as follows: in section 2 the scheduling problem under consideration is presented. Section 3 presents and describes MASDScheGATS Systems and describes implemented mechanisms. Finally, the paper presents some conclusions and puts forward some ideas for future work.

2. Problem Definition

Real world scheduling problems have received a lot of attention in recent years. In this work we consider the resolution of realistic problems. 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, many scheduling problems include further restrictions and relaxation of others [5]. Thus, for example, precedence constraints among operations of the different jobs are common because, often, mainly in discrete manufacturing, products are made of several components that can be seen as different jobs whose manufacture must be coordinated. Additionally, since a job can be the result of manufacturing and assembly of parts at several stages, different parts of the same job may be processed simultaneously on different machines (concurrent or simultaneous processing). Moreover, 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.

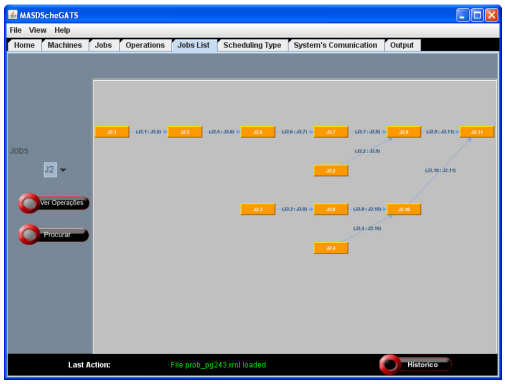

Figure 1 - Graph Operations of a Complex Job

The problem, focused in our work, which we call *Extended Job-Shop Scheduling Problem* (EJSSP) [1],[2], has major extensions and differences in relation to the classic Job-Shop Scheduling Problem (JSSP). In this work, we define a job as a manufacturing order for a final item, that could be *Simple* or *Complex* (Figure 1). It may be Simple, like a part, requiring a set of operations to be processed. We define it as Simple Product or Simple Final Item. Complex Final Items, requiring processing of several operations on a number of parts followed by assembly operations at several stages, are also dealt with.

3. MASDScheGATS System

Distributed environment approaches are important in order to improve scheduling systems flexibility and capacity to react to unpredictable events. It is accepted that new generations of manufacturing facilities, with increasing specialization and integration, add more problematic challenges to scheduling systems. For that reason, issues like robustness, regeneration capacities

and efficiency are currently critical elements in the design of manufacturing scheduling system and encouraged the development of new architectures and solutions, leveraging the MAS research results.

It starts focusing on the solution of the dynamic deterministic EJSSP problems. For solving these we developed a framework, leading to a dynamic scheduling system (Figure 2) having as a fundamental scheduling tool, a hybrid scheduling system, with two main pieces of intelligence.

One such piece is a Hybrid Scheduling Module that is a combination of Tabu Search and Genetic Algorithm based method and a mechanism for inter-machine activity coordination. The objective of this mechanism is to coordinate the operation of machines, taking into account the technological constraints of jobs, i.e. job operations precedence relationships, towards obtaining good schedules. The other piece is a dynamic adaptation module that includes mechanisms for neighbourhood/population regeneration under dynamic environments, increasing or decreasing it according new job arrivals or cancellations.

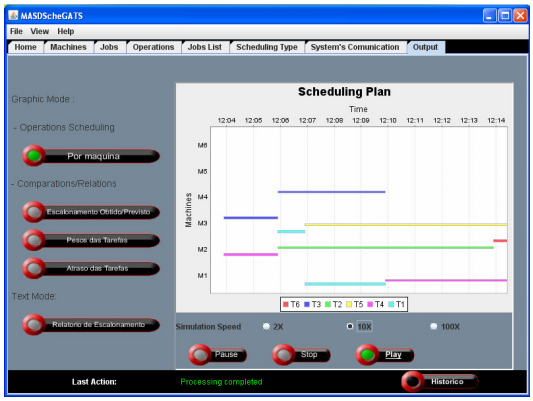

Figure 2 - MASDScheGATS Prototype System

3.1. Hybrid Scheduling Module

Initially, we start by decomposing the deterministic EJSSP problem into a series of deterministic Single Machine Scheduling Problems (SMSP). We assume the existence of different and known job release times rj, prior to which no processing of the job can be done and, also, job due dates dj. Based on these, release dates and due dates are determined for each SMSP and, subsequently, each such problem is solved independently by a TS or a GA(considering a self-parameterization issue). Afterwards, the solutions obtained for each SMSP are integrated to obtain a solution to the main EJSSP problem instance.

The integration of the SMSP solutions may give an unfeasible schedule to the EJSSP. This is why schedule repairing may be necessary to obtain a feasible solution.

3.2 Dynamic Adaptation Module

For non-deterministic problems some or all parameters are uncertain, i.e. are not fixed as we assumed in the deterministic problem. Non-determinism of variables has to be taken into account in real world problems. For generating acceptable solutions in such circumstances our approach starts by generating a predictive schedule, using the available information and then, if perturbations occur in the system during execution, the schedule may have to be modified or revised accordingly, i.e. rescheduling is performed. Therefore, in this process, an important decision must be taken, namely that of deciding if and when rescheduling should happen. The decision strategies for rescheduling may be grouped into three categories: continuous, periodic and hybrid rescheduling. In the continuous one rescheduling is done whenever an event modifying the state of the system occurs. In periodic rescheduling, the current schedule is modified at regular time intervals, taking into account the schedule perturbations that have occurred. Finally, for the hybrid rescheduling the current schedule is modified at regular time intervals if some perturbation occurs.

In the scheduling system for EJSSP, rescheduling is necessary due to two classes of events:

- Partial events which imply variability in jobs or operations attributes such as processing times, due dates and release times.
- Total events which imply variability in neighbourhood structure, resulting from either new job arrivals or job cancellations.

While, on one hand, partial events only require redefining job attributes and re-evaluation of the objective function of solutions, total events, on the other hand, 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. In this work, this is carried out by mechanisms described in [1] for SMSP.

Considering the processing times involved and the high frequency of perturbations, rescheduling all jobs from the beginning should be avoided. However, if

work has not yet started and time is available, then an obvious and simple approach to rescheduling would be to restart the scheduling from scratch with a new modified solution on which takes into account the perturbation, for example a new job arrival. When there is not enough time to reschedule from scratch or job processing has already started, a strategy must be used which adapts the current schedule having in consideration the kind of perturbation occurred.

The occurrence of a partial event requires redefinition of job attributes and a re-evaluation of the schedule objective function. A change in job due date requires the re-calculation of the operation starting and completion due times of all respective operations. However, changes in the operation processing times only requires re-calculation of the operation starting and completion due times of the succeeding operations. A new job arrival requires definition of the correspondent operation starting and completion times and a regenerating mechanism to integrate all operations on the respective single machine problems. In the presence of a job cancellation, the application of a regenerating mechanism eliminates the job operations from the SMSP where they appear. After the insertion or deletion of positions, neighbourhood regeneration is done by updating the size of the neighbourhood and ensuring a structure identical to the existing one. Then the scheduling module can apply the search process for better solutions with the new modified solution.

3.3 Hybrid Architecture

The work described in this paper is a system where a community of distributed, autonomous, cooperating and asynchronously communicating machines tries to solve scheduling problems.

The main purpose of MASDScheGATS (Multi-Agent System for Distributed Manufacturing Scheduling with Genetic Algorithms and Tabu Search) is to create a Multi-Agent system where each agent represents a resource (Machine Agents) in a Manufacturing System.

Each Machine Agent must be able: to find an optimal or near optimal local solution trough Genetic Algorithms or Tabu Search meta-heuristics; 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.

The original Scheduling problem defined in section 2, is decomposed into a series of Single Machine Scheduling Problems (SMSP)[1]. The Machine Agents (which has a Meta-Heuristic associated) obtain local solutions and later cooperate in order to overcome inter-agent constraints and achieve a global schedule.

The proposed Team-Work architecture is based on three different types of agents. In order to allow a seamless communication with the user, a User Interface Agent is implemented. This agent, apart from being responsible for the user interface, will generate the necessary Task Agents dynamically according to the number of tasks that comprise the scheduling problem and assign each task to the respective Task Agent.

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.

Finally, 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.

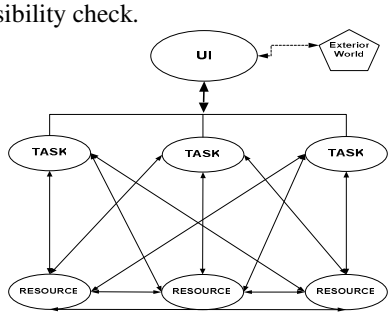

Figure 3 - MASDScheGATS System Architecture

The architecture was implemented using the Java Agent Development framework (JADE). The main challenge of the implementation was the message propagation and the synchronization of the agents as they advance to the next round. This situation occurs because it is impossible to control the delay of messages between the agents and the order of the messages in the message queue. To ensure that agents do not move towards the next round before handling every operation schedule (and potential conflicts between operations), each agent waits for a message (with a flag indicating the round number) from all agents of a different type (Task Agents and Machine Agents types) with which it has some relation. As such,

each Task Agent will wait for a message of all Machine Agents that handle operations belonging to the job managed by the Task Agent and each Machine Agent will wait for a message from each Task Agent, to which the operations to be processed in the machine managed by the Machine Agent belong.

Also, within agents of the same type was necessary to implement some synchronization aspects. Task Agents need to communicate to check whether they are in cycle or not and Machine Agents need to communicate to ensure that they do not attempt to resolve conflicts when an agent with a lower priority value has conflicts with operations that are processed in Machine Agents with higher priority values.

It starts focusing on the solution of the dynamic deterministic EJSSP problems. For solving these we developed a framework, leading to a dynamic scheduling system having as a fundamental scheduling tool, a hybrid scheduling system, with two main pieces of intelligence (Figure 1).

One such piece is a combination of TS and GA based method and a mechanism for inter-machine activity coordination. The objective of this mechanism is to coordinate the operation of machines, taking into account the technological constraints of jobs, i.e. job operations precedence relationships, towards obtaining good schedules. The other piece is a dynamic adaptation module that includes mechanisms for neighbourhood/population regeneration under dynamic environments, increasing or decreasing it according new job arrivals or cancellations.

3.4 Cooperation Mechanism

Once the Machine Agents find their respective best local solution to the set of assigned operations, it is likely that the assembly of such solutions in a final plan will not establish a feasible schedule. The reason for this situation derives from the fact that each Machine Agent does not take into account, due to the concurrent procedure of local searching, the plans of other agents with which it has inter-agent constraints. It is therefore necessary a subsequent coordination mechanism so that a global feasible schedule is attained whilst minimizing the adjustments to the initial local solutions [2].

The implemented mechanism gets its inspiration from the Asynchronous Weak-Commitment Search Algorithm [7]. The cornerstone of the mechanism is the assignment of priority values to Machine Agents, according to an altruistic stance, so that lower priority agents will satisfy the constraints of higher priority agents. A set of coordination messages are broadcasted amongst the agents, within each coordination round, in order to ensure a coherent communication of conflicts and avoid unnecessary processing of solutions that will be discarded in succeeding steps.

There are two events on which agents can increase their priority values. The first one occurs when a Machine Agent cannot find a satisfactory solution. In this event, the system will increase that machine's priority value so that other Machine Agents will attempt to change their schedules in order to find a solution to the conflicting constraints. The second event is the discovery of a solution that was not found previously, i.e., if a Machine Agent changes its schedule to a state that was not reached before, the system increases the priority of this agent. The increase of priority in such an event is necessary firstly as a way to further explore the new state and secondly as a way to provide the algorithm the necessary completeness. Typically, the increase of the priority value in the case of the second event is much greater then in the first event.

Although the mechanism demonstrates to be a remarkable approach in the resolution of distributed constraint satisfaction problems, it is imperative to take notice of potential loops that may arise when operations in conflict have a narrow solution space, hindering the possibility of a wider diversification of the solutions. It so happens because Machine Agents can only resolve conflicts by swapping operations or filling empty time slots in the time window, assuring the non violation of the earliest due dates and conflict resolution. One way to avoid this situation is to schedule the operations in conflict towards a later time. While feasible, this option is not ideal as deteriorates the quality of the global schedule. Another way is the introduction of a disturbance onto the schedule when a cycle is detected. This is done through the relaxation of the conflict resolution constraint. If an agent cannot solve the conflict, it may at least try to position the operation at a time closer to the required time of resolution. Even if the conflict is not solved, a different schedule is generated and other agents can try to solve the conflict with the new scheduling state.

Also essential to mention is the tabu-list implemented in the mechanism. This tabu-list holds the duple (op,round#) that is employed to ensure persistence of the solutions found so far. In fact, operations in the tabu-list cannot be moved in the current round, and for subsequent n rounds, introducing the parameter n, which values enclose a significant influence in the mechanism's performance.

Important to mention as well is the way convergence to optimal solutions is achieved. Via our

particular design of the mechanism, every schedule change is performed taking into consideration the earliest due date. If the required time for solving the conflict falls within a certain time interval, the mechanism will attempt to schedule the operation at the closest possible time of the earliest due date.

Finally, two mechanisms were implemented to control the running time of the algorithm. Given the fact that a complete algorithm is at hand, it is imperative to control its running time, that, while solving problems of a substantial dimension, may take several minutes to reach a solution.

The first mechanism is the already mentioned cycle detector. Basically, this mechanism is no more than the Task Agents' memory, which consists of all the different allocations that the Machines have found for the particular operations of the task managed by the agent. The memory can be kept for the all duration of the resolution or for some rounds only.

The second mechanism is the task locking mechanism. The purpose of this mechanism is to maintain stationary the plan of the operations of a task as soon as there are no more conflicts within its operations. Once the operations are fixed, the Machine Agents will try to adapt the allocation of operations of other tasks to the time slots still available. For instance, if at a certain moment in the resolution two tasks are locked, and if each task has an operation on every machine, this means that two operations on each Machine Agent are fixed and these Agents must now find feasible plans for the other operations without generating conflicts with the operations already locked. The mechanism is adjustable by parameterization. It is possible to keep the task locked until a global solution is found. As an alternative, the task can be unlocked after a certain number of rounds or if the time slots allocated to its operations are in high demand in other Machine Agents.

4. Conclusions and Future Work

This paper presents a prototype of an Hybrid Scheduling System that assumes the combination of different Meta-Heuristics and Multi-Agent Systems. To solve the scheduling problem, Machine Agents and Task Agents must interact and cooperate with other agents in order to obtain optimal or near-optimal global performances trough Meta-heuristics. The idea is that from local, autonomous and often conflicting agent's objectives, a global solution emerges from a community of machine agents solving locally their schedules while cooperating with other machine agents. Agents have to manage their internal behaviors and their relationships with other agents via cooperative negotiation in accordance with business policies defined by the user manager.

We believe that a new contribution for the resolution of more realistic scheduling problems (Extended Job Shop Problems) was described in this paper. The particularity of our approach is the procedure to schedule operations, as each machine will first find local optimal or near optimal solutions, succeeded by the interaction with other machines trough cooperation mechanism as a way to find an optimal or near-optimal global schedule.

5. Acknowledgment

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

6. References

[1] Ana M. Madureira, Meta-Heuristics Application to Scheduling in Dynamic Environments of Discrete Manufacturing. PhD Dissertation. University of Minho, Braga, Portugal, 2003 (in portuguese).

[2] Ana Madureira, Nuno Gomes and Joaquim Santos, Cooperative Negotiation Mechanism for Agent Based Distributed Manufacturing Scheduling. WSEAS Transactions on Systems, Issue 12, Volume 5, ISSN:1109-2777, 2899-2904 2006.

[3] Brian Horling and Victor Lesser, A Survey of Multi-Agent Organizational Paradigms, University of Massachusets, 2005.

[4] H. Aytug, M.A. Lawley, K. McKay,S. Mohan and R. Uzsoy, Executing production schedules in the face of uncertainties: A review and some future directions, European Journal of Operational Research,Volume 16 (1), 2005, pp. 86-110.

[5] M. C. Portmann, Scheduling Methodology: optimization and compu-search approaches, in the planning and scheduling of production sys-tems, Chapman &Hall, 1997.

[6] M. Wooldridge, An Introduction to Multiagent Systems, John Wiley and Sons, 2002

[7] M. Yokoo and K.Hirayama, Algorithms for Distributed Constraint Satisfaction: A Review, Journal of Autonomous Agents and Multi-Agent Systems, 2000.

[8] P. Cowling, & M. Johansson, Real time information for effective dynamic scheduling. European J. of Operat. Research,139 (2), 2002, 230-244.

[9] S. Jain and S. Meeran, Deterministic Job Shop scheduling: past, present and future, European Journal of Operational Research, nº113, 390-434, 1999.

[10] W. Shen, and D. Norrie, Agent-based systems for intelligent manufacturing: a state of the art survey, Int. J. Knowl. Inform. Syst., vol. 1, no. 2, 1999, pp. 129– 156.