Neurocomputing 132 (2014) 97–110

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Negotiation mechanism for self-organized scheduling system with collective intelligence

A. Madureira [a,*], I. Pereira [a], P. Pereira [a], A. Abraham [b]

[a] GECAD – Knowledge Engineering and Decision Support Research Center, School of Engineering, Polytechnic of Porto (ISEP/IPP), Portugal
[b] Machine Intelligence Research Labs (MIR Labs), USA

ARTICLE INFO

Article history:
Received 31 October 2012
Received in revised form
2 September 2013
Accepted 23 October 2013
Available online 27 November 2013

Keywords:
Negotiation in MAS
Self-organization
Swarm intelligence
Dynamic scheduling
Agile manufacturing

ABSTRACT

Current Manufacturing Systems challenges due to international economic crisis, market globalization and e-business trends, incites the development of intelligent systems to support decision making, which allows managers to concentrate on high-level tasks management while improving decision response and effectiveness towards manufacturing agility.

This paper presents a novel negotiation mechanism for dynamic scheduling based on social and collective intelligence. Under the proposed negotiation mechanism, agents must interact and collaborate in order to improve the global schedule. Swarm Intelligence (SI) is considered a general aggregation term for several computational techniques, which use ideas and inspiration from the social behaviors of insects and other biological systems. This work is primarily concerned with negotiation, where multiple self-interested agents can reach agreement over the exchange of operations on competitive resources. Experimental analysis was performed in order to validate the influence of negotiation mechanism in the system performance and the SI technique. Empirical results and statistical evidence illustrate that the negotiation mechanism influence significantly the overall system performance and the effectiveness of Artificial Bee Colony for *makespan* minimization and on the machine occupation maximization.

1. Introduction

For today's manufacturing environments, it is increasingly necessary that a close relationship between manufacturing decision making and corporate business strategy exists, so that manufacturing decisions complement and are fully aligned with the strategic objectives of organizations through agility concerns and requirements. Agility refers to the manufacturing systems ability to efficiently adapt to market and environmental changes in an cost-effective ways.

Real world scheduling requirements are related with complex systems operated in dynamic environments frequently subject to several kinds of imponderables and perturbations, such as:

- Scheduled orders could take more time than estimated;
- Machines could become unavailable or additional ones may be introduced;
- New orders arrive continuously to the system while scheduled orders could be cancelled;
- Unexpected events occur in the system (employees sickness, rush orders, lateness on raw-materials or components)

These scenarios make the current schedules easily outdated and unsuitable. Scheduling under this environment is known as dynamic, which could be defined as a continuous and ongoing reactive process where the real time information implies the revision and dynamic adaptation of current schedules to the perturbations [1,3].

A Job-Shop like manufacturing system has associated a dynamic nature observed through several kinds of perturbations on working conditions and requirements over time. For this kind of environment, it is important that the ability to efficiently and effectively adapt, on a continuous basis, existing schedules according to the referred disturbances, are mandatory for keeping business performance levels. The application of optimization techniques to the resolution of this class of real world scheduling problems seems really promising. Although, most of the known work on scheduling deals with optimization of classical Job Shop Scheduling Problems (JSSP) problems, on static and dynamic environments [1,2].

The problem of finding good solutions is very important to real manufacturing systems considering that production rate and production costs are very dependent on the schedules used for controlling the flow of work through the system. Production planning and distribution, transport planning, allocation of resources (raw materials, manpower or machines in time) and

* Correspondence to: GECAD, Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal. Tel.: +351 22 8340500; fax: +351 22 8321159.
E-mail address: amd@isep.ipp.pt (A. Madureira).
URL: http://www.gecad.isep.ipp.pt/ (A. Madureira).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.10.032

task scheduling are combinatorial optimization problems common in industrial reality. It is not possible to always adopt the optimal solution for two reasons: due to its complex nature, the resolution to optimality in an acceptable time for making decisions is normally intractable, and many problems in reality are so dynamic that when we process/execute the solution, the characteristics of the problem have already changed, and this is not the optimal solution for the new problem. Such dynamic scheduling has receiving increasing attention amongst researchers and practitioners [3–6]. However, scheduling is still having difficulties in real world environments and, hence, human intervention is required to maintain real-time adaptation and optimization.

The interest and research on Decision Support Systems (DSS) that exhibit self-organization properties is increasingly drawing to formalize some of the ideas from Autonomic Computing [7,8] for handling problems in complex manufacturing systems and to identify mechanisms that makes use of autonomous entities in solving hard computational problems and in modelling complex systems through Self-organized or Self-managed behaviours. Self-managed systems have the ability to manage themselves and to dynamically adapt to change in accordance with evolving or dynamic business policies and objectives, allowing the addition and removal of resources/tasks without service disruption [8]. This field of research has received much attention in Autonomic Computing (AC) paradigm [7]. As a result, managers and professionals can focus on tasks with higher value to the business process. Agent based Computing technology is well adapted to model and solve production planning problems in manufacturing systems and can easily integrate social issues and self-organized mechanisms into multi-agent architectures.

Nature provides several and diverse examples of social systems and collective intelligence, such as: insect colonies foraging behaviour for food; bacteria which appear able to act in a finalized way; the human brain considering that intelligence and mind arises from the interaction and coordination of neurons; the molecule and cell formation considering homeostasis and the capability of adapting and reproducing arise from protein interactions and antibody detection. Several efforts and contributions have been related on literature that take collective intelligence as an inspiration and basis for optimization algorithms developing based on analogy with social and self-organized behaviour [4,6,11,10,11]. These approaches have been generally referred as Swarm Intelligence (SI), and are based on assumption that an organized behaviour emerges from the interactions of many simple agents like observed in nature [9,10].

To address DSS for dynamic scheduling with self-organized capabilities, we intend to integrate and explore the following paradigms: Multi-Agent Systems (MAS) [12–14], Coordination and Competition [15,16], Autonomic Computing [7,8] and Swarm Intelligence [9,10].

In this research, we propose a novel negotiation mechanism, to the resolution of scheduling in real manufacturing systems, which is by nature intrinsically a Complex Adaptive System, through negotiation. *Complex* in the sense that manufacturing systems are composed of many components (jobs, operations, machines). *Adaptive* when referring to the fact that the system must dynamically adapt to external perturbations, like rush orders, or lateness on raw-materials, and *System* considering that all components are interconnected and interdependent. A negotiation mechanism is proposed considering the following assumptions: A set of autonomous resource agents, each implementing a SI method for Single Machine Scheduling Problems (SMSP) are engaged in finding the optimal or sub-optimal solution; A coordination mechanism combining the single solutions obtained by the resource agents into a global solution is performed; A negotiation mechanism to improve global solutions by machine idle times reducing could be established.

The remaining sections of this paper are organized as follows: in section 2 the scheduling problem definition is presented. Theoretical foundations, biological motivation and fundamental aspects of SI Paradigm namely with focalization on Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Artificial Bees Colony (ABC) algorithms are summarized in Section 3. Section 4 presents some related work on negotiation for scheduling through MAS. In Section 5, the competitive architecture for the self-organized dynamic scheduling is presented and in Section 6 it is described the proposed negotiation mechanism, which integrates the ideas from collective intelligence and negotiation in a Multi-Agent System. The computational study and discussion of results is presented on Section 7. Finally, the paper presents some conclusions and provides some ideas for future works.

2. Problem definition

Real world scheduling problems have received a lot of attention in recent years. In this work, we consider the resolution of more 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 [1,2]. 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 environments tend to be dynamic, i.e. new jobs arrive at unpredictable intervals, machines breakdown, jobs can be cancelled and due dates and processing times can frequently change.

In this work, solutions are encoded by the direct representation, where the schedule is described as a sequence of operations, i.e. each position represents an operation index with initial and final processing times. Each operation is characterized by the index (i, j, k, l), where i defines the machine where the operation k is processed, j the job it belongs to, and l the graph precedence operation level (level 1 (one) corresponds to initial operations, without precedents [3].

The minimization of total completion time, also known as makespan [1,2] is given by

$$Min\ C_{\max} = max(F_j), \quad \forall j = 1, \ldots, n,$$

Subject to

$$ST_{ijkl} + p_{ijkl} \le ST_{ij'k'l'} \quad \forall j = 1, \ldots, n, \quad \forall (O_{ijkl}, O_{ij'k'l'}) \tag{1}$$

The constraint from (1) represents the precedent relationship between two operations k and $k'(k \neq k'$ and $k < k'$ and $l < l')$ of the same job j, that could be executed on different machines k and k', and at different levels l and l'.

$$ST_{ijkl} \ge t_{ijkl+1} \quad \forall O_{ijkl} \tag{2}$$

The constraint shown in (2) represents that the processing time to start operation O_{ijkl} must be greater or equal to the earliest start time for the same operation. The constraint, specified on (3), represents machine occupation, where only one operation could