International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: http://www.tandfonline.com/loi/tprs20

Distributed scheduling: a review of concepts and applications

Ayşegül Toptal & Ihsan Sabuncuoglu

To cite this article: Ayşegül Toptal & Ihsan Sabuncuoglu (2010) Distributed scheduling: a review of concepts and applications, International Journal of Production Research, 48:18, 5235-5262, DOI: 10.1080/00207540903121065

To link to this article: http://dx.doi.org/10.1080/00207540903121065

International Journal of Production Research
Vol. 48, No. 18, 15 September 2010, 5235–5262

Taylor & Francis
Taylor & Francis Group

RESEARCH PAPER

Distributed scheduling: a review of concepts and applications

Ayşegül Toptal* and Ihsan Sabuncuoglu

Industrial Engineering Department, Bilkent University, Ankara, 06800, Turkey

(*Received 17 September 2008; final version received 29 May 2009*)

Distributed scheduling (DS) is an approach that enables local decision makers to create schedules that consider local objectives and constraints within the boundaries of the overall system objectives. Local decisions from different parts of the system are then integrated through coordination and communication mechanisms. Distributed scheduling attracts the interest of many researchers from a variety of disciplines, such as computer science, economics, manufacturing, and service operations management. One reason is that the problems faced in this area include issues ranging from information architectures, to negotiation mechanisms, to the design of scheduling algorithms. In this paper, we provide a survey and a critical analysis of the literature on distributed scheduling. While we propose a comprehensive taxonomy that accounts for many factors related to distributed scheduling, we also analyse the body of research in which the scheduling aspect is rigorously discussed. The focus of this paper is to review the studies that concern scheduling algorithms in a distributed architecture, not, for example, protocol languages or database architectures. The contribution of this paper is twofold: to unify the literature within our scope under a common terminology and to determine the critical design factors unique to distributed scheduling and in relation to centralised scheduling.

Keywords: distributed scheduling; heterarchical systems; hierarchical systems; agents; decomposition

1. Introduction

Scheduling deals with the allocation of scarce resources to tasks over time (Pinedo 2002, Leung 2004). It is a vital component of shop-floor control systems, in the sense that a well-prepared schedule can significantly improve the system performance in terms of utilisation, production lead times, and due-date-related measures. Because scheduling is frequently encountered in practice, it is extensively studied in industrial engineering and operations research literature. Numerous studies investigate a wide variety of scheduling problems. All these studies indicate that the majority of the scheduling problems are NP-hard. Moreover, the dynamic and stochastic nature of the scheduling environments makes it almost impossible to obtain optimum schedules in practice. Hence, various approximate methods (i.e. both constructive and iterative heuristics) are proposed to cope with real-life scheduling problems.

In production systems, scheduling is considered as an operational (short-term) activity that has to interface with other functions (e.g. aggregate production planning, capacity

*Corresponding author. Email: toptal@bilkent.edu.tr

ISSN 0020–7543 print/ISSN 1366–588X online

DOI: 10.1080/00207540903121065
http://www.informaworld.com

planning, material planning) of the planning process. This activity often prevails at two levels of detail among the various stages of the production planning and control; those are, master production scheduling and short-range scheduling. Master production scheduling is based on an aggregate production plan and concerns the decisions regarding the specific products to be produced in particular time periods. These periods are in length of days or weeks. The decisions made at this stage are used for material planning. In short-range scheduling, they are narrowed down to a finer time frame to suggest which jobs are to be run on each machine on the shop floor (Silver *et al.* 1998).

In order to be competitive in today's rapidly changing business world, organisations have shifted from a centralised to a more decentralised structure, in many areas of decision making including scheduling. In this context, decision problems are delegated to lower levels of the organisational hierarchy and solved locally and independently by different entities of the system. The solutions are then coordinated together under a global objective. These entities may be different departments in a company, cells in a flexible manufacturing system (FMS), multiprocessors in a communication network, or companies in a supply chain. Decision making in such a distributed manner increases system responsiveness, which may be especially important for scheduling. Scheduling as a short term decision-making process requires up-to-date and timely information to generate feasible schedules in practice.

The need for making scheduling decisions in decentralised systems has given rise to a new area, that is, distributed scheduling. We define distributed scheduling (DS) as an approach in which smaller parts of a scheduling problem are solved by local decision makers who possibly have conflicting objectives and who coordinate their subsolutions through certain communication mechanisms to achieve overall system objectives. This way, local decisions are quickly made using the most recent system information, and the overall schedule is more responsive to dynamic and unpredictable events, such as machine breakdowns, new job arrivals, or order cancellations.

In this paper, we provide a survey and a critical analysis of the literature on distributed scheduling. Examining previous work on distributed systems we note that several papers discuss a variety of issues, such as the design of distributed architectures for production organisation (see, e.g. Biemans and Vissers 1991, Crowe and Stahlman 1995); the development of modelling languages for negotiation (Wang *et al.* 2002); and methodologies for the integration of distributed information systems (Sikora and Shaw 1998, Jeong and Leon 2002). In this paper, however, we review in depth the body of research in which scheduling algorithms within a proposed distributed architecture are explicitly discussed. In view of this perspective, we seek answers to the following questions:

Q1. How do the decentralised scheduling systems differ in theory and in implementation from the centralised scheduling systems?

Q2. What are the critical design decisions in a distributed scheduling system?

Q3. What are the important commonalities and differences among different distributed scheduling systems proposed in the literature?

Q4. How do the design aspects affect computational time and solution quality?

Distributed scheduling can be considered as part of a broader concept in production planning and control, that is, *multi-agent systems*. These systems assume the presence of multiple decision makers (i.e. agents) interacting and cooperating with each other in order

to achieve a global performance (Caridi and Cavalieri 2004). There are several applications of this concept ranging from order quotation to design to distribution. Several papers provide reviews of the literature on multi-agent systems (Sen 1997, Tharumarajah 2001, Shen 2002, Caridi and Cavalieri 2004, Giret and Botti 2004, Shen *et al*. 2006). However, with the exceptions of Shen (2002) and Shen *et al*. (2006), none of these studies focus on a specific application area. These two papers provide an account of the literature on manufacturing scheduling. Shen (2002) discusses the issues in agent-based manufacturing scheduling in relation to the general concept of multi-agent systems. Shen *et al*. (2006) extend this work to include the studies on agent-based approaches to manufacturing process planning. However, these studies provide limited evidence about how the existing work on distributed scheduling fits into the classical literature on scheduling theory.

It is important to emphasise that, although there are several review papers in many areas of scheduling (Graves 1981, Cheng and Sin 1990, Basnet and Mize 1994, Sabuncuoglu 1998), there is no survey of the literature on distributed scheduling with the traditionally adapted view of the scheduling theory. Our work contributes to the existing literature by (i) providing a perspective on distributed scheduling as it relates to scheduling theory and practice, (ii) unifying the current distributed scheduling literature and synthesising it under a comprehensive taxonomy, (iii) offering future research directions and opportunities for improving existing models.

Since the late 1980s, distributed scheduling has received considerable attention in the scheduling literature (Shaw 1987, Roundy *et al*. 1991, Kutanoğlu and Wu 1999, Babayan and He 2004, Jeong and Leon 2005). Although using DS concepts in solving scheduling problems has largely been limited to the manufacturing area (Burke and Prosser 1991, Chung *et al*. 1996, Maturana and Norrie 1996), a growing interest has emerged in other applications such as network power scheduling, fair scheduling in wireless LANs and packet scheduling (Chiussi and Francini 2000, Hohlt *et al*. 2004, Vaidya *et al*. 2005).

Another important application area of distributed scheduling is supply chains. In these systems, companies make their decisions independently and with partial information from one another. Proper coordination of decisions in supply chains for inventory replenishment and transportation has been shown to increase overall system profits (Tsay *et al*. 2000). The potential benefits of coordinating scheduling decisions have been recently recognised (Lau *et al*. 2005a, 2005b). This is primarily due to the fact that information that is entered into the scheduling problem changes quickly, and overcoming these difficulties has become possible only with recent advancements in information technology. This progress has not only made it possible to achieve coordination through distributed scheduling, but it has also changed the structures of supply chains, indicating a need for distributed scheduling. For example, 'Business To Business' (B2B) marketing on the Internet is a new practice in which the ability to make quick scheduling decisions and to coordinate them with those of the other parties is vital. Therefore, the changing structure of supply chains, along with advancements in information technology, suggests that distributed scheduling will attract more attention in the future (Pinedo 2002).

In the next section, we provide the basic definitions and concepts in distributed scheduling, and summarise the specific notation used in the current paper. Then, we propose a classification framework and a brief review of the existing literature, including the answer to Q3. In the final section, we present our concluding remarks with an emphasis on the other questions we have raised and outline further research directions in this area.

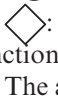
2. Basic definitions and notation

In distributed scheduling, the problem is divided into subproblems, each of which is assigned to a local decision maker. Local decision makers are called *agents* (Sycara *et al.* 1991, Kouiss *et al.* 1997, Shen 2001). The concept of agents was first introduced in computer science to create autonomous and intelligent entities that act like humans. In theory, agents should posses some basic properties such as autonomy to operate without human intervention, social ability to communicate with others, pro-activeness to take an initiative role, and reactivity to respond to changes in the system (Rahimifard and Newman 1998). In production settings, an agent is normally associated with, among others, a part, a product, an order, a resource, a department, or a group technology cell.

We classify agents into two types: local and global. Local agents, a.k.a. regular agents, take a role in local decisions, such as developing and updating the partial schedule for a subproblem. In some systems, assignment of subproblems to the local agents is handled by the global agent, who is alternatively called a manager agent, mediator agent, or a master agent. A global agent should have access to all information sources in the system. In light of the overall system objectives, this agent also resolves any conflicts among independently-made local scheduling decisions. A global agent is viewed as the only intelligent entity with infinite life and has the liberty to create and delete regular agents. In that sense, regular agents are sometimes called temporary agents. All these concepts related to agents, multi-agent techniques, and their applications in different areas are discussed in detail by Zhang *et al.* (1998).

Before presenting a detailed classification scheme to synthesise the distributed scheduling literature, let us introduce the schematic notation that will be used to illustrate different entities in a typical DS system.

 : Local agent

 : Global agent (In some systems, local schedulers can temporarily carry out some functions of a global agent. In such cases, we use this figure within a circle, i.e. )

The above two figures are used to represent agents, who are the only intelligent entities. We also introduce the following notation for other entities that do not take part in decision making, but are useful in understanding the dynamics of a distributed scheduling system:

 : Coordinator (A coordinator acts more like a global information database. It aids in information exchange between agents, but has no role in decision making.)

 : Controller, which simply implements the scheduling decisions.

3. Classification framework

Previously, scheduling research has been classified as static versus dynamic, deterministic vs. stochastic, and online vs. offline (Leung 2004). With the increasing importance of information technology in the implementation of scheduling systems, there is a need for new classification schemes. For this purpose, we propose two general attributes: the *information flow structure* and the *communication mechanism* between decision makers. With respect to the former, we classify the literature into two main groups: centralised (or hierarchical) systems and decentralised (or distributed) systems. For the latter attribute, we identify six mechanisms: bidding, iterative bidding, negotiation, cooperation, domination, and iterative refinement.

The above two attributes, which will be explained in more detail later in this section, are the primary bases for our general classification. However, in order to capture the

characteristics of the distributed-scheduling papers in the broader context of scheduling theory, we propose a detailed classification framework. This framework is based on the following seven attributes: *information flow structure*, *communication mechanism*, *local agent types*, *assignment method*, *schedule generation*, *objective*, and *machine environment*.

The third entry in this framework identifies the physical entities in the system that possess the features of an agent, as we explained in the previous section. In some studies, several system modules that do not take part in decision making are also referred to as 'agents' (e.g. a database management agent in which information is stored and updated, a communication agent that aids in distributing messages). It is important to note here that we exclude such entities from being considered within this attribute. Also, because the information flow structure in our classification scheme already identifies whether there is a separate manager agent or not, here, we only consider the *local agent types*. We refer to an agent that stands for an operation or a collection of operations as an *order agent*. Similarly, an agent that is associated with a resource (e.g. machine, tool, or material handling equipment) or a collection of resources is referred to here as a *resource agent*.

In distributed scheduling, the main scheduling problem is decomposed into subproblems to be solved by the local agents. The *assignment method* element in the notation indicates the main procedure or tool used in relating the subproblems to the local agents. The fourth entry, s*chedule generation*, describes how the subproblems are solved by the local agents after their assignments are made. For example, scheduling the operations of a job may be assigned to a group technology cell using bidding, and the resource agent responsible for that cell may generate an output in the form of a detailed schedule using an algorithm (e.g. mathematical model, tabu search, or simulated annealing).

The sixth attribute of our classification scheme is the *objective*, which describes the performance measure to be optimised (e.g. reducing flowtime, tardiness, WIP, or lead time or increasing utilisation or throughput). A distributed scheduling system enables the local decision makers to have their own scheduling objectives, which we refer to as *local objectives*. The local objectives of various decision makers may be different from one another and from the overall system objectives, which we refer to as *global objectives*. As will be discussed in Section 5, our review suggests that almost all the studies in the existing literature assume that local objectives are the same as the global objectives, and usually there is only a single objective. In fact, one of our observations is that, although DS allows the local agents to act on behalf of themselves, the degree to which local objectives are achieved is not thoroughly addressed in the current literature. Finally, *machine environment* defines the setting in which the prospective scheduling system is intended to work (e.g. job shop, single machine, or FMS).

3.1 *Communication mechanisms*

In *bidding* systems, a new job, either by itself or with the help of a manager agent, broadcasts its arrival and requests bids for its processing. The agents (machines or cells) prepare bids by considering their own capabilities. Those incapable of processing the job do not bid. The best bid is selected by the manager agent according to some pre-determined criteria. The Contract-Net protocol is a pioneer work that employs bidding concepts in distributed problem solving (Smith 1980, Smith and Davis 1981), and is also a basis for several distributed production planning systems (Parunak 1987, Shaw and Whinston 1988, Lima *et al.* 2006). Different variations of bidding mechanisms have

been studied in the literature. Yang *et al.* (1993) proposed an aggregate bidding scheme in which pre-emption is handled within the bidding process for scheduling urgent jobs. In some cases, decisions are finalised after one iteration of bid collection, whereas in others, agents are allowed to revise their bids in light of new information gained from the previous bids. To distinguish this case from regular bidding mechanisms, we call it an *iterative bidding* mechanism (e.g. Lin and Solberg 1992).

In a *negotiation* mechanism, agents at the same hierarchical level communicate with each other or exchange information to prepare better schedules for their subproblems. In the negotiation case, optimisation of local goals is of high priority, but in a *cooperation* mechanism, agents collaborate with each other to achieve better system performance. As a result of this mechanism, agents may choose suboptimal policies for their local goals for the sake of overall system performance. The work by Crowe and Stahlman (1995) provides a good discussion of the above four mechanisms. Another communication mechanism is *iterative refinement*. It operates by an exchange of schedule information between different agent types, for eliminating conflicts or for revising the existing schedule towards better system performance. The coordination mechanisms discussed above are unique to distributed scheduling systems: they require different intelligent entities to be involved in the decision process, rather than a single entity preparing the schedule for the whole system.

As a final communication mechanism, we note *domination*, which prevails both in centralised and decentralised systems. In this mechanism, a higher-level agent or module decides on the schedule, and then, lower-level ones implement these decisions. The degree of domination used in a decentralised system depends on the extent of decentralisation. If an agent has dominance over another agent, a decision made by the dominant (or master) agent becomes a constraint on the latter. The communication mechanisms covered in this subsection will also be used in the following discussion on information flow structures.

3.2 *Information flow structure (centralised versus decentralised)*

A scheduling system is called centralised (or hierarchical) if a global scheduler develops a schedule for the entire system. As seen in Figure 1, the flow of information is basically from top to bottom with a domination-based communication mechanism. Figure 1(a) illustrates a *single-layer centralised* system, in which there is one level of local controllers. These controllers are in charge of implementing the part of the final schedule that is communicated to them by the global controller. Another variation of centralised systems is the *multi-layer* structure, which is illustrated in Figure 1(b). In this case, there is a hierarchical level of local controllers in which a higher level controller passes the received scheduling information in smaller parts to the lower levels. This one-way information flow is represented by the down arrows in Figure 1. Most of the existing scheduling systems are of this type (i.e. centralised).

In contrast to the centralised systems, there is no single decision maker in a decentralised system. Local agents make their decisions to solve the smaller parts of the scheduling problem. We can further classify the decentralised systems as *pure heterarchical* and *quasi-heterarchical*. Pure heterarchical systems contain no command hierarchy. Each agent communicates through a coordinator agent or in pairs (see Figures 2(a) and 2(b), respectively). These systems mostly utilise iterative refinement as a communication mechanism, which usually yields no more than a feasible schedule. Local agents in pure

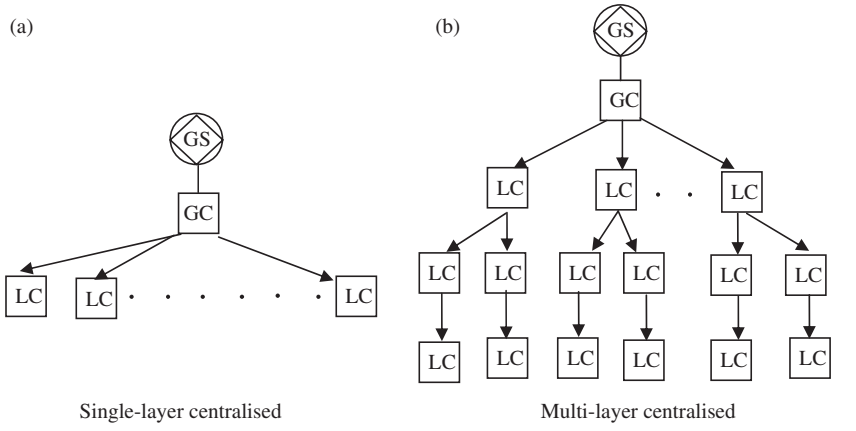


Figure 1. Centralised scheduling architectures (LC: local controller, GC: global controller, GS: global scheduler).

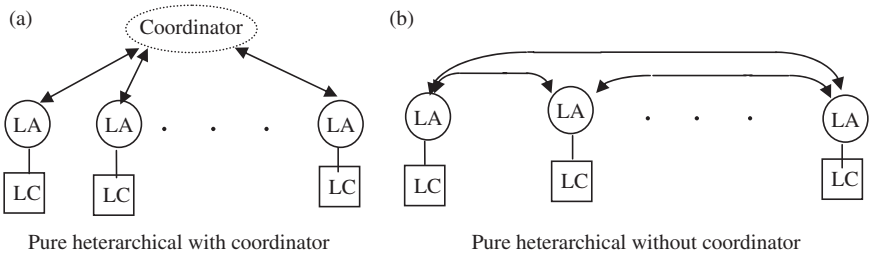


Figure 2. Pure heterarchical DS architectures (LA: local agent, LC: local controller).

heterarchical systems only consider their own goals, and they do not cooperate with each other (e.g. Sycara *et al*. 1991).

In a quasi-heterarchical system, local schedulers again make their own schedules considering their goals. However, these local schedules are then evaluated by the master agent with respect to the overall system objective. More specifically, the role of the master agent in these systems is to resolve potential conflicts, to select bids, and to finalise the scheduling decisions. We further classify the quasi-heterarchical systems as *single-layer quasi-heterarchical* if there is one layer of local decision makers (Figures 3(a) and 3(b)), and *multi-layer quasi-heterarchical* if there is more than one such layer (Figure 4).

There are variations of single-layer quasi-heterarchical systems depending on whether there is a separate manager agent or not. Figure 3(a) illustrates a *single-layer quasi-heterarchical system without a separate manager agent*, in which the functions of a manager agent are temporarily carried out by a local agent. Figure 3(b) illustrates the complementary case: the *single-layer quasi-heterarchical system with a separate manager agent*. In quasi-heterarchical systems, communication between local agents usually involves cooperation, negotiation, bidding or a combination of these. Note that in multi-layer quasi-heterarchical systems, there is always a manager agent for achieving better control of multiple layers (see Figure 4 for an illustration).

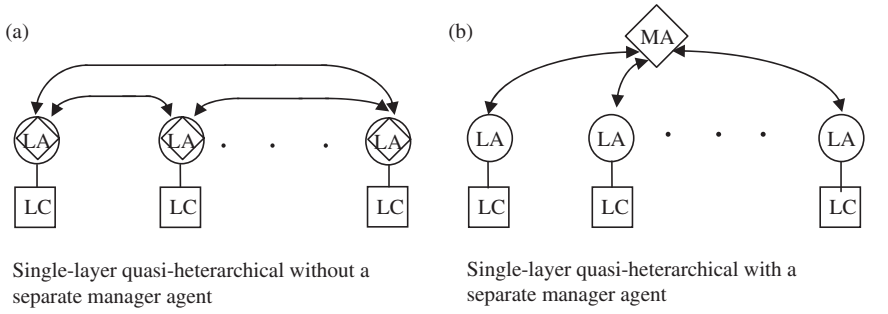


Figure 3. Single-layer quasi-heterarchical DS architectures (MA: manager agent).

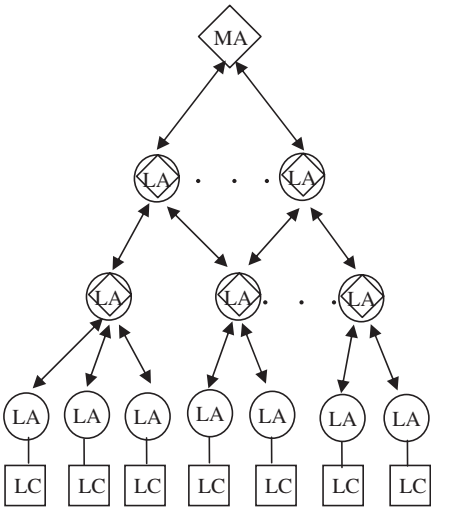


Figure 4. Multi-layer quasi-heterarchical DS architecture.

3.3 *A comparison of the proposed taxonomy with the other relevant categorisations in the literature*

Distributed scheduling has been an area of research that attracts the interest of many researchers from a variety of disciplines such as computer science, economics, manufacturing and service operations management. One reason is that, the problems in this area cover a large span of issues ranging from information architectures to negotiation mechanisms, to design of scheduling algorithms. While a classification of distributed scheduling papers cannot be done ignoring the issues related to the distributed or the multi-agent structure, they alone are not enough to characterise a scheduling paper. There are various classification schemes that can be used to give an account of how papers on distributed scheduling fit into the multi-agent systems literature. Examples include Sen (1997), Tharumarajah (2001), Shen (2001), Shen (2002), Giret and Botti (2004), Caridi and Cavalieri (2004), Shen *et al*. (2006). However, these classifications ignore the issues addressed in the classical scheduling theory that are also relevant to distributed scheduling, such as the underlying scheduling problem or the solution approach.

After carefully analysing the existing classification schemes, we have also observed that some of them are very focused (e.g. Giret and Botti 2004) or variations proposed by the same author (e.g. Shen 2001, Shen 2002, Shen 2006). Therefore, in the rest of this section, we provide a comparison of the classification schemes by Tharumarajah (2001), Shen *et al*. (2002), and Caridi and Cavalieri (2004) with ours, as they are the most comprehensive ones relevant to our study.

Tharumarajah (2001) classifies the literature on distributed manufacturing systems based on the following attributes: *problem decomposition*, *problem-solving organisation*, *coordination and control*. The attribute *problem decomposition* describes how the global problem is decomposed. The three values that this attribute may attain (resource view, task view and hybrid view) are sufficient to describe the decomposition method, which is also reflected by the *agent* attribute in our classification scheme. However, it does not by itself provide sufficient evidence about how the individual subproblems are assigned to a specific decision maker among alternative ones. This is achieved by the *assignment* attribute in our classification scheme. The other two attributes *problem-solving organisation*, *coordination and control* in Tharumarajah (2001) are similar to *information flow structure* and *communication mechanism* in ours. The author also identifies *communication*, *local problem solving* and *performance* as issues relevant to distributed manufacturing systems, however, he does not provide a classification of the literature with respect to these attributes.

Shen (2002) discusses *agent encapsulation*, *coordination and negotiation protocols*, *system architectures* and *decision schemes for individual agents* as four issues of agent-based manufacturing scheduling. The first three of these issues are captured by *agent*, *communication mechanism*, *information flow structure*, respectively, in our classification scheme. Within the *decision schemes for individual agents*, the author includes both the coordination or negotiation mechanisms, and the local decision-making mechanisms. The discussion on local decision-making is confined to the knowledge update and information sharing of the decision makers rather than an analysis of scheduling methods and/or solution approaches for local problems. Furthermore, the author does not provide a classification of the reviewed papers in terms of the above issues listed.

Caridi and Cavalieri (2004) provide a taxonomy for classifying the multi-agent systems. This taxonomy includes *application domain*, *agent*, *control*, *organisation*, *communication*. Our study focuses on distributed scheduling as an application domain. The attributes *agent*, *communication mechanism*, *information flow structure* in our classification scheme are similar to *agent*, *control*, *organisation* in this taxonomy. The attribute *communication* in Caridi and Cavalieri (2004) is not included in ours. The authors use this attribute to describe the communication vehicle and the semantic structure of the messages exchanged between agents. This is a general issue on multi-agent systems, and in our paper, we do not provide a detailed analysis of the literature with respect to this attribute.

Within the context of classification taxonomies, our study contributes to the existing literature in three ways. First, in addition to the attributes that reflect the multi-agent or distributed structure of a system, we have attributes to describe the scheduling aspect (i.e. *objective*, *machine environment*, *assignment method* and *schedule generation*). Our careful review of the literature leads to an identification of a wide variety of values that each one of these attributes may assume. Second, we provide a detailed analysis and a summary table for how we classify each paper within our focus in terms of these attributes. Last but not least, our classification scheme derives from concepts put forward in the

literature and proposes a unified system of terminology for similar issues which are most often referenced under varying names by different authors.

4. Analysis of existing studies

In this section, we review the current literature on distributed scheduling, beginning with a very simple heterarchical structure (a pure heterarchical system) and continuing with single-layer quasi-heterarchical systems. Finally, multi-layer quasi-heterarchical systems are considered. The papers are also summarised in Table 1, according to the new classification framework we propose.

4.1 *Pure heterarchical systems*

Pure heterarchical systems have the simplest structure in terms of information flow. Agents communicate directly with each other, and no agent has the right to override the decisions of the others. Sycara *et al*. (1991) and Liu and Sycara (1993) are examples of this class. In both of these studies, local agents are associated with orders and resources. Scheduling decisions are revised iteratively, and information is exchanged between the order and resource agents until a feasible schedule is reached. A feasible schedule is defined in terms of precedence and capacity constraints. A precedence constraint is satisfied when the precedence relations between the activities of an order are maintained. A capacity constraint requires that no more than one activity be scheduled for the same period for the same resource. Accordingly, a feasible schedule is obtained when the precedence constraints over all orders and the capacity constraints over all resources are satisfied. Within this framework, the details of these two studies are given below.

In Sycara *et al*. (1991), each order agent first determines its overall demand in time for the required resources. It then presents this data to the resource agents. After collecting the demands from all order agents, each resource agent calculates the overall aggregate demand in time for its resource. The peak of the aggregate demand determines the critical time interval of the resource. The operations that are scheduled within this interval are called critical activities. Starting with the most critical activity, each resource agent puts the current activity into its existing schedule. Different methods are proposed for the resource agents to schedule activities. One such method is based on the survivability measure of an activity: this represents the probability that a start time will not cause a capacity constraint violation for the current activity. After each activity is scheduled, order agents update their demand information at the related resource. The information flow between the resource agents and order agents continues in an iterative manner, until a feasible schedule is developed (Figure 2(b) illustrates the information-flow structure).

In Liu and Sycara (1993), in addition to the order and resource agents, there is a coordinator responsible for the exchange of information and for communication between different agent types (Figure 2(a)). The system begins operation with the generation of initial schedules by the resource agents. Each resource agent simply sequences all the operations to be scheduled on its resource using the EDD (Earliest Due Date) rule. As the resource agents act independently from each other, this schedule may violate precedence constraints. Information about the operations of each job is extracted and sent to the related order agent by the coordinator. Next, order agents identify and eliminate precedence violations. This may be followed by an update of the schedule by the resource

Table 1. List of the studies in distributed scheduling and their characteristics.

Authors	Information flow structure	Communication mechanism	Local agent types	Assignment method	Schedule generation	Objective	Machine environment
Shaw (1987)	Single-layer Q.H.* without a separate MA	Bidding	Resource (GT† cell)	Bidding algorithm with EFT	FCFS Dispatching	Not explicitly stated	FMS
Shaw and Whinston (1988)	Single-layer Q.H. without a separate MA	Bidding	Resource (GT cell)	Bidding algorithm with SPT	FCFS Dispatching	Not explicitly stated	FMS
Ow *et al.* (1988)	Single-layer Q.H. with a separate MA	Cooperation and Bidding	Resource (Workcentre)	Bidding	Detailed schedule	Minimising completion times and flowtimes	Job shop
Parunak (1988)	Multi-layer Q.H.	Cooperation, Bidding and Negotiation	Resource (GT cell)	Bidding algorithm with EFT	FCFS Dispatching	Meeting due dates and balancing the load	FMS
Burke and Prosser (1991)	Multi-layer Q.H.	Cooperation	Resource	Depth first search and Backtracking	Detailed schedule	Meeting due dates and balancing the load	Job shop
Sycara, Roth, Sadeh and Fox (1991)	Pure heterarchical without coordinator	Iterative refinement	Order and Resource	Iteration	Detailed schedule	Not explicitly stated	Not given
Roundy *et al.* (1991)	Single-layer Q.H. with a separate MA	Cooperation	Resource	No assignment	Dispatching	Minimising total weighted tardiness	Job shop
Lin and Solberg (1992)	Pure heterarchical without coordinator	Negotiation and Iterative bidding	Order and Resource	Bidding	Detailed schedule	Several objectives	Not given
Hadavi, Hsu, Chen and Lee (1992)	Single-layer Q.H. with a separate MA	Cooperation	Resource	Dispatch rule selection via simulation	Dispatch rule selection	Meeting due dates and Reducing inventories	Job shop

Table 1. Continued.

Authors	Information flow structure	Communication mechanism	Local agent types	Assignment method	Schedule generation	Objective	Machine environment
Liu and Sycara (1993, 1994, 1995)	Pure heterarchical with coordinator	Iterative refinement	Order and Resource	Iteration	Detailed schedule	Meeting due dates	Job shop
Duffie and Prabhu (1994)	Single-layer Q.H. without a separate MA	Cooperation	Resource	Dispatch rule selection via on-line simulation	Dispatching	Minimising tardiness	FMS
Chiu and Yih (1995)	Single-layer Q.H. with a separate MA	Cooperation	Resource	Dispatch rule selection via GA and off-line simulation	Dispatching	Weighted sum of several objectives	FMS
Agarwal *et al.* (1995)	Single-layer Q.H. with a separate MA	Cooperation and Bidding	Resource (GT cell)	Bidding with SPT and customer approval	Detailed schedule	Minimising completion times	FMS
Duffie and Prabhu (1996)	Single-layer Q.H. without a separate MA	Cooperation and Bidding	Resource	Dispatch rule selection via on-line simulation, and bidding	Dispatching	Minimising tardiness	FMS
Chung, Park, Kang and Park (1996)	Single-layer Q.H. without a separate MA, (Centralised in IP case)	Cooperation and Bidding	Resource	Bidding with SPT and IP model	SPT Dispatching	Minimising flowtime	FMS
Maturana and Norrie (1996)	Multi-layer quasi-heterarchical	Cooperation and Bidding	Order and Resource	Bidding	Dispatching	Meeting due dates	Job shop
Kouiss *et al.* (1997)	Single-layer Q.H. with a separate MA	Cooperation	Resource (workcentre)	Random	Dispatch rule selection	Not explicitly stated	FMS
Oguz (1998)	Single-layer Q.H. with a separate MA	Cooperation	Resource	Decomposition	Dispatching	Meeting due dates and Minimising makespan	Job shop
Kutanoğlu and Wu (1999)	Single-layer Q.H. with a separate MA	Bidding	Order	Lagrangian relaxation using subgradient search	Detailed schedule	Minimising total weighted tardiness	Job shop

Dewan and Joshi (2000)	Pure heterarchical without coordinatior	Iterative bidding and Negotiation	Order and Resource	Decomposition through Lagrangian relaxation	Dispatching	Minimisation of total earliness and tardiness penalties	Single machine
Wellman and Walsh (2001)	Single-layer Q.H. with a separate MA	Bidding and Negotiation	Order	Bidding	Detailed schedule	Meeting due dates	Single machine
Dewan and Joshi (2002)	Pure heterarchical without coordinator	Iterative bidding and Negotiation	Order and Resource	Decomposition through Lagrangian relaxation	Dispatching	Minimisation of total earliness and tardiness penalties	Job shop
Babayan and He (2004)	Single-layer Q.H. with a separate MA	Cooperation and Negotiation	Order	Cooperative game theory	Mixed integer programming	Minimising makespan	3-stage flexible flowshop
Lau *et al.* (2005a, 2005b)	Pure heterarchical with coordinator	Iterative bidding and Negotiation	Order and Resource	Integer programming	Detailed schedule	Minimisation of earliness and tardiness costs	Supply chain
Liu *et al.* (2007)	Pure heterarchical without coordinatior	Iterative bidding and Negotiation	Order and Resource	Decomposition through Lagrangian relaxation	Dispatching	Minimisation of total weighted tardiness	Job shop
Wang *et al.* (2008)	Single-layer Q.H. with a separate MA	Cooperation and Bidding	Resource (GT cell)	Bidding	Dispatching and mixed integer programming	Minimising weighted sum of makespans over workcells	FMS
Wang *et al.* (2009)	Pure heterarchical without coordinator	Bidding and Negotiation	Order and Resource	Winner determination problem	Constraint-based branch and bound algorithm	Maximisation of sum of bid prices for satisfaction of due-dates	Job shop

*Quasi-heterarchical, †Group Technology, ‡Integer Programming.

agents to satisfy capacity constraints. The procedure continues in an iterative manner until a feasible schedule is generated. In later studies, the authors (Sycara and Liu 1994, 1995) propose mechanisms for loop prevention that aim at minimising conflicts involving the same operations.

Another pure heterarchical system is proposed by Lin and Solberg (1992). In this system, scheduling decisions are carried out by iterative bidding between resource and order agents. There is an order agent for each part and a resource agent for each resource in the system, including machines, tools, and transporters. Resource agents have varying charge prices for different time slots, and the objective of an order agent is to optimise a weighted sum of customer needs (e.g. due date or quality) while minimising the resource charge prices it pays. When the current operation of a part is close to completion, or when a new part enters the system, the corresponding order agent solicits bids from eligible resource agents for the subsequent operation. After collecting bids that arrive within a certain period, the order agent awards the resource agent that gives the best bid. However, a resource agent may submit bids for several other parts simultaneously. Thus, by the time it is awarded the operation, the resource agent may not be available to process it, or it may choose to do another task that better satisfies its local objectives. In this case, it rejects the bid, and the order agent restarts the bidding process. Therefore, there is a strong negotiation scheme through which order agents compete with each other to reserve resources.

A similar system based on price adjustment is proposed by Dewan and Joshi (2000) for minimising the total earliness and tardiness penalties on a single machine. In this system, however, the resource agent solicits bids from the order agents who act as bidders. Subproblems of the agents are decomposed from the original problem using Lagrangian relaxation of the integer programming formulation. Whenever the machine becomes available, the resource agent announces an auction and the order agents bid for the time slots on the machine. At each step of the iterative bidding mechanism, by solving their subproblems, order agents may update the money they are willing to pay for their desired periods, and the resource agent may adjust the prices of the slots. The resource agent and the order agents negotiate until a termination condition is reached. Building upon Dewan and Joshi (2000), the authors suggest in a later study a similar system for solving the job shop scheduling problem with the same objective (Dewan and Joshi 2002). In this system, resource and order agents are associated with machines and jobs, respectively. The suggested system incorporates a similar approach of decomposition and iterative bidding mechanism between the resource agents and the order agents. Liu *et al.* (2007) extend this study with a rolling time horizon procedure to minimise the total weighted tardiness in a dynamic job shop environment.

Wang *et al.* (2009) solves the job shop scheduling problem modelling it as Winner Determination Problem (WDP). In auction theory, WDP is defined as finding an allocation of items to bidders that maximises the auctioneer's revenue (Rothkopf *et al.* 1998). The proposed system by Wang *et al.* (2009) regards jobs as bidders, that is, an order agent is associated with each job. Order agents may give more than one bid for the processing of their jobs. Each bid includes a latest completion time and the valuation of the order agent in terms of price for the completion of the job before the specified time. WDP is used to decide on the selection of jobs and their schedules with the objective of maximising the sum of bidder prices. This problem is solved using a constraint-based branch and bound algorithm by an intelligent entity that represents all the resources. Therefore, we presume the existence of a resource agent associated with all the resources in

the job shop. We also note that this problem setting differs from the classical job shop environment in the sense that only the selected jobs through the WDP are scheduled.

The above studies suggest that the local agents in pure heterarchical systems normally share little or no global information. Agents not only make their scheduling decisions with respect to local information, but they also compete selfishly to optimise their local goals, rather than the overall system objective. The eligibility of the local agents to coordinate the schedules they generate in their own information privacy, makes the pure heterarchical structure convenient to apply for supply chain scheduling. The next paper is an example that draws on the literature discussed above and captures the nature of this structure to coordinate the scheduling decisions of companies.

The scheduling system Lau *et al*. (2005a) propose has a pure heterarchical structure with a coordinator. Companies that want to contract the operations of their jobs (project agents) act as order agents, and those who offer their bids (contractor agents) act as resource agents. The coordination between these supply chain entities is enhanced by an iterative revision of schedules and an exchange of information until an agreement about the start times of operations is reached. This information on schedule flexibility consists of three measures: the time window, the lower bound on the start time, and penalty costs. A time window is a range of start times for an operation within which shifts do not affect the earlier scheduled operations. An agent is not allowed to schedule an operation before the lower bound and pays a penalty cost, if the start time is not within the time window. As part of the negotiation process between the order and resource agents, this information is generated at each pass, after an agent solves its subproblem. Order agents collect the bids for all operations of a job simultaneously. An order agent's subproblem is to minimise the sum of earliness and tardiness costs and the penalty costs of violating the time windows. The solution to an order agent's subproblem identifies the start times of all its operations and a selection of resource agents. A resource agent's subproblem is to maximise the net gain from operations, considering the penalty costs of violating time windows. Lau *et al*. (2005b) show the convergence of this iterative procedure and report the results of their extensive computational study on the proposed system.

4.2 *Single-layer quasi-heterarchical systems without a separate manager agent*

Quasi-heterarchical systems are similar to pure heterarchical systems in terms of their architecture: they both have one level of local decision makers. But one major difference between them is the amount of global information owned by the local agents. Our analysis in this section focuses on a specific group of quasi-heterarchical systems. These have no separate manager agent, but local agents have enough information about the global objectives to temporarily act as a manager agent to resolve the conflicts in the system (see Figure 3(a)).

The earliest study in this area is by Shaw (1987). In this system, each agent (representing an FMS cell) acts as a bid manager to route the job to another cell for the next task or set of operations. FMS cells, which are capable of performing this task, offer their bids to the bid manager. In the event of a new job arrival, any idle cell in the system can act as a bid manager. When a cell receives the task announcement message via the communication network, it calculates the earliest finishing time (EFT) using the processing time, travelling time, and expected waiting time information. This information is sent back to the bid manager, who makes the final decision. If more than one task announcement

comes to an FMS cell, the cell ranks the tasks according to some local criteria (i.e. setup time or job urgency). In a follow-up study, Shaw and Whinston (1988) employed the SPT (shortest processing time) rule as the bidding criterion instead of the EFT, in a Petri net-based communication protocol.

According to Duffie and Prabhu (1994), each resource is assigned to an agent. In addition to maintaining local information, each agent is capable of simulating alternative plans for their scheduling decisions. Also, a separate entity keeps the information regarding new job arrivals. However, this entity does not have any role in decision making (Figure 3(a)). As a new job arrives, this information is passed to resource agents. Resource agents affected by the arrival of this job develop alternative scheduling plans and simulate them to calculate local performance measures. Because resource agents act independently, their decisions may conflict. One or more of the agents finds the conflicts and calculates the global performance measure. With this feedback, local agents come up with alternative local plans. When a local plan results in better system objectives, it is declared to be the final schedule. This system is a perfect example of a cooperation mechanism. In their later study, the authors (Duffie and Prabhu 1996) consider alternative machines and use a bidding mechanism to make the selection.

The third study in this area is by Chung *et al*. (1996). The information flow structure of the proposed system is similar to that of Shaw (1987). In the former paper, more issues specific to cellular manufacturing (e.g. scheduling of material handling systems, limited buffer capacities) are addressed through a detailed consideration of alternative scenarios for the resource assignment problem. When a part is to be assigned to one of the m machines, both Shaw (1987) and Chung *et al*. (1996) propose that the assignment decision be made through a bidding procedure. Here, the machine that processes the most recent operation on the part acts as a bid manager. When the assignment decision is to be made between n parts and m machines, Shaw (1987) suggests that for those parts that are unassigned after an iteration of a bidding process, and the process is repeated until all parts are assigned. Chung *et al*. (1996), on the other hand, propose an integer programming model to solve this assignment problem. Once the resource assignment decisions are made, the sequencing of parts over the resources uses the SPT dispatching rule.

4.3 *Single-layer quasi-heterarchical systems with a separate manager agent*

Because local decision makers in a DS environment act independently from each other, some of the existing applications use a separate manager agent to resolve conflicts. The existence of such an agent with a global view also helps to improve the global performance of the system. We have identified 11 studies with this structure.

Hadavi *et al*. (1992) present a system called REDS. In addition to performing as a predictive scheduler, this system is particularly designed for reactive scheduling. The architecture consists of several servers with different functionalities; however, only three of them take an active role in scheduling: event handler, order watcher, and capacity watcher. Event handler, which acts like a manager agent, keeps and updates the system information, receives incoming orders, and makes the actual dispatch decisions. The order watcher (order agent) makes both an aggregate plan of tasks to be processed by each workcentre, over a distant time horizon, and a detailed plan for a shorter time period. The system allows for multiple-capacity watchers (resource agents), each associated with a

different workcentre. Both the capacity watcher and the event handler can request a job. The communication mechanism is of the cooperation type, as the sequencing decisions are made considering the overall system objectives (Table 1).

Chiu and Yih (1995) propose another single-layer quasi-heterarchical scheduling system. Here, the objective is to minimise the weighted sum of the makespan, the number of tardy jobs, and the maximum lateness. A scheduling point occurs when the input queue of a machine is empty. At each scheduling point, a resource agent, which stands for a machine, changes its dispatching rule according to the current system state to select the next part. Possible system states are generated by the manager agent using simulation. The best dispatching rule at a scheduling point for any system state is also decided by the manager agent, using a genetic algorithm. This is referred to as a 'knowledge-based approach'. The function of a resource agent in decision making is fairly limited. It can only initiate the retrieval of a dispatching rule by the manager agent for the current system state, whenever it reaches a scheduling point. Because the ultimate selection of the dispatching rule is done by the manager agent, we consider this system to be single-layer quasi-heterarchical with a separate manager agent (Figure 3(b)). The best dispatching rule is chosen solely to maximise the overall system objective; hence, the communication mechanism is of the cooperation type.

A similar system to that of Chiu and Yih (1995) is proposed by Kouiss *et al*. (1997) for FMSs. In this system, an agent refers to a workcentre. Agents make their schedules using dispatching rules which they dynamically choose according to the system state and their local objectives. It is not explicitly stated how a job is assigned among alternative workcentres, however, it is assumed in the experimental analysis that routing of jobs is random. The proposed system also incorporates a manager agent that monitors the global objectives and imposes particular dispatching rules to the agents if he/she sees necessary. This system enables the workcentres to choose their own dispatching rules while forcing cooperation through the manager agent for better system performance.

Agarwal *et al*. (1995) developed a distributed scheduling system for flexible cellular job shops. Resource agents (referred to as 'Cell Level Managers') represent cells that are formed according to group technology principles, i.e. each cell produces parts belonging to a certain part family. Furthermore, with proper tooling, all the operations of a job (customer order) can be performed on a single machine. It is assumed that jobs arrive at the system one at a time. A manager agent initiates a bidding process upon the arrival of a job and coordinates the communication between cells. Resource agents quote completion times for the current job considering machine utilisation and load balancing within their cells. The bidding process is executed in four phases. In phase 1, each resource agent appends the new job to the end of the schedule on the preferred machine (i.e. the one that results in the minimum makespan for the cell). The manager agent evaluates the offers and informs the customer about the earliest completion time. If it is not acceptable, phase 2 is initiated and cells prepare their bids to find the earliest available time slot on a machine. If the earliest completion time at the end of this phase is not acceptable, resource agents prepare new bids in phase 3 by breaking down the operations of the job into subsets and allowing each subset to be produced on a different machine. If a feasible offer is still not found, in phase 4, the manager agent requests bids for operation subsets from different cells considering both precedence relationships between operation subsets and the transportation time between cells. If there is no feasible offer for the job, the job either is rejected or returned to the bidding process, after negotiating its due date with the customer. Because the manager agent makes the final reservations, this is an example of a

single-layer quasi-heterarchical system (Figure 3(b)), which also utilises cooperation and bidding type communication mechanisms.

An alternative to holding a bidding process for the operations of a job, one at a time and in their precedence order, is scheduling all of them collectively with a single bid. However, due to the precedence dependencies between operations, this requires more advanced techniques. Kutanoğlu and Wu (1999) developed a single-layer quasi-heterarchical system using combinatorial auctions, which overcomes this difficulty. Jobs are the local agents who prepare bids for discrete time slots on machines. In choosing the time slots for its operations, each order agent tries to minimise the sum of reservation prices it pays and the resulting weighted tardiness cost. At each iteration, the manager agent, who acts as a bidder, updates the prices for the time slots using a subgradient search method for a Lagrangian relaxation of the corresponding job shop scheduling formulation. This continues until a feasible schedule is found in which no two order agents request the same time slot. The pricing scheme directed by the manager agent enhances the cooperation between order agents to minimise the total weighted tardiness objective for the entire system.

Similar to Kutanoğlu and Wu (1999), Wellman and Walsh (2001) make use of auction theory in distributed scheduling. They model the factory as a single machine and the time slots on it as goods to be sold. Each buyer is assumed to have a single job with one operation and a maximum price that he/she is willing to pay for the processing of the job before its deadline. The factory, which acts as a bid manager, has a minimum price that it attaches to each time slot. Under this simplistic setting, the authors present an analytical investigation of the existence of equilibrium prices and the effectiveness of ascending auctions in reaching equilibrium.

In most bidding mechanisms, it is assumed that when a bid for a job is awarded, its operations cannot be shifted any longer. However, Ow *et al.* (1988) propose a system that allows shifts of scheduled operations to prepare better bids for new operations. In this system, a resource agent is associated with each workcentre, which consists of similar resources. The manager agent is responsible for initiating and finalising bids for the operations of a job. Its objective is to minimise the completion time and flowtime of jobs. A resource agent may come up with alternative time slots as bids for an operation, with the objective of minimising the workcentre's production costs and queueing times. After all bids for an operation are received, the completion time implied by the best one is taken as the earliest start time of the succeeding operation. The corresponding resource agents are awarded the individual operations only when the completion time of the job is accepted by the customer. In awarding an operation, the manager agent specifies a slack period within which the operation can be shifted into the future, without violating the committed completion time of the job and without delays on other workcentres.

In a study by Babayan and He (2004), the authors use cooperative game theory for scheduling jobs in a three-stage flexible flowshop. In this system, local agents correspond to jobs. There also exists a manager agent that sets the rules of the game and decides whether rescheduling should be performed at any point. The objective is to minimise the makespan. Cooperation and competition among the agents are enabled through the outer game and the inner game. The outer game determines the agents that can enter the game. These agents are the ones that are eligible to schedule their jobs at a stage of the game. The inner game enables competition among the agents by allowing them to reschedule their jobs for obtaining better solutions. A mixed integer programming formulation is used to schedule the operations of the jobs whose agents are in the game.

In the previous studies, the manager agent acts as a conflict solver or a bid manager who can override the decisions given by local agents in an attempt to improve global performance. There is another group of studies in which the manager agent has a different role. Specifically, it generates the whole schedule to extract information from it for guiding local decisions. This kind of approach may not reduce the complexity of the solution procedure because the whole problem is formulated and/or solved anyway. The idea behind this methodology is that although global performance of the system is a major concern, local schedulers make the final decisions according to disruptions and the current status of the shop floor. This ensures a more up-to-date and valid schedule. We next present some representative papers in this line of research.

Roundy *et al.* (1991) provide a two-module scheduling system in which resource agents make their real-time scheduling decisions by dispatching, using the cost information generated by a global agent (i.e. Planning Module). The system is designed for a job shop where only one machine can process a particular operation. That is, no assignment decision of operations to the resources is made. The overall objective is to minimise the total weighted tardiness. At the top level, the scheduling system works by the global agent finding the Lagrangian dual variables corresponding to the IP formulation of the whole system, over a finite planning horizon (one to four weeks). Taking these values as machine usage prices and considering the tardiness penalties, it then calculates a cost for completing each operation on a machine over time and passes this information to the resource agents. This is repeated periodically or when a major disruption occurs. Whenever a resource becomes available, the local agent selects the next operation using a dispatching rule with respect to a measure that is a function of the cost information. A similar system is also proposed by Morton *et al.* (1988). However, in this study, machine usage prices are calculated using simulation, and the overall objective considers several measures such as minimising earliness costs, inventory holding costs, or tardiness costs.

In another study, Oguz (1998) developed a single-layer quasi-heterarchical system with a separate manager agent. In this system, a central scheduler (manager agent) generates an off-line schedule for the entire system. This solution is then used to guide the dispatching decisions at the machine level. Each machine agent is responsible for the decisions at trigger points (i.e. the arrival of a new job, the completion of an operation, or a machine breakdown) in an online fashion. Oguz (1988) proposes different ways for local schedulers to handle the scheduling task. In the first approach, each machine generates an optimum sequence for all of its unscheduled operations. If the first operation in the sequence is available, it is assigned to the machine. If it is not in the sequence, the operation that has become available before its release time is chosen. If such an operation cannot be found, a predetermined cost function is used to select a job. In the second approach, each machine uses the sequence generated by the central scheduler and dispatches the first operation in the sequence if it is available. Otherwise, it chooses the first available operation in the sequence. Alternatively, the author proposes that local schedulers use the minimum slack algorithm within the boundaries generated by the global schedule or the FIFO rule. Although the original problem is broken down into a number of single machine problems, decomposition does not bring much benefit, in terms of solution complexity, because the global scheduler should generate the entire schedule anyway. But the proposed approach does enable the system to implement the schedule according to the original sequence and to respond to disruptions as quickly as possible.

A recent model proposed by Wang *et al.* (2008) considers a shopfloor that consists of several workcells. Each workcell is assigned a scheduler which we consider as an agent.

It is assumed that an initial allocation of all jobs among workcells has already been made for minimising the weighted sum of makepan. However, the details of this allocation procedure are not discussed explicitly in the paper. The agent associated with each cell can use either a mixed integer programming or a dispatching rule to minimise the makespan for the workcell. If disruptions occur in real time, then the scheduler can request bids from other cells for the operations that can no longer be processed within the cell. Since the initial allocation of jobs to the workcells is made considering the overall objective and by a centralised authority, this system makes use of cooperation besides bidding for coordinating independently made schedules of workcells.

4.4 *Multi-layer quasi-heterarchical systems*

Parunak (1988) was the first to propose a multi-layer quasi-heterarchical system. In this system, the global scheduler prepares a schedule for the entire system for a given time period (i.e. months). Then, each workcell (i.e. department or GT cell) in the lower levels of the hierarchy adds the necessary details to the schedule using a bidding algorithm (i.e. workcells prepare bids for the tasks defined at this higher level). There may be several workcells embedded in a workcell. More than one workcell can also negotiate to share the tasks of a job or the operations of a task. In this system, bidders may be either individual agents (workcells) or a group of agents. The bidding criterion is to meet due dates, while preserving the load balance of the system. Apart from bidding and negotiation, there is also cooperation type communication among the agents because a higher level workcell or the global scheduler makes decisions to optimise the overall system performance.

Burke and Prosser (1991) propose another multi-layer quasi-heterarchical system called the DAS (Distributed Asynchronous Scheduler). In this system, there is a three-level hierarchy of agents through which the scheduling problem is solved. At the lowest level, an agent (O-agent) represents a resource. Similar resources are grouped, and each group is also associated with an agent (T-agent). Operations are assigned to the T-agents by the manager agent (S-agent) at the release of a job. Considering the load balance among its resources, the T-agent delegates the operation to an O-agent. The O-agent uses its local scheduling rule to fit the new operation into its existing schedule so that the due date for the operation is met. If the operation cannot be scheduled, the O-agent informs its superior T-agent with a record of underlying conflicts. The T-agent keeps this information as part of its learning mechanism and uses it in later assignments. After a repetitive application of this process, if the current operation cannot be scheduled by any of the O-agents in the group, the S-agent is informed about the reasons. The S-agent has two mechanisms to solve conflicts: backtracking and due-date relaxation. First, backtracking is utilised to undo earlier operations that were scheduled with relative ease, to reduce conflicts. If the current operation and all effected operations cannot be successfully scheduled owing to backtracking, the S-agent resorts to relaxing the due dates of some jobs. The cooperation among agents enhanced by the learning and messaging mechanism between agents enables the solution of the scheduling problem in a distributed, yet collaborative manner.

Maturana and Norrie (1996) developed a general task planning and coordination architecture for distributed decision making, which they applied to a manufacturing scheduling problem. This architecture exhibits a multi-layer quasi-heterarchical structure with a manager agent (template mediator), resource agents and a hierarchy of order agents (Figure 4). The upper-level order agents (data agent managers) represent the products.

Each data agent manager (DAM) is, in turn, associated with a collection of active mediators. An active mediator (AM) is an order agent that stands for a part of the product. At the beginning of the planning period, for each operation on its part, an AM initiates a bidding mechanism to select alternative resources that can process it. The actual assignment of resources to operations is done in real time. That is, when a resource becomes available, it informs all AMs that associated with it earlier. Considering the current status of the part on the shop floor, each AM computes a starting time for its operations and presents it to the superior DAM. If a DAM receives plans from several active mediators competing for a resource in the same time slot, it selects one plan based on priorities of the parts. The assignment is finalised by the template manager, who collects the decisions from the DAMs. Again, if multiple DAMs produce conflicting plans, the AM makes the selection considering overall system performance, such as the due dates of the products. Clearly, this DS architecture is based on cooperation and on a bidding mechanism.

A related concept to multi-layer quasi-heterarchical systems is holonic manufacturing. A 'holon' is defined as an autonomous and cooperative system entity (Bongaerts *et al*. 1995, 1996, Brussel *et al*. 1999). In its most general form, a holarchy, namely, a hierarchy of holons, is very similar to a multi-layer quasi-heterarchical system (e.g. a holon for a workstation may be composed of other holons that stand for machines). However, the levels in a holarchy, and hence, the degree of decentralisation may change dynamically according to the system needs. Resource holons, order holons and product holons are the *basic holons* for a minimalistic implementation of this concept (Brussel *et al*. 1999). Resource and order holons have a role in decision making, however, a product holon is in charge of carrying information about the products and the processes. Because a holon is not necessarily a decision maker, it does not exactly correspond to an agent. In a typical holonic manufacturing system, there are also *staff holons* that have a global view of the system and optimisation capabilities. Often times, staff holons advise the basic holons, but the system can be configured such that basic holons obey the staff holons. The effectiveness of both this temporary hierarchy and changing levels of decentralisation is worth further study and should be evaluated considering different attributes of distributed scheduling, such as communication mechanisms and schedule generation methods. A review of several manufacturing concepts including holonic, bionic, and fractal manufacturing can be found in Tharumarajah *et al*. (1996).

5. Conclusions and future research paths

Owing to the increasing pressure on companies to be competitive in global markets, coordination between independent parties has become important in all areas, including scheduling. Distributed scheduling is a relatively new concept for solving large-scale scheduling problems, particularly those in which multiple decision makers work for their own benefit. In view of the increasing importance of DS systems, this paper aims to unify the current literature under a new and comprehensive taxonomy and to identify further research issues in the area. Our review suggests several important observations about the current state of research in distributed scheduling. In this section, we discuss these observations and our conclusions in light of the questions (i.e. Q1, Q2, Q3, and Q4) raised earlier in the paper. Based on our observations and conclusions, we also suggest some directions for future research in the area of distributed scheduling.

5.1 *Conclusions on Q1: decentralised versus centralised scheduling systems*

We start reporting our findings about the first question (i.e. Q1), by noting that every distributed scheduling model is based on a decomposition method. Ovacik and Uzsoy (1997) discuss examples of such methods in which the original problem is divided into smaller subproblems and solved in pieces (e.g. machine-based, time-based or job-based decomposition). Decomposition can also be part of a centralised scheduling model. What distinguishes DS systems from centralised models that utilise decomposition is their focus on communication schemes, which enable the subproblems solved on different computers to be combined together to form a final schedule. In fact, given the required computational power, every decentralised model can be run on a centralised processor. But, the thrust of research in distributed scheduling lies more in practical needs. For example, subcontractors of a major company, who may have alternative processing capabilities, would typically prepare their schedules independently from one another, yet in coherence with the whole schedule. In such a highly distributed environment, local decision makers are competitors with individual objectives. Not only would they want to hide their shop-floor scheduling information from each other, but this information might well become obsolete before being updated on a central processor.

In order to fully characterise a distributed scheduling system, we believe the following issues should be considered:

(1) *Decomposition of the problem into subproblems*: some systems are already distributed in nature, which leads to an inherent decomposition of the scheduling problem. Examples include supply chain systems or flexible manufacturing systems in which the subproblems are defined as the scheduling of individual companies or the scheduling of individual cells, respectively. Machine-based decomposition and job-based decomposition are other methods that are common in DS literature.

(2) *Assignment of subproblems to the agents*: an agent is an independent decision maker that has all the information to solve a subproblem and the ability to communicate with other agents. In some systems, there may be alternative agents competing with one another for the same subproblem. In this case, the assignment decision can be made using some form of a bidding scheme (e.g. Shaw 1987), an integer programming formulation (e.g. Chung *et al*. 1996) or a priority-rule based algorithm (Adacher *et al*. 2000).

(3) *Design of algorithms for the agents to solve subproblems*: in a typical distributed structure, agents have their own goals, which may be different from the global system objectives. An algorithm designed for an agent to solve its subproblem should account for the tradeoff function between global and local objectives. How a particular agent uses any system information, is also dependent on this tradeoff function. In pure heterarchical systems, local agents have limited global information. As the hierarchy increases, the amount of global information shared by the local agents increases.

(4) *Communication mechanisms and algorithms to allow agents to integrate their solutions and to solve conflicts*: because the local decision makers prepare their schedules independently from each other, these partial schedules may produce some conflicts. Carefully designed communication mechanisms can eliminate such conflicts. The efficiency of these mechanisms affects both the quality of the resulting solution and the time needed to reach a conflict-free schedule.

5.2 *Conclusions on Q2: critical design decisions in a distributed scheduling system*

The components of a DS system discussed in Section 5.1 suggest important observations about the second fundamental question, Q2. In explicitly reporting these observations about the design factors, we group them according to three perspectives: for the whole system, for an individual agent, and considering inter-agent relations. Accordingly, some of the important design factors include, but are not limited to, the following:

From a system-wide perspective:

(1) how the scheduling problem is decomposed;
(2) who the independent decision makers are;
(3) how the independent decision makers are assigned to subproblems.

For an individual agent:

(4) what system information is prevalent to each agent;
(5) what the tradeoffs are between local and global objectives;
(6) how the local scheduling problem is solved;
(7) how the solution is updated to eliminate conflicts.

Considering inter-agent relations:

(8) what command structure exists between these decision makers;
(9) how the partial schedules are communicated to identify conflicts.

Our survey facilitates an understanding of how these design factors have been incorporated into various DS systems. The brief summary of the papers in Section 4 also provides insights into the answer to question Q3 about the commonalities and the differences of the proposed systems. In the next section, we report some of our general observations on this issue.

5.3 *Conclusions on Q3: commonalities and differences of existing distributed scheduling systems*

The issue of commonalities and differences of existing distributed scheduling systems has in fact been explicitly accounted for in Table 1. We would like to note that, our analysis of the literature in Section 4 and the summary table not only take a DS point of view, but also indicate how these studies fit into the scheduling literature in general. Below, we discuss our inferences from Table 1.

In our classification scheme, 'information flow structure', 'communication type', and 'agent type' are the three attributes unique to distributed scheduling. Table 1 demonstrates that a majority of the studies propose a quasi-heterarchical structure in terms of information flow. We have identified eight papers with a pure heterarchical structure. The use of a quasi-heterarchical structure in most systems appears to be motivated by the fact that the scheduling problem is defined for a single company, e.g. a job shop or a flexible manufacturing system. Obviously, more information sharing and cooperation between agents and a system view of optimisation produce better performance in the overall objectives. If the scheduling problem is limited to a single company, it is only natural to have a certain degree of hierarchy to keep better track of these objectives. By the same reasoning, a heterarchical structure is more useful when the optimisation of local objectives is of higher priority. More often than not, developing a feasible schedule is the

only global objective in these systems. Iterative refinement, negotiation, and bidding are commonly used as communication mechanisms to achieve this objective.

Although the tradeoffs between global and local objectives are essential in designing a DS system, no study in the literature explicitly makes this distinction. In most cases, local objectives are not defined. For this reason, we have only included global objectives in our taxonomy. We also acknowledge that seemingly similar studies may exhibit differences in terms of their user interface and how they are applied in practice.

While the literature has made substantial progress in the area of distributed scheduling, there are many areas for possible investigation. In the next subsection, we briefly summarise our findings related to Q4 and we elaborate on open areas that need further research.

5.4 *Conclusions on Q4: effects of the design aspects on computational time and solution quality*

The analysis of the literature and the results presented in this paper clearly indicate that much of the previous research effort has been spent on designing different systems under various factors related to distributed scheduling. However, a major deficiency in the current literature is that no studies focus on how these design factors affect computational time and solution quality, an issue raised in question Q4. It is worthwhile to note that even though many studies are similar and, in fact, build on the knowledge gained from previous studies, there is lack of a systematic analysis and comparison of how the underlying differences affect performance measures. The work of Veeramani and Wang (1997) is an exception. In this paper, the authors investigate the effects of several design factors, including the number of agents, the decision time, and the message size on the communication-related performance measures in a bidding-based system. These performance measures are the number of bidding processes completed per unit time and the time taken to complete a bidding process. This work is important in its recognition of our last question of interest and should be expanded to consider other design factors and performance measures.

We also note that most of the proposed distributed scheduling systems have not been adequately tested either in a simulated environment or in a real manufacturing environment. Nor have they been compared with centralised scheduling systems. In general, the strengths and weaknesses stated in many papers have yet to be studied. We are aware of only two papers that partially address these issues (Ottaway and Burns 2000, Dewan and Joshi 2001). Ottaway and Burns (2000) show that introducing hierarchical control on a prototype FMS as the system load increases, results in higher performance in terms of throughput, resource utilisation, and in-process inventory. Dewan and Joshi (2001) compare a bidding-based distributed-scheduling model using a network of four processors versus using a single processor. They investigate communication delays and processor utilisation under varying values of the problem parameters. As a result, they identify instances in which distributed scheduling outperforms centralised scheduling in terms of computational measures, and thereby, they quantify the degree of benefits provided by distributed scheduling.

As discussed earlier, distributed scheduling can be considered as a subset of the literature on multi-agent systems. A special issue of the journal *Production Planning and Control* gives emphasis to research and industrial experiences of multi-agent system applications to the various domains of production planning and control (Caridi *et al.* 2004). The special issue presents studies on a wide range of application areas including

customer order planning, production scheduling, diagnosis and control, production layout planning. A general review of the multi-agent systems is made available by the editors as an introduction to the special issue (Caridi and Cavalieri 2004). In this review, the maturity degree of several multi-agent system applications is discussed. It is suggested that among the models proposed in the literature, the ones that have been translated into a commercial product are few. The authors also identify that most of the applications are at an emulated stage. They conclude that even the most mature models, which are on scheduling and monitoring, cannot be used in the form of a commercial on-the-shelf software. They attribute this to the reason, which we also agree, that there is no clear understanding of how a multi-agent system can provide better results than a traditional model. Therefore, within the specific context of distributed scheduling, we consider comparing distributed scheduling models to the centralised alternatives as an evident research area.

It is important to note that, in comparing distributed and centralised scheduling systems, expected long-run costs should be considered as a measure in addition to the classical performance criteria used in scheduling theory. The investment and maintenance costs of the resources for computing and monitoring in a distributed scheduling system may be larger than they are for a centralised system. Therefore, these costs should be weighed against the potential benefits of a distributed system.

A final research topic concerns the tradeoffs between optimising local and global objectives. As Internet technologies become more widespread in business, scheduling of competing entities with individual objectives gains more importance. Auction theory, which has been widely used in economics to model the negotiations between a seller and a group of buyers, has recently become popular in analysing the conduct of trade in online markets (Wellman and Walsh 2001, Reeves *et al*. 2005). This approach can play a significant role in increasing our knowledge of how certain design factors jointly affect local and global objectives.

References

Adacher, L., Agnetis, A., and Meloni, C., 2000. Autonomous agents architectures and algorithms in flexible manufacturing systems. *IIE Transactions*, 32 (10), 941–951.

Agarwal, R., De, P., and Wells, C.E., 1995. Cooperative distributed problem solving: an investigation in the domain of job shop scheduling. In *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*, 3–6 January, Kihei, Hawaii.

Babayan, A. and He, D., 2004. Solving the n-job 3-stage flexible flowshop scheduling problem using an agent-based approach. *International Journal of Production Research*, 42 (4), 777–799.

Basnet, C. and Mize, J.H., 1994. Scheduling and control of flexible manufacturing systems: a critical review. *International Journal of Computer Integrated Manufacturing*, 7 (6), 340–355.

Biemans, F.P. and Vissers, C.A., 1991. A systems theoretic view of computer integrated manufacturing. *International Journal of Production Research*, 29 (5), 947–966.

Bongaerts L., *et al*., 1995. Schedule execution for a holonic shop floor control system. In *Proceedings of the ASI-95 of NOE on ICIMS* (Advanced Summer Institute 1995 of the Network of Excellence in Intelligent Control and Integrated Manufacturing Systems), 24–26 June, Lisboa, pp. 115–124.

Bongaerts L., *et al*., 1996. Identification of manufacturing holons. In *Proceedings of the European Workshop for Agent-Oriented Systems in Manufacturing*, 27 September, Berlin, pp. 57–73.

Brussel, H.V., *et al*., 1999. A conceptual framework for holonic manufacturing: identification of manufacturing holons. *Journal of Manufacturing Systems*, 18 (1), 35–52.

Burke, P. and Prosser, P., 1991. A distributed asynchronous system for predictive and reactive scheduling. *International Journal for Artificial Intelligence in Engineering*, 6 (3), 106–124.

Caridi, M. and Cavalieri, S., 2004. Multi-agent systems in production planning and control: an overview. *Production Planning and Control*, 15 (2), 106–118.

Caridi, M., Garetti, M., and Cavalieri, S., 2004. Editorial. *Production Planning and Control*, 15 (2), 103–105.

Cheng, T.C.E. and Sin, C.C.S., 1990. A state-of-the-art review of parallel machine scheduling research. *European Journal of Operational Research*, 47 (3), 271–292.

Chiu, C. and Yih, Y., 1995. A learning-based methodology for dynamic scheduling in distributed manufacturing systems. *International Journal of Production Research*, 33 (11), 3217–3232.

Chiussi, F.M. and Francini, A., 2000. A distributed scheduling architecture for scalable packet switches. *IEEE Journal on Selected Areas in Communication*, 18 (12), 2665–2683.

Chung, D.-Y., *et al*., 1996. Developing a shop floor scheduling and control software for an FMS. *Computers and Industrial Engineering*, 30 (3), 557–568.

Crowe, T.J. and Stahlman, E.J., 1995. A proposed structure for distributed shopfloor control. *Integrated Manufacturing Systems*, 6 (6), 31–36.

Dewan, P. and Joshi, S., 2000. Dynamic single-machine scheduling under distributed decision-making. *International Journal of Production Research*, 38 (16), 3759–3777.

Dewan, P. and Joshi, S., 2001. Implementation of an auction-based distributed scheduling model for a dynamic job shop environment. *International Journal of Computer Integrated Manufacturing*, 14 (5), 446–456.

Dewan, P. and Joshi, S., 2002. Auction-based distributed scheduling in a dynamic job shop environment. *International Journal of Production Research*, 40 (5), 1173–1191.

Dilts, D.M., Boyd, N.P., and Whorms, H.H., 1991. The evolution of control architectures for automated manufacturing systems. *Journal of Manufacturing Systems*, 10 (1), 79–93.

Duffie, N.A. and Prabhu, V.V., 1994. Real-time distributed scheduling of heterarchical manufacturing systems. *Journal of Manufacturing Systems*, 13 (2), 94–107.

Duffie, N.A. and Prabhu, V.V., 1996. Heterarchical control of highly distributed manufacturing systems. *International Journal of Computer Integrated Manufacturing*, 9 (4), 270–281.

Giret, A. and Botti, V., 2004. Holons and agents. *Journal of Intelligent Manufacturing*, 15 (5), 645–659.

Graves, S.C., 1981. A review of production scheduling. *Operations Research*, 29 (4), 645–675.

Hadavi, K., *et al*., 1992. An architecture for real-time distributed scheduling, *In*: A.F. Famili, *et al*., eds. *Artificial Intelligence Applications in Manufacturing*. Cambridge USA: AAAI Press, 215–234.

Hohlt, B., Doherty, L., and Brewer, E., 2004. Flexible power scheduling for sensor networks. In *Proceedings of the IEEE and ACM International Symposium on Information Processing in Sensor Networks*, pp. 205–214.

Jeong, I.-J. and Leon, V.J., 2002. Decision-making and cooperative interaction via coupling agents in organizationally distributed systems. *IIE Transactions*, 34 (9), 789–802.

Jeong, I.-J. and Leon, V.J., 2005. A single-machine distributed scheduling methodology using cooperative interaction via coupling agents. *IIE Transactions*, 37 (2), 137–152.

Kouiss, K., Pierreval, H., and Mebarki, N., 1997. Using multi-agent architecture in FMS for dynamic scheduling. *Journal of Intelligent Manufacturing*, 8 (1), 41–47.

Kutanoğlu, E. and Wu, S.D., 1999. On combinatorial auction and Lagrangean relaxation for distributed resource scheduling. *IIE Transactions*, 31 (9), 813–826.

Lau, J.S.K., *et al*., 2005a. Distributed project scheduling with information sharing in supply chains: part I – an agent-based negotiation model. *International Journal of Production Research*, 22 (15), 4813–4838.

Lau, J.S.K., *et al*., 2005b. Distributed project scheduling with information sharing in supply chains: part II—theoretical analysis and computational study. *International Journal of Production Research*, 23 (1), 4899–4927.

Leung, J.Y.-T., Ed., 2004. *Handbook of Scheduling*: *Algorithms*, *Models*, *and Performance Analysis*. Boca Raton, FL: Chapman and Hall/CRC.

Lima, R.M., Sousa, R.M., and Martins, P.J., 2006. Distributed production planning and control agent-based system. *International Journal of Production Research*, 44 (18–19), 3693–3709.

Lin, G.Y. and Solberg, J.J., 1992. Integrated shop floor control using autonomous agents. *IIE Transactions*, 24 (3), 57–71.

Liu, J. and Sycara, K.P., 1993. Distributed constraint satisfaction through constraint partition and coordinated reaction. *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, May, Hidden Valley, PA.

Liu, J. and Sycara, K.P., 1994. Distributed problem solving through coordination in a society of agents. *Proceedings of the 13th International Workshop on Distributed Artificial Intelligence*, July, Seattle, WA.

Liu, J. and Sycara, K.P., 1995. Exploiting problem structure for distributed constraint optimization. *Proceedings of the First International Conference on Multiagent Systems*, June, San Francisco, California.

Liu, N., Abdelrahman, M.A., and Ramaswamy, S., 2007. A complete multiagent framework for robust and adaptable dynamic job shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 37 (5), 904–916.

Maturana, F.P. and Norrie, D.H., 1996. Multi-agent mediator architecture for distributed manufacturing. *Journal of Intelligent Manufacturing*, 7, 257–270.

Morton, T., *et al*., 1988. SCHED-STAR: a price based shop scheduling module. *Journal of Manufacturing and Operations Management*, 1, 131–181.

Oguz, S., 1998. Object oriented design of a distributed scheduling system. Unpublished Master Thesis, Bosphorous University, Turkey.

Ottaway, T.A. and Burns, J.R., 2000. An adaptive production control system utilizing agent technology. *International Journal of Production Research*, 38 (4), 721–737.

Ovacik, I.M. and Uzsoy, R., 1997. *Decomposition methods for complex factory scheduling*. Massachusetts: Kluwer Academic Publishers.

Ow, P.S., Smith, S.F., and Howie, R., 1988. A cooperative scheduling system, *In*: M.D. Oliff, ed. *Expert Systems and Intelligent Manufacturing*. North Holland: Elsevier Science Publishing Co., 70–89.

Parunak, H.V.D., 1987. Manufacturing experience with the contract Net, *In*: M.N. Huhns, ed. *Distributed Artificial Intelligence*. Vol. I, Los Altos, CA: Morgan Kaufmann, 285–310.

Parunak, H.V.D., 1988. Distributed artificial intelligence systems, *In*: A. Kusiak, ed. *Artificial Intelligence: Implications for CIM,* IFS Conferences. Bedford, UK: Springer-Verlag, 223–251.

Pinedo, M., 2002. *Scheduling: theory, algorithms, and systems*. Englewood Cliffs, N.J.: Prentice Hall.

Rahimifard, S. and Newman, S.T., 1998. Reference architectures for team based distributed production planning and control. Eureka–Factory: Project No 1629, January.

Reeves, D.M., *et al*., 2005. Exploring bidding strategies for market-based scheduling. *Decision Support Systems*, 39, 67–85.

Rothkopf, M.H., Peke, A., and Harstad, R.M., 1998. Computationally manageable combinatorial auctions. *Management Science*, 44 (8), 1131–1147.

Roundy, R.O., *et al*., 1991. A price-directed approach to real-time scheduling of manufacturing operations. *IIE Transactions*, 23, 149–160.

Sabuncuoglu, I., 1998. Scheduling with neural networks: a review of the literature and new research directions. *Production Planning*, 9 (1), 2–12.

Sen, S., 1997. Multiagent systems: milestones and new horizons. *Trends in Cognitive Sciences*, 1 (9), 334–340.

Shaw, M.J., 1987. A distributed scheduling method for computer integrated manufacturing: the use of local area networks in cellular systems. *International Journal of Production Research*, 25 (9), 1285–1303.

Shaw, M.J. and Whinston, A.B., 1988. A distributed knowledge-based approach to flexible automation: the Contract Net framework. *International Journal of Flexible Manufacturing Systems*, 1 (1), 85–104.

Shen, W., 2001. Agent-based cooperative manufacturing scheduling: an overview. *COVE Newsletter*, 2. Available online at: http://www.uninova.pt/~cove/newsletter.htm/2/Shen.pdf (accessed March 19, 2009).

Shen, W., 2002. Distributed manufacturing scheduling using intelligent agents. *IEEE Intelligent Systems*, 17 (1), 88–94.

Shen, W., Wang, L., and Hao, Q., 2006. Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 36 (4), 563–577.

Sikora, R. and Shaw, M.J., 1998. A multi-agent framework for the coordination and integration of information systems. *Management Science*, 44 (11), 65–78.

Silver, E.A., Pyke, D.F., and Peterson, R., 1998. *Inventory management and production planning and scheduling*. 3rd ed. New York: John Wiley.

Smith, R.G., 1980. The Contract Net protocol: high-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29 (12), 1040–1113.

Smith, R.G. and Davis, R., 1981. Frameworks for cooperation in distributed problem solving. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11 (1), 61–70.

Sycara, K.P., *et al*., 1991. Resource allocation in distributed factory control. *IEEE Expert*, 6 (1), 29–40.

Tharumarajah, A., Wells, A.J., and Nemes, L., 1996. Comparison of the bionic, fractal and holonic manufacturing system concepts. *International Journal of Computer Integrated Manufacturing*, 9 (3), 217–226.

Tharumarajah, A., 2001. Survey of resource allocation methods for distributed manufacturing systems. *Production Planning and Control*, 12 (1), 58–68.

Tsay, A., Nahmias, S., and Agrawal, N., 2000. Modeling supply chain contracts: a review, *In*: S. Tayur, R. Ganeshan and M. Magazine, eds. *Quantitative Models For Supply Chain Management*. Norwell, MA: Kluwer Academic Publishers, 299–330.

Vaidya, N., *et al*., 2005. Distributed fair scheduling in a wireless LAN. *IEEE Transactions on Mobile Computing*, 4 (6), 616–629.

Veeramani, D. and Wang, K.J., 1997. Performance analysis of auction-based distributed shop-floor control schemes from the perspective of the communication system. *International Journal of Flexible Manufacturing Systems*, 9 (2), 121–143.

Wang, H., Liao, S., and Liao, L., 2002. Modeling constraint-based negotiation agents. *Decision Support Systems*, 33 (2), 201–217.

Wang, C., Ghenniwa, H., and Shen, W., 2008. Real time distributed shop floor scheduling using an agent-based service-oriented architecture. *International Journal of Production Research*, 46 (9), 2433–2452.

Wang, C., Ghenniwa, H., and Shen, W., 2009. Constraint-based winner determination for auction-based scheduling. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39 (3), 609–618.

Wellman, M.P. and Walsh, W.E., 2001. Auction protocols for decentralized scheduling. *Games and Economic Behavior*, 35, 271–303.

Yang, E.H., Barash, M.M., and Upton, D.M., 1993. Accommodation of priority parts in a distributed computer-controlled manufacturing system with aggregate bidding schemes. *Proceedings of the 2nd Industrial Engineering Research Conference Proceedings*, pp. 827–831.

Zhang T., *et al*., 1998. Multi-agent techniques in holonic manufacturing systems, in *Proceedings of the 2nd International Symposium on Intelligent Manufacturing Systems*, 7–8 August, Sakarya, Turkey.