Team-Work based Architecture for Distributed Manufacturing Scheduling

ANA MADUREIRA[†] NUNO GOMES[*] JOAQUIM SANTOS [†]

Computer Science Department
Institute of Engineering - Polytechnic of Porto
GECAD – Knowledge Engineering and Decision Support Research Group
Porto, Portugal

Abstract: - This paper presents a Team-Work based architecture for Distributed Manufacturing Scheduling with Genetic Algorithms and Tabu Search. We consider that a good global solution for a scheduling problem may emerge from a community of machine agents solving locally their schedules and cooperating with other machine agents. Social aspects are considered when a community of autonomous agents cooperate to reach a common goal. Agents negotiate in a cooperative way, in order to find a consistent overall plan, while avoiding significant changes onto their current best possible local plans. A cooperative negotiation mechanism is proposed.

Key-Words: - Multi-Agent Systems, Dynamic and Distributed Scheduling, Meta-Heuristics, Manufacturing

1. Introduction

Real world manufacturing scheduling systems are related to complex systems operated in continuous changing environments. Such environments are frequently subject to several kinds of random occurrences and perturbations, such as new job arrivals, machine breakdowns, employee's sickness, jobs cancellation, due dates and time processing changes causing established schedules to become easily outdated and unsuitable. Scheduling under this environment is known as dynamic.

Traditional scheduling methods, encounter great difficulties when they are applied to some real-world situations. Several attempts have been made to modify algorithms, to tune them for optimization in a changing environment. The interest in optimization algorithms for dynamic optimization problems is growing and a number of authors have proposed an even greater number of new approaches, the field lacks a general understanding as to suitable benchmark problems, fair comparisons and measurement of algorithm quality [1][3][4][13].

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 main purpose of this work is the resolution of more realistic scheduling problems in the domain of manufacturing environments, known as Extended Job-Shop Scheduling Problems [13-14], combining Multi-Agent Systems (MAS) and Meta-Heuristics potentialities.

The proposed Team-based approach is rather different from the ones found in the literature; as we try to implement a system where each agent (Machine Agent) is responsible for optimize the scheduling of operations for one machine through Tabu Search or Genetic Algorithms. After local solutions are found, each Machine Agent is required to cooperate with other Machine Agents in order to find a global optimal schedule. A Cooperative Negotiation mechanism is proposed for coordinate this process.

The remaining sections are organized as follows: Section 2 summarizes some related work and the research on multi-agent technology for dynamic scheduling resolution. In section 3, some organizational issues such as negotiation, cooperation and MAS architectures are described. In section 4 the scheduling problem under consideration is defined. Section 5 presents the Team-Work based Model for Dynamic Manufacturing Scheduling and describes the proposed coordination mechanism. Finally, the paper presents some conclusions and puts forward some ideas for future work.

2. Related Work

Scheduling problems arise in a diverse set of domains, ranging from manufacturing to hospitals settings, transports, computer and space environments, amongst others. Most of these domains are characterized by a great amount of uncertainty that leads to significant system dynamism. The problem of dynamic scheduling is one that is receiving increasing attention amongst both researchers and practitioners. In spite of all previous contributions the scheduling problem still known to be NP-complete [3]. This fact incites researchers to explore new directions. Multi-Agent technology has been considered as an important approach for developing industrial distributed systems.

In [19] Shen and Norrie presented a state-of-the-art survey referencing a number of publications that attempted to solve distributed dynamic scheduling problems. According to these authors, there are two distinct approaches in the mentioned work.

The first is based on an incremental search process that may involve backtracking. In this approach, orders are assigned to agents that search for solutions to their local problems. If a solution holds violations of inter-agent constraints, with regard to other agents' solutions, agents backtrack and decide for a different path. Agents repeat this process until a feasible solution is found.

The second approach is based on systems in which an agent represents a single resource and is therefore responsible for scheduling that resource. Agents then negotiate with other agents in order to accomplish a feasible solution. Typically, this latter approach builds upon a constructive heuristic, where agents start scheduling operation per operation, in succession, until all operations are scheduled.

A different approach is presented by Logie et al. [12]. Here, a sliding window frame is implemented and all agents with processes inside the current window frame schedule their operations ignoring any operations outside that window. This process goes on until either the sliding window has advanced or gaps have opened between tasks inside the window frame.

Any other approaches to solve dynamic scheduling problems with multi-agents systems are somehow, and to our knowledge, variations of those presented above. For the interested reader, a quite extensive compilation of work in this domain is available at [6]. For further works developed on MAS for dynamic scheduling, see for example, [4][11][13][14][15][18][21].

As it will be shown later in this paper, our approach will introduce a rather different way of undertaking these problems.

3. Multi-Agent Systems

A Multi-Agent System (MAS) can be defined as "*a system composed by population of autonomous agents, which cooperate with each other to reach common objectives, while simultaneously each agent pursues individual objectives*" [7]. According to Russell and Norving [17] multi-agent systems "[...] *solve complex problems in a distributed fashion without the need for each agent to know about the whole problem being solved*". Both these definitions entail the idea that each agent has its own individual goals and therefore coordination concerns necessarily arise when the purpose of the system is the resolution of a global problem. However, effective coordination of multiple agents interacting in dynamic environments is a problem on its own and several strategies have been put forward to handle such challenges. More specifically, expressions like negotiation, coordination and cooperation have been employed to describe mechanisms that allow the management of multi-agent systems.

3.1 Multi-Agent Negotiation

Negotiation can be defined as the process in witch at least two operators, a sender and a receiver, communicate across a communication protocol in order to accomplish an agreement. A well known negotiation protocol is the contract net protocol. Since in multi-agent systems' context negotiation will always be specified by a protocol, which can be simple or sophisticated, deterministic or non-deterministic. Any type of negotiation can be seen as similar in its nature from the contract net protocol [4].

There are other negotiation models like game theory, economical or psycho-sociologic. However, most of the systems that employ negotiation use economical models like auctions where intervenient expectations can be identified. It is then possible to describe auctions by the strategy that agents use to develop the negotiation method.

MAS consisting of several autonomous entities, called agents, that interact with each other to either further their own interests (competition) or in pursuit a common objective (cooperation).

Negotiation research in multi-agent systems can be categorized into two main categories [10]: Competitive Negotiation which occurs among self-interested agents, each trying to maximize its local utility; while in Cooperative Negotiation agents try to reach the maximum global utility that takes into account the worth of all their activities. For further works developed on MAS negotiation, see for example [2][5[16].

3.2 MAS Architectures

In this section we present a brief description of some Multi-Agent Architectures (MAA), related on literature, for building distributed software systems. Multi-Agent Architectures (MAA) are fundamental for MAS development, as they establish a significant outcome on the system performance. Horling and Lesser [8] identified a range of architectural strategies that sprang from agent systems, their main advantages and disadvantages.

Paradigm	**Characteristics**	**Benefits**	**Drawbacks**
Hierarchy	Decomposition	Maps to many common domains; handles scale well	Potentially brittle; can lead to bottlenecks or delays
Holarchy	Decomposition with autonomy	Exploit autonomy of functional units	Must organize holons; lack of predictable performance
Coalition	Dynamic, goal-directed	Exploit strength in numbers	Short term benefits may not outweigh organization construction costs
Team	Group level cohesion	Address larger grained problems; task-centric	Increased communication
Congregation	Long-lived, utility-directed	Facilitates agent discovery	Sets may be overly restrictive
Society	Open system	Public services; well defined conventions	Potentially complex, agents may require additional society-related capabilities
Federation	Middle-agents	Matchmaking, brokering, translation services; facilitates dynamic agent pool	Intermediaries become bottlenecks
Market	Competition through pricing	Good at allocation; increased utility through centralization; increased fairness through bidding	Potential for collusion, malicious behavior; allocation decision complexity can be high
Matrix	Multiple managers	Resource sharing; multiply-influenced agents	Potential for conflicts; need for increased agent sophistication
Compound	Concurrent organizations	Exploit benefits of several organizational styles	Increased sophistication; drawbacks of several organizational styles

Table 1 - MAA Description [8]

Table 1 summarizes the characteristics, objectives, benefits and drawbacks of some MAA.

From the architectural strategies identified, we considered only the distributed because they would be appropriate to the objectives of our work considering the distributed nature of dynamic scheduling.

The explanation for this decision derives from the disadvantages that hierarchical architectures comprise. In fact, hierarchical architectures may lead to fragile systems as the concentration of control on agents of higher level can disrupt the entire system if these same agents fail. Also, such decision may lead to bottleneck effects, as agents from lower levels need to communicate with agents from higher levels for coordination and control decisions. Thus, and to avoid that kind of problems, the architectures initially considered for our work, namely the Market based architectures and Team based architectures, strongly reduce such disadvantages.

On the other hand, the drawbacks of the team and market based architectures can be reduced by the progress of technology, specifically network communication and security technology improvement.

Finally, we decided for Team based architecture due to its philosophy of cooperation. Agents agree to work together in order to solve a problem that is shared by all agents in the team. Such approach allows for the resolution of large-scale problems that a single agent would not be able to solve. Moreover, Team - based architecture has the ability to meet global constraints given the capability that agents possess to act in concert. As we shall see later, this characteristic is critical for the problem at hand.

4. Problem Definition

Most real-world multi-operation scheduling problems can be described as dynamic and extended versions of the classic or basic Job-Shop scheduling combinatorial optimization problem. The general Job-Shop Scheduling Problem (JSSP) can be generally described as a decision-making process on the allocation of a limited set of resources over time to perform a set of tasks or jobs. Most real-world multi-operation scheduling problems can be depicted as dynamic as already described before.

In this work we consider several extensions and additional constraints to the classic JSSP, namely: the existence of different job release dates; the existence of different job due dates; the possibility of job priorities; machines that can process more than one operation in the same job (recirculation); the existence of alternative machines; precedence constraints among operations of different jobs (as quite often, mainly in discrete manufacturing, products are made of several components that can be seen as different jobs whose manufacture must be coordinated); the existence of operations of the same job, on different parts and components, processed simultaneously on different machines, followed by components assembly operations (which characterizes the Extended Job-Shop Scheduling Problem (EJSSP)[13][14]).

5. Multi-Agent System for Distributed Manufacturing Scheduling with Genetic Algorithms and Tabu Search

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.

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 is able to find an optimal or near optimal local solution trough Genetic Algorithms or Tabu Search meta-heuristics, to change/adapt the parameters of the basic algorithm according to the current situation or even to switch from one algorithm to another.

The original Scheduling problem defined in section 4, is decomposed into a series of Single Machine Scheduling Problems (SMSP)[13-14]. The Machine Agents obtain local solutions and later cooperate in order to overcome inter-agent constraints and achieve a global schedule.

5.1 MASDScheGATS Architecture

The proposed architecture, to handle the problem, 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 regarding the task. 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 task 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 1).

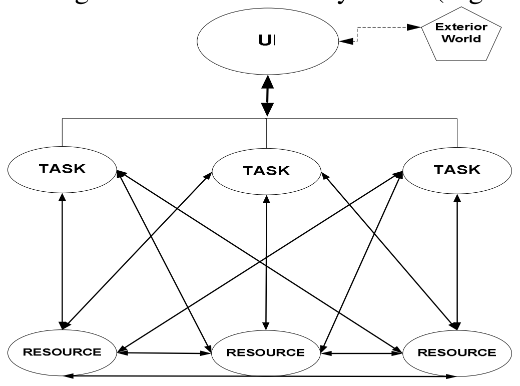

Figure 1- MASDScheGATS System Architecture

5.2 Negotiation Mechanisms

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.

The mechanism to be implemented gets its inspiration from the Asynchronous Weak-Commitment Search Algorithm [20]. 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. When a Machine Agent can not find a satisfactory solution, 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 major procedures of the coordination mechanism algorithm are described, in sequential order, below in Figure 2:

```
Step 1: User Interface Agent
for (each machine agent)
  Assign initial priority values according to their
  alphanumeric name.
end for;

Step 2: Task Agent
for (each task)
  Assign each operation to the respective Machine Agent
end for;

Step 3: Machine Agent
When received (set of operations, criteria) do
  Perform meta-heuristics in order to optimize criteria.
  Communicate sequence position of each operation to
  respective Task Agent.
End do

Step 4: Task Agent
When received (set of operations with sequence position)
do
  Verify feasibility of operations' sequence
  if (sequence is feasible) then
    Communicate feasibility to Machine Agents through
    "allgood" messages
  else
    Identify conflicts and decide, based upon priority
    values, which Machine Agent should attempt to solve
    the conflict.
    Communicate decision to Machine Agents through
    "nogood" messages.
  End if else
End do

Step 5: Machine Agent
When received (all "nogood" and "allgood" messages) do
  Verify if there is a conflict to be solved on the
  machine
  if (no conflict needs to be solved) then
    communicate old sequence to each Task Agent
  else
    Verify if this same conflict was (within n rounds)
    already communicated
    if (conflict was communicated) then
      force solution.
      Return

    if (inter-agent constraints exist with lower priority
    agent that has also to solve conflict) then
      Halt processing of conflict.
      Communicate old sequence to each Task Agent.
    else attempt to solve conflict via local search.
      if (conflict solved) then
        Communicate new sequence to each task agent
      else
        Increase priority of agent to max-priority+1.
        Communicate old sequence to each Task Agent.
      End if else
    End if else
  End if else
end of round
return to step (4)
End do
```

Figure 2 - Coordination Mechanism Algorithm

5.4 Multi-Agent platform – JADE

In order to justify our decision of use JADE to develop our proposal, we will describe in a summarized way some of its most important capabilities.

The main objective of JADE (Java Agent DEvelopment Framework) is to turn simple the development of agent systems ensuring at the same time compliance with the most well known standards of this specific area. To guarantee this, JADE was developed in full compliance with FIPA specifications, implementing features like naming service and yellow-page service, message transport and parsing service, and a library of FIPA interaction protocols ready to be used.

This agent framework can be spread for several hosts in which one Java application and one Java Virtual Machine are running simultaneously. Each Java Virtual machine functions like a container of autonomous agents that provides a complete concurrently execution environment.

The communication architecture offers flexible and efficient messaging, where JADE creates and manages a queue of incoming ACL messages, private to each agent; agents can access their queue via a combination of several methods: blocking, polling, timeout and pattern matching based [9].

6. Concluding Remarks and Future Work

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 mechanisms as a way to find a optimal or near-optimal global schedule.

We have discussed several options for coordination mechanisms and justified our selection for cooperation. A description of our cooperation mechanism is presented. Furthermore, we have also discussed two main approaches for MAS architectures and justified our option for a distributed architecture considering the drawbacks of hierarchical architectures with regards to system robustness and propensity for communication bottlenecks.

Work still to be done includes the testing of the proposed system and negotiation mechanisms under dynamic environments subject to several random perturbations. We realize, however, that this is not an easy task because it is difficult to find test problems and computational results for the considered dynamic environment where the jobs to be processed have release dates, due dates and different job assembly levels (parallel/concurrent operations).

Additionally, we envisage to develop a learning mechanism that supported by a knowledge base will permit to Machine Agents recognize scheduling patterns and therefore improve the overall efficiency of the system.

ACKNOWLEDGEMENTS

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

References:

[1] Aytug, H., Lawley, M.A., 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.

[2] Beer, Martin, d'Inverno, Mark, Luck, Michael, Jennings, Nick, Preist, Chris, and Schroeder, Michael (1998). Negotiation in Multi-Agent Systems, Panel discussion at the Workshop of the UK Special Interest Group on Multi-Agent Systems.

[3] Blazewicz, J., Ecker, K. H., Pesch, E., Smith, G. Weglarz, J.(2001). Scheduling Computer and Manufacturing processes. Springer. 2nd edition. New York.

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

[5] Doran , J. E., Franklin , S., Jennings ,N. R., Norman , T. J. (1997). On Cooperation in Multi-Agent Systems, Knowl. Eng. Rev., 12(3):309—314.

[6] FARMS LAB - Laboratory for Fundamental and Applied Research in Multi-agent Systems, Multi-Agent Scheduling in Manufacturing Systems: http://farm.ecs.umass.edu/~pschiegg/bib/lit.html

[7] Ferber, J. (1995). Les Sístemes multi-agents: versune intelligence collective. Interedition.

[8] Horling, Brian, Lesser, Victor (2005). A Survey of Multi-Agent Organizational Paradigms, University of Massachusets.

[9] JADE – Java Agent DEvelopment Environment. Technical Description. Retrieved on July 26, 2006 :http:// jade.tilab.com/description-technical.htm

[10] Jennings, N. R. (1996). Coordination Techniques for Distributed Artificial Intelligence, in Foundations of Distributed Artificial Intelligence(eds. G. M. P. O'Hare and N. R. Jennings), Wiley, 1996, 187-210.

[11] Lind , Jurgen (1999). A Process Model for the Design of Multi-Agent Systems, Research Report TM- 99-03, German Research Center for AI (DFKI).

[12] Logie, S., Sabaz, D., Gruver, W.A. (2004). Sliding Window Distributed Combinatorial Scheduling unsing JADE, IEEE International Conference on Systems, Man and Cybernetics.

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

[14] Madureira, Ana, Ramos, Carlos & Silva, Sílvio (2004). Toward Dynamic Scheduling Through Evolutionary Computing. WSEAS Transactions on Systems. Issue 4. Volume 3. 1596-1604.

[15] Milano, M. and Roli, A. (2004). MAGMA: a Multiagent Architecture for Metaheuristics, IEEE Transaction on Systems Man and Cybernetics, Part B, Vol 34 N. 2.

[16] Monett-Díaz, Dagmar (2004). +CARPS: Configuration of Metaheuristics based on Cooperative Agents, Proceedings of First International Workshop on Hibrid Metaheuristics (HM 2004).

[17] Russel, S. and Norvig, P. (2003). Artificial Intelligence: A Modern Approach, Prentice Hall/Pearson Education International: Englewood Cliffs (NJ), (2nd Ed).

[18] Shehory, O. and Sturm, A. (2001), Evaluation of modeling techniques for agent-based systems, in Proceedings of the 5th International Conference on Autonomous Agents, ACM Press: Montreal (CA).

[19] Shen, W. and Norrie, D. (1999). Agent-based systems for intelligent manufacturing: a state of the art survey, Int. J. Knowl. Inform. Syst., vol. 1, no. 2, pp. 129– 156. [20] Wooldridge, M. (2002). An Introduction to Multiagent Systems, John Wiley and Sons.

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

[21] Zambonelli, F. and Parunak, H. V. D. (2004). Toward a change of paradigm in computer science and software engineering: A synthesis, Knowl. Eng. Rev.,