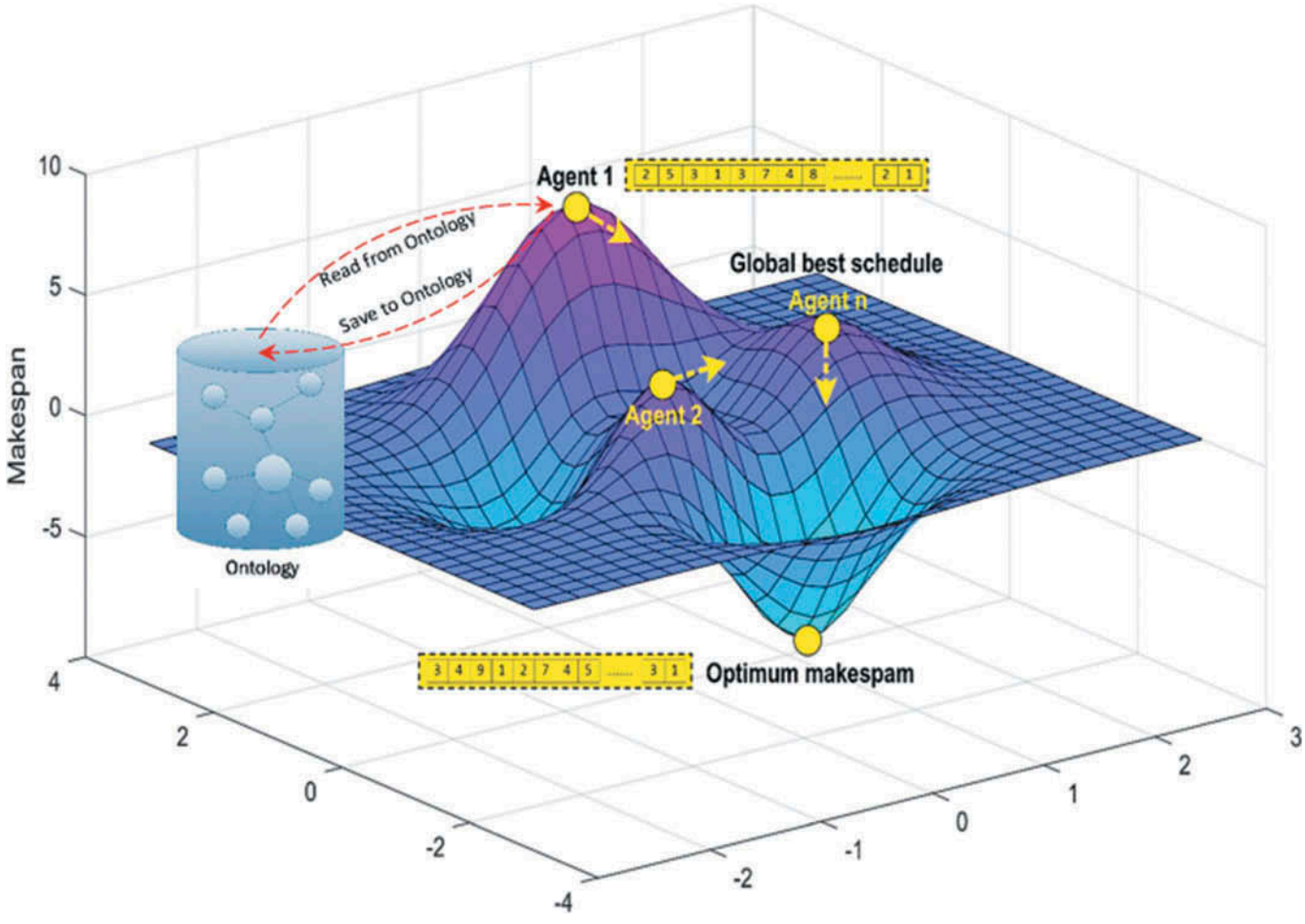
cogent engineering



PRODUCTION & MANUFACTURING | RESEARCH ARTICLE

Agent-based distributed manufacturing scheduling: an ontological approach

Salman Saeidlou, Mozafar Saadat, Ebrahim Amini Sharifi and Guiovanni D. Jules

Cogent Engineering (2019), 6: 1565630

Saeidlou et al., *Cogent Engineering* (2019), 6: 1565630
https://doi.org/10.1080/23311916.2019.1565630

cogent engineering

PRODUCTION & MANUFACTURING | RESEARCH ARTICLE

Agent-based distributed manufacturing scheduling: an ontological approach

Salman Saeidlou[1]*, Mozafar Saadat[1], Ebrahim Amini Sharifi[2] and Guiovanni D. Jules[1]

Received: 08 October 2018
Accepted: 01 January 2019
First Published: 08 January 2019

*Corresponding author: Salman Saeidlou, School of Mechanical Engineering, University of Birmingham, Edgbaston, B15 2TT, Birmingham, UK
E-mail: sxs989@bham.ac.uk

Reviewing editor:
Tao Peng, Zhejiang University, China

Additional information is available at the end of the article

Abstract: The purpose of this paper is the need for self-sequencing operation plans in autonomous agents. These allow resolution of combinatorial optimisation of a global schedule, which consists of the fixed process plan jobs and which requires operations offered by manufacturers. The proposed agent-based approach was adapted from the bio-inspired metaheuristic- particle swarm optimisation (PSO), where agents move towards the schedule with the best global makespan. The research has achieved a novel ontology-based optimisation algorithm to allow agents to schedule operations whilst cutting down on the duration of the computational analysis, as well as improving the performance extensibility amongst others. The novelty of the research is evidenced in the development of a synchronised data sharing system allowing better decision-making resources with intrinsic manufacturing intelligence. The multi-agent platform is built upon the Java Agent Development Environment (JADE) framework. The operation research case studies were used as benchmarks for the evaluation of the proposed model. The presented approach not only showed a practical use case of a decentralised manufacturing system, but also demonstrated near optimal makespans compared to the operational research benchmarks.

Subjects: Artificial Intelligence; Automation; Cybernetics; Evolutionary Computing; Simulation & Modeling; IndustrialEngineering & Manufacturing; Operations Research; ProductionSystems; Manufacturing & Processing; Manufacturing Engineering

Keywords: multi-agent systems; knowledge –based systems; ontology; distributed systems; metaheuristics

Salman Saeidlou

ABOUT THE AUTHORS

Salman Saeidlou received the MEng (Hons.) degree in Mechanical Engineering from the University of Birmingham, Birmingham, U.K., and the Ph.D. degree in the same field of study from the University of Birmingham, Birmingham, U.K. He is currently working within the Automation and Intelligent Manufacturing (AIM) Research Group in the School of Mechanical Engineering. His research interests include holonic manufacturing, distributed systems, agent-based modelling and intelligent manufacturing systems.

PUBLIC INTEREST STATEMENT

The desirable characteristics of production plants have been identified to be the high levels of productivity and flexibility. The stated characteristics are essential for the manufactuing firms to be able to compete with other companies within today's modern economy. Optimum scheduling algorithms are crusial means of achieving the efficiency required to reach the optimum levels of productivity and flexibility. In this research, an intelligent manufacturing scheduling model is presented using the concepts of ontology, multi agent system and metaheuristic algorithms.

1. Introduction

Manufacturing companies need to be very efficient in modern competitive economies. Consequently, production plants currently have a requirement for greater flexibility and productivity in order to be able to compete. Optimal scheduling algorithms are also important (Nguyen, Mei, & Zhang, 2017), and scheduling in a manufacturing context refers to the allocation of resources to plants in a way that will minimize the time taken to produce products.

A common problem in manufacturing scheduling is the job shop scheduling problem (JSSP), which provides an obstacle to increasing the efficiency of a manufacturing plant and is evidenced in numerous manufacturing industries, where resources need to be efficiently assigned (Zhao, Gao, Chen, & Guo, 2018). Job shop scheduling (JSS) refers to a set of jobs which need to be processed by a set of machines, and where each job consists of a chain of operations to be processed during a specific time period on a given machine (Jules & Saadat, 2017). Each job is composed of a set of partially ordered operations, where each operation has a deterministic processing time and pre-assigned materials that need to be minimised subject to certain constraints (Jules & Saadat, 2017). The constraints can relate to material quantity, job prioritisation and the capacity of the operational centres in addition to deadlines (Vallikavungal Devassia, Salazar-Aguilar, & Boyer, 2018).

This can be explained as being a non-polynomial (NP) time complete combinatorial optimisation problem, where combinatorial optimisation problems are seen as having a finite number of feasible solutions that are difficult to solve in real-time (Gen, Zhang, & Hao, 2017). The JSSP is NP hard; its algorithmic complexity increases exponentially with the number of operators and the number of machines defining the JSSP, dynamic nature and practical interest for industrial applications. This means that unless P = NP, there exists no polynomial time algorithm that finds the optimal solution of a given JSSP instance (Gen et al., 2017).

A large number of algorithms have been developed in order to solve general scheduling problems, including the JSSP (Hasan, Sarker, Essam, & Cornforth, 2009; Martin et al., 2016). These include the genetic algorithm, particle swarm optimisation, simulated annealing, ant colony optimisation, artificial bee colony, and bee colony optimisation (Hasan et al., 2009; Martin et al., 2016; Vallikavungal Devassia et al., 2018).

This paper proposes an approach to a combined, shared knowledge base, which leads to the development of a decentralised manufacturing network. It is an attempt to help small and medium manufacturing enterprises (SMEs) achieve an informal nature of leadership amongst their firm by sharing knowledge and resources (Müller-Seitz & Sydow, 2012). In order to end the decision- making in agreement, a shared knowledge base is required to act as a medium in which the members of a firm can share their complementary data (Artto, Kulvik, Poskela, & Turkulainen, 2011; Li, Veliyath, & Tan, 2013).

The manufacturing industry is now more focused on open organisational structures, decentralised operations and market predictability. This has indeed led to the development of intelligent virtual systems within a software architecture also known as a multi-agent system. This distributed artificial intelligence allows the representation of complex and dynamic real-life systems with the help of its internal agents. These perceive their environments and act accordingly with respect to their nature of computational entities (Komma, Jain, & Mehta, 2011).

In this context, simple algorithms are developed for each existing agent, which can also enable new agents to be placed in the software network. This multi-agent system's framework can easily identify and comprehend its environment and act accordingly by semantically enriching any data and metadata (Guo & Zhang, 2009; Luck, McBurney, & Preist, 2003). This data will need to be produced and managed independently. Ontology, as a well-known knowledge base, will openly signify structures and enable them to hold the domain data to allow automated queries, reasoning and coalition between the agents with a common communication language and mutual area of

focus (Guo & Zhang, 2009; Lin, Zhang, Lou, Chu, & Cai, 2011; Monostori, Váncza, & Kumara, 2006; Saeidlou, Saadat, Amini Sharifi, & Jules, 2017).

The proposed network paradigm is a combination of ontology and a multi-agent system, designed to prevail over each system by optimising scheduling objectives on its own and to form an advanced innovatory decentralised scheduling system.

The aim of this paper is to explore the development of algorithms for the distributed scheduling of manufacturing operations. Hence, a decentralised scheduling system was developed that enables intelligent agents to interact with one another in order to find an optimum solution. The structure and interaction protocol of the agents are adapted from the classical particle swarm optimisation (PSO) with some modifications. For the scope of this study a number of job shop scheduling instances were used; the ABZ6 is shown in Table 1 for illustration. These problem instances are the basis for research evaluation and validation within the operational research community (Lawrence, 1984). When the fidelity of the proposed model is validated, extensive computational experiments will be carried out in different benchmark instances as future work. Section 2 of this paper provides a literature review of related research. Section 3 delivers detailed methodology used for this research. Section 4 presents the results and discussion of the proposed framework and finally Section 5 gives the concluding remarks.

2. Related work

2.1. Classical job shop scheduling solutions

There are two key approaches to solving the JSSP. One is the centralised approach, wherein a centralised computing unit performs the scheduling. Alternatively, a decentralised approach, such as the multi-agent approach, can be used. Various techniques, such as branch and bound, bottleneck-based heuristics and the disjunctive graph have previously been used to solve the JSSP (Shen, Dauzère-Pérès, & Neufeld, 2018). The branch and bound technique involves the use of a dynamic tree structure, which signifies the resolution space of different achievable arrangements (Gabel, 2009). Using this method, candidate solutions are frequently calculated and checked against upper and lower limits; they are discarded if they are outside the range, in order to ensure that the best suboptimal solution is achieved.

Another common method is the disjunctive method, first proposed by Roy and Sussmann in 1964 (Gabel, 2009). It can be used to formulate JSSP in addition to representing schedules. It is a useful model that can describe instances of the JSSP. Disjunctive graphs model a system of tasks to be scheduled and show their respective timing constraints. This method is efficient when the objective function is regular, as there will always be an optimal solution for the problem represented.

2.2. Multi-agent approach to job shop scheduling

As the centralised approach requires a powerful, central computing facility to handle the huge amount of data, it cannot react to machine failures (Wu, 2005). Moreover, it uses simplified theoretical models to allocate machines/resources to the jobs (Gabel, 2009). However, the multi-agent system can react to data adaptively. This is helpful in order to improve the stability of the system and increase its robustness and scalability. In this context, robustness suggests that the performance of the schedule can overcome disruptions, while adaptable systems are capable of reacting to numerous unpredicted disruptions. This is important because decisions can be based on a shorter planning horizon and on limited problem knowledge (Yoo & Müller, 2002).

In contrast to centralised schedulers, an agent-based manufacturing scheduling system supports distributed scheduling, wherein each agent is assigned to one machine. An agent could be one production plant; for example, each steel production process was represented by one agent which was responsible for scheduling (Cowling, Ouelhadj, & Petrovic, 2003). It has long been known that distributed solution approaches can be utilised in finding optimal or near-optimal solutions for manufacturing

industry problems because of the nature of modern-day manufacturing (Wu, 2005). Modern-day manufacturing requires many plants, each of which is responsible for a certain task, and distributed scheduling enables communication between these plants in order for an optimal solution to be obtained. This connection is obtained through communication between the different agents.

In general, a multi-agent system is a useful alternative to the restrictions imposed by centralised scheduling schemes, as they are more autonomous and can react to real-time situations (Wang & Liu, 2006). Moreover, different scheduling models and methods can be used to solve the original optimisation problem, whereas their overall architecture helps with flexibility, scalability and fault tolerance (Wang & Liu, 2006).

2.3. Agent-based modelling in manufacturing

The well-known multi-agent system ADACOR, built on the Java development framework (JADE) reconfiguration infrastructure, follows the principle of reaction according to the effects of the previous action which took place in the system. This stigmergy is set by Barbosa, Leitão, Adam, and Trentesaux (2015) to identify the query stream in the environment so as to signal a deviation of plan and provide an opportunity for self-reassessment. In our research, however, the use of message track is not desirable since a knowledge base is a rather more scalable approach.

The development of an agent-based simulator within JADE, with a knowledge base framework capable of reasoning, was presented by Komma et al. (2011) for the manufacturing shop floor domain modelling agents, namely AGV-agent, machine agent and part agent. The part-agents' dispatch algorithm in this work was produced on a first come first served basis and lacked focus on tasks with sequence set-up limitations.

Whilst our proposed research focuses on operation sequencing and timing within a manufacturing network, Vrba and Marik (2010) have developed a multi-agent system to comply with changes of the factory floor layout. The system will reassess the virtual map and search for the nearest product destination using a disturbance simulated as a failed conveyor within the system of conveyors in the factory.

2.4. Distributed scheduling

Agent-based manufacturing scheduling has dealt with numerous issues, and these have been the centre of focus for the research community. Independent projects, which have mutual sets of resources, are limited in operation prioritisation and resource availability and face conflicts regarding the shared resources. Adhau, Mittal, and Mittal (2012) introduced a multi-agent architecture system, which allows project agents to bid with resource agents for time spans using negotiation algorithms called virtual schedule and utility calculations, bid generation and modification through five steps of auctioning. The assessment of the final resource allocation and winner identification is carried out by the exchange agent with the aim of minimising delay as measured in average project delay and total makespan, compared to a 140 multi-project instances' set of available data.

The existing scepticism in the context of disturbance will negatively affect the solutions, leaving them valueless and impracticable. Harjunkoski et al. (2014) introduced a method to allow frequent rescheduling by using deterministic closed-loop scheduling. This avoids unreasonable time spent on random aimless scheduling; since the required time spent on deterministic scheduling is more efficient, allowing a longer scheduling horizon at a time uncertainty, and resulting in quicker reactions to the alterations.

Researchers have also been concerned about the case of distributed scheduling, such as scheduling where a factory layout is set with a dynamic nature consisting of mobile robots. This dynamic nature is classified within flexible manufacturing, where the effective capacity alternates. This issue has been dealt with by a two-level scheduling algorithm introduced by Giordani, Lujak, and Martinelli (2013) comprising iterative auctioning in the initial phase and

the Hungarian method in the final process. This allows the tasks to match up to a robot, following which, robots will then randomly approach different tasks. Unlike the centralised approach, the downside of this method concerns the control policy being substandard and minimal, with resources being misspent. When the resources are insufficient, a favourable partner must be found in a more cost-effective manner, as compared to the conventional tendering process. This has been achieved by a method consisting of a list of buyers and suppliers along with a practicability analyser to co-operate with a negotiation protocol. This produces a decision matrix considering the feasibility with respect to the material flow resulting from the network power and the precedence of the required qualities. The proposed method introduced by Mohebbi and Shafaei (2012) has surpassed the conventional tendering process considering the cost assessment of backlog order, misspent capacity and total cost.

With the increasing rate of dependencies, the resulting exponential increase in the complication of each factory job allocation and unexpected incidents such as amendments of orders and lack of machinery systems, the necessity of an iterative cost adjustment method is irresistible. Alvarez (2007) has resolved this in the form of an auctioning process followed by contract net protocol. This provides the foundation of negotiating time-critical constraints in order to be utilised for bid analyses models produced by each agent; which will create a new foresight with large-scale decision-making.

Finally, in order to save the nature-like quality of the agent technology, concerning the state of the art, a potential field control architecture is set to act as a vector of resource services. An indirect relation between the field intensity and the distance of resources away from the specified job will be analysed to be applied to a matrix of potential fields to form a decisional node. This node will be used as the reference point to identify the resource with the highest potential field for the job. Leitão, Barbosa, and Trentesaux (2012) have successfully surpassed the contract net protocol by achieving an average production time gain of 10%. Hopefully this will help to maintain a distributed nature for the agent technology alongside the applied improvements, allowing self-configuration and self-optimisation to multi- agent systems.

2.5. Distributed agents' interaction protocols

Each agent has partial information that can be used to solve the global optimisation problem, and through the use of a communication protocol, these agents can collaborate in order to obtain a global optimisation solution. Most agent-based manufacturing scheduling systems use negotiation protocols for resource allocation (for example, a contract net protocol or an auction protocol) (Wu, 2005). An agent communication language (ACL) is a language that forms the basis for communication between the different software agents at each manufacturing plant. The agents are connected through a local network (for example, the Ethernet). In separate research, a hybrid network for agents was presented with four types of agents (Wang & Liu, 2006). These four agents were: the resource agent (RA), which represented a resource such as a device, a tool and an automated guided vehicle (AGV); the task agent (TA), which represented a part or a part group; and the task management agent (TMA) and resource management agent (RMA), which were responsible for managing and monitoring the TAs and RAs in the system respectively. The task agent and the resource agent were equipped with a certain degree of autonomy to reason and make decisions for themselves. Additionally, they were equipped with the knowledge to independently solve a part of the global problem. Such a framework is both flexible and adaptive, as agents may be replaced or added freely. The system configuration can be continuously changed to accommodate changing requirements.

The negotiation mechanism is a direct approach to assign scheduling activities, which falls in either market-based or threshold-based methods. The latter depends on whether the agent accepts the proposed task in the event of a variety of events taking place, and it is fed with knowledge in the form of pheromones. The market-based method is specified for the agents with self-seeking objectives, who compete to be rewarded for favourable system-wide actions and provide the required

knowledge. The agents function with respect to a certain range of implicit and explicit data and rules. The advantage of the threshold-based method in comparison to the market-based approach is that it avoids the effects of a continuous change of bids, which leaves the indirect threshold approach free from communication scalability complexities (Goldingay & Van Mourik, 2013; Shen, 2002). The other component which acts alongside the negotiation mechanism is the interaction protocol, which manages the time and structure of exchanging data between each agent (Jules, Saadat, & Saeidlou, 2015; Owliya, Saadat, Jules, Goharian, & Anane, 2013).

2.6. Centralised versus decentralised scheduling system

A number of researchers have compared the multi-agent approach to centralised scheduling approaches for solving the job shop scheduling problem. In related research, the authors compared the performance of operational research algorithms to the multi-agent distributional algorithms (Frey, Nimis, Wörn, & Lockemann, 2003). A simulation-based benchmarking scenario was developed in order to perform this comparison. The results showed that the disruption duration was much lower than what the operational research algorithm produced as long as vacant capacity was available at the production unit. The same was observed for the standard deviation of the throughput time.

It was also shown that when disturbances occur, the multi-agent approach produces better results by means of shorter waiting queues. This was an expected result due to the adaptability inherent in the multi-agent framework. Moreover, it was shown that the centralised approach is unable to regulate disturbances from the original schedule and does not perform as well as the multi-agent approach in complex environments. Similar results were obtained by Aydin and Fogarty (2004a) and Wong, Leung, Mak, and Fung (2006). Constraints for the successful implementation of multi-agent scheduling were also investigated by Frey et al. (2003). Additionally, multi-agent system robustness was investigated and was shown to perform better than the operational research algorithm (Frey et al., 2003).

2.7. Manufacturing ontology

Ontology is defined as a data model which provides information about a domain, with the aid of relationships established among various concepts under that particular domain (Gruber, 1993; Guarino, Oberle, & Staab, 2009). Organizational knowledge is encapsulated to build a set of knowledge bases that can combine details such as those of rules, policies, processes, definitions and relationships. Ontology has been applied in many different areas and domains, including manufacturing. Simple to complicated processes are easily explained by ontology in the manufacturing domain; this makes it easier to create patterns, algorithms and frameworks for the automation of any process.

Ontology represents a specified, formal way to comprehensively share a complex concept encapsulating a broad set of terminologies for cross-domain integration and collaboration. Formal ontology incorporates complex semantics of relative concepts, their controlling rules, constraints, values, attributes and properties (Schalkoff, 2011). An ontology representation language is a set of categorically defined rules and in-depth semantics; one such example is the web ontology language (OWL).

A five-stage methodological approach is widely accepted as the design process of ontology, originally proposed by Uschold and Gruninger (1996):

- Purpose and scope identification
- Conceptualization and knowledge acquisition
- Reconstruction and integration of multiple relating ontologies
- Specifications
- Documentation and evaluation

Ontology presents a formal tool for bridging the gap between interdisciplinary and interdisciplinary collaboration and knowledge sharing, which plays a significant role in multiple fields of application. According to Kulon, Broomhead, and Mynors (2006), a lack of ontology creates multiple bottlenecks for every repetition of an engineering task, emanating from impediments in cross-disciplinary data sharing, and consequently considerably slowing down the repetitive processes. In such circumstances, the delays are multiplied with each design loop, badly affecting the cyclic processes. They demonstrated that a lack of ontology results in designers using individualistic approaches and terminologies to document a product design, which makes it difficult for other designers to reconstruct and implement it. Ontology, as a categorical design of a conceptualization, plays an instrumental role in collaborative infrastructures through integrating applications, sharing cross-field information, enabling interoperability, and reconstructing knowledge. More elaboration on the development of a manufacturing ontology and its components is presented by Jules, Saadat, and Li (2013).

A comprehensive review of the literature sheds light on the most recent approaches in the use of ontology and multi-agent system in manufacturing domain. One approach includes developing an agent based model and constructing production schedule, whilste disregarding the interaction protocols and knowledge interoperability. Alternative studies were designed to address the manufacturing domain specific problems using semantic web technology and ontology, not taking into account the multi-agent scheduling. This research aims to address this knowledge gap by the employment of an ontological approach of multi-agent manufacturing scheduling.

For the scope of this research, a manufacturing ontology is built with Protégé ontology editor software, which is a free and open-source leading ontological engineering tool. Protégé has relevant advantages over the other platforms. It provides a graphical user interface to develop the ontology that requires minimum expertie and time to be built. It provides an interface with other knowledge-based tools such as Java Expert System Shell (JESS) and is compatible with various ontology languages and formats such as eXtensible Markup Language (XML), DARPA Agent Markup Language (DAML) and Ontology Inference Layer (OIL) (Gaševic, Djuric, & Devedžic, 2009). In addition, it is particularly suitable for this research as it has deductive classifiers for validation of the ontology consistency. Furthermore, it can also export into other formats such as RDF, which is the basis of this research. When modelling in a domain, developers must be able to focus more on the concepts and relations rather than the syntax of the final results. This can be achieved through a Protégé-based editor, resulting in modelling at a conceptual level (Gennari et al., 2003; Noy et al., 2001).

2.8. Manufacturing domain data formalisation

In order for knowledge storage and inline analytics of scheduling data, this paper proposes the use of open source ontology editor protégé. The idea was lit by Kotulski, Sędziwy, and Strug (2014), who set the system of graph transformation for the agents who share knowledge. This approach has been improved to increase the pace of graph processing and to maintain the cohesion of the graph by using algorithms, which helped with the handling of the increased workload applied by the multi-agent systems.

The negative effects on the environment of moving urban goods demonstrates the need for an improved system of decision-making. This is in order to achieve efficient and environmentally friendly transportation of goods. Anand, van Duin, and Tavasszy (2014) achieved this by forming a model considering the stakeholder agents and their interactions, which works along with a knowledge base that illustrates the city logistics domain.

2.9. Collaboration mechanism in agent-based model

In order to avoid the communication overhead in manufacturing control systems, ontology is proposed in this paper as the main source of knowledge, considering its scalability quality as the result of its ability to store a different range of signals. Wang et al. (2012) on the other hand, have approached this by introducing a pheromone-based co-ordination system that allows the agents

to select signals that can act on them. This is done by providing a potential field which emits a variety of fields in terms of strength.

Another fundamental difference in the proposed method of Wang and Choi (2014) as compared to this project is the decomposition-based holonic approach (DBHA) to scheduling. This consists of a genetic algorithm control (GAC) and a minimum process time contract net protocol (SPT CNP), and it is possible to switch between them. Considering the makespan produced by GAC in low stochastic conditions, SPT CNP prefers a high stochastic process time. Therefore, an estimation of the switch-over threshold was made using a back-propagation network with respect to a flexible flow shop problem with stochastic processing time (Wang & Choi, 2014). This project on the other hand, presents a method which uses fixed processing times in disturbance situations, which is of a lower standard than the DBHA method.

The novelty of this research is evidenced in the development of an ontology-based decentralised manufacturing scheduling allowing better decision making resources with intrinsic manufacturing intelligence. Through a rigorous review of relevant literature in this paper, it was found out that most of the research studies on decentralised manufacturing scheduling are focused on the development of the overall architecture of multi-agent systems, the type of agents, and the priority rules during scheduling. However, by investigating the interaction protocols between agents and the interoperability of data through ontology, this work was able to fill a gap in current research.

3. Methodology

3.1. Problem definition

This paper concerns job shop scheduling for instances with multiple routings' associations. Job shop scheduling is the problem associated with the scheduling of n jobs on a set of m machines, where individual jobs contain equal operations' numbers to be processed via unique machine routes. The JSSP may be formalised as an instance of a problem P = (M, O, J) in the scheduling of a job shop, which is made of: a set of machines (M); a set of operations (O) each having an association with a machine over a given duration of time; a set of jobs (J) with one occurrence ability for each operation; a schedule (S) for P which gives an assignment to all operations; and a starting time T(o), where T(o) is greater or equal to 0. The processing time for each operation is P (o) = T (o) + d (o) with some precedent constraint (Bai & Tang, 2013). The overall aim is to minimise the total completion time of all jobs; this is also called the makespan Cmax. The mathematical parameters for the JSSPs are defined as follows:

n	total number of jobs
m	total number of machines (manufacturers)
o_{ji}	operation i_{th} required by job j
t_i	processing time of operation
c_j	completion time of job j
C_{max}	completion time of last operation of all jobs

Table 1 shows the original process plan, where in each row (each job) the sequence of operations is fixed and the order of operations cannot be changed. In contrast, when all operations presented in Table 1 are rearranged based on the machine number, the outcome would be the sequence of operations, which could be processed in any order for each row (each machine). In other words, each job must be processed in the predefined order, while each machine or manufacturer is free to process its operations in any preferable order that satisfies its own objective. This notion is illustrated in Table 2.

Various techniques and approaches to solving these problems have been devised previously with many studies directed along this line (Owiti, Omulo, Okelo-Odongo, & Manderick, 2014). This paper

Table 1. Job process plans for ABZ6 10 × 10 problem

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
J1	17 (62)	18 (24)	15 (25)	13 (84)	14 (47)	16 (38)	12 (82)	10 (93)	19 (24)	11 (66)
J2	25 (47)	22 (97)	28 (92)	29 (22)	21 (93)	24 (29)	27 (56)	23 (80)	20 (78)	26 (67)
J3	31 (45)	37 (46)	36 (22)	32 (26)	39 (38)	30 (69)	34 (40)	33 (33)	38 (75)	35 (96)
J4	44 (85)	48 (76)	45 (68)	49 (88)	43 (36)	46 (75)	42 (56)	41 (35)	40 (77)	47 (85)
J5	58 (60)	59 (20)	57 (25)	53 (63)	54 (81)	50 (52)	51 (30)	55 (98)	56 (54)	52 (86)
J6	63 (87)	69 (73)	65 (51)	62 (95)	64 (65)	61 (86)	66 (22)	68 (58)	60 (80)	67 (65)
J7	75 (81)	72 (53)	77 (57)	76 (71)	79 (81)	70 (43)	74 (26)	78 (54)	73 (58)	71 (69)
J8	84 (20)	86 (86)	85 (21)	88 (79)	89 (62)	82 (34)	80 (27)	81 (81)	87 (30)	83 (46)
J9	99 (68)	96 (66)	95 (98)	98 (86)	97 (66)	90 (56)	93 (82)	91 (95)	94 (47)	92 (78)
J10	100 (30)	103 (50)	107 (34)	102 (58)	101 (77)	105 (34)	108 (84)	104 (40)	109 (46)	106 (44)

P1 = first process; 17 = needed by Job 1, provided by Manufacturer 7; (62) = operation processing time

Table 2. Machine process plans for ABZ6 10 × 10 problem

	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
M0										
M1	31 (45)	91 (95)	101 (77)	61 (86)	21 (93)	51 (30)	41 (35)	81 (81)	11 (66)	71 (69)
M2										
M3										
M4										
M5										
M6										
M7	17 (62)	37 (46)	77 (57)	57 (25)	107 (34)	97 (66)	27 (56)	67 (65)	87 (30)	47 (85)
M8										
M9										

O1 = first operation, 17 = needed by Job 1, provided by Manufacturer 7, (62) = operation processing time

will mainly focus on the multi-agent solution approach to the JSSP. The multi-agent system will be a system that is comprised of agents who are autonomous entities with the ability to cooperate with each other in order to fulfil a common goal. Using a multi-agent system will enable the decentralised scheduling of tasks within the manufacturing supply chain. Case studies of scheduling problems were used to test the proposed scheduling algorithm.

3.2. Solution representation

The proposed hybrid agent-based architecture consists of two main functions for solving JSSPs, namely: a multi-agent function, which is adapted from classical PSO with some modifications, and a genetic algorithm (GA) function for further solution optimisation. When solving JSSP, the coding mechanism and solution representation are extremely important in regard to the efficiency and complexity of the platform. In general, PSO is suitable for linear and continuous search spaces, whereas and in contrast, JSSP is a discontinuous problem with distinctive solutions. Therefore, an encoding method was needed that could transform the feasible schedules into manageable arrays of solutions both for the PSO and GA.

For solutions of n jobs and m machines the JSSP can be represented with an array of n x m real numbers. These arrays are similar to chromosomes in GA, albeit they can be used for PSO operators. In this method, the solution space is represented with an n x m array. For instance, the ABZ6, which is a 10 × 10 problem, has solution schedules which are arrays with 100 cells. Each job from Table 1 appears 10 times in the solution array. For instance, number 4 appears in the array 10 times which represents job 4. Each occurring number 4 in the array represents the corresponding operation for that job in turn. For example, the first number 4 represents the first operation of job 4, then the second number 4 in the array denotes the second operation of job 4, and this will continue 10 times to fulfil all operations of that job number.

As explained previously, the sequence of operations is important and should be done with respect to the predefined process plan. For instance, job 2 must start with operation 25 and then operation 22 and then operation 28 and so on. Operations 25 and 22 are predecessors for operation 28 and, therefore, operation 28 is not allowed to start before operations 22 and 25 are finished. The unique and simple aforementioned technique for solution representation will efficiently satisfy this sequence constraint. Figure 1 shows an example solution to a 3 × 3 problem.

In the above example, 3 means job number 3, and the first number 3 represents the first operation of job 3. Consequently, the second 3 means the second operation of job 3, and the third number 3 means the last operation of job 3. The same logic is applied to the rest of the jobs.

3.3. Evolutionary algorithms for combinatorial optimisation

The interaction protocol for the agent's behaviours is adapted from the particle swarm optimisation (PSO) algorithm for this research. The PSO is a computational search technique, which can be used for optimisation of continuous linear problems (Lin et al., 2010). It was developed in 1995 by Eberhart and Kennedy.

Central to the working of the PSO is the concept of swarming theory, in which organisms normally considered simple and not very intelligent individually, like ants, bees or termites, are able to group together or "swarm" to collectively perform complex functions towards a common goal. This goal might be gathering food or building sophisticated nest structures and filling them with food (Bonabeau, Corne, Knowles, & Poli, 2010). In doing this, they demonstrate behaviour and function as a "swarm", which is well above the ability of any individual organism within the group. These self-organised and decentralised problem-solving systems result in "swarm intelligence" used to solve the complex problems that the organisms encounter daily. It provides flexibility and efficiency in adapting quickly to changes in the environment during the process of problem solving (Deepa & Senthilkumar, 2016).

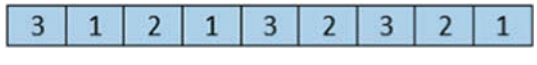


Figure 1. Sample solution array for a 3 × 3 job shop scheduling problem.

The PSO is based on a simulation of this social behaviour of individual agents in a swarm, such as a bird flock; they adapt to their environment using a combination of their own individual experience and the benefits of collective knowledge and information gathered from the swarm as a whole. This social sharing provides an evolutionary advantage for solving complex problems (Kennedy & Eberhart, 1995), such as an individual fish in a school swimming away from a threat not directly observed. Similar to this, the PSO algorithm is modelled towards the behaviour of a bird flock searching for food. The best condition is arrived at by the swarm through communication with members who have better information and others are informed so that they all move simultaneously towards the food (Rini, Shamsuddin, & Yuhaniz, 2011). This process is repeated continuously until optimal conditions are reached or the food is found.

The algorithm starts by generating a random set of potential solutions (particles) which can fly through the search space, just like a bird looking for optimal conditions and concentrating the search around beneficial areas (Lin et al., 2010). Each swarm particle *sp* iterates an optimum solution search based on its own best experience, global best experience of the swarm, and using newly generated velocity. This is in addition to its previous position to determine the position change, as shown in the equations below (Lin et al., 2010):

Assuming X_{sp} is the position of the *spth* particle:

$$X_{sp} = X_{sp} + V_{sp} \tag{1}$$

$$V_{sp} = \omega \times V_{sp} + C_1 \times Rand() \times \left(X_{sp}^{best} - X_{sp}\right) + C_2 \times Rand() \times \left(X_{sg}^{best} - X_{sp}\right) \tag{2}$$

Where X_{sp} is the moving distance of the *spth* particle in one iteration; ω is defined as the inertial weight for one step movement distance for a particle; C_1 is the self-learning factor; C_2 is the social learning factor; X_{sp}^{best} represents the personal best position of the *spth* particle; and X_{sg}^{best} indicates the best global position.

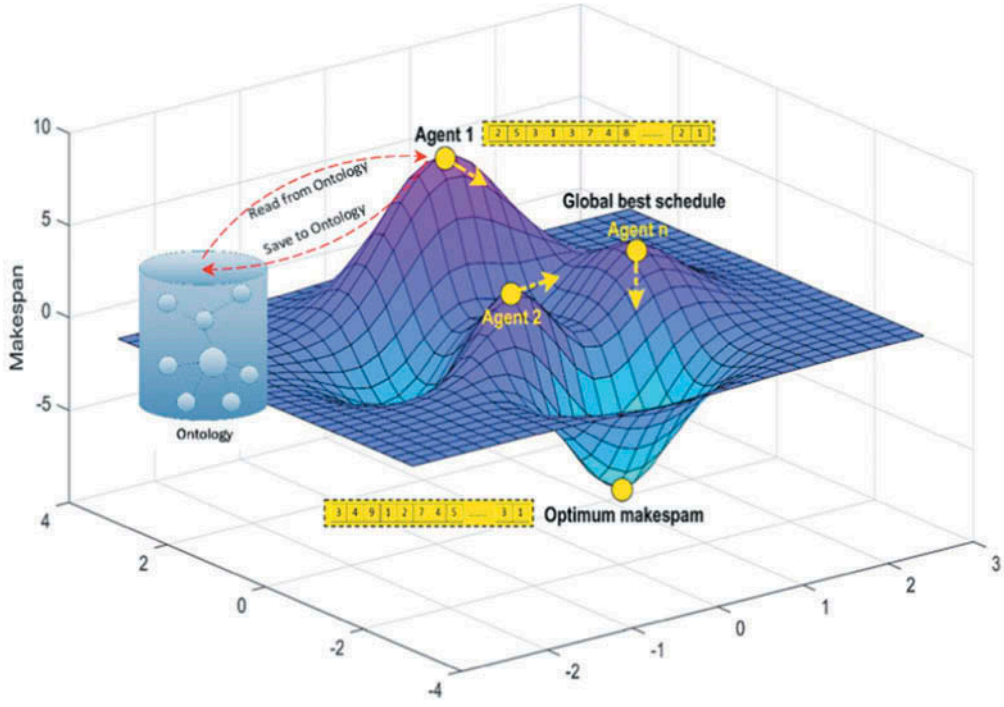
3.4. Proposed decentralised scheduling architecture

The proposed multi-agent based system consists of two main functions and an ontology at its foundation. During the initial phase, a set of agents are created and randomly distributed across the search space. Each one of these agents has a schedule solution represented by an array, as explained in the previous section. After the initialisation phase, all agents evaluate their fitness values (makespan) in order to determine the *lbest* (local best, the best makespan so far for each agent) and the *gbest* (global best, the minimum makespan) in each iteration. The agents then try to move towards the best solution by updating their position and velocity as the method is adapted from the PSO.

One key factor to consider is that the PSO is originally developed for linear and continuous problems. However, JSSP is a problem with discrete instances. Therefore, by mapping the schedule solutions into the proposed arrays and parsing them into individual agents, the system is modified and is compatible to perform the PSO's procedure on a discontinuous problem.

An ontology with scheduling concepts has been developed to cater for agents' communications. Agents extract machine and job information from the ontology for their reasoning and evaluation and also assert their operation plans into the ontology for data sharing. This knowledge representation facilitates automated data query, data extraction and agents' collaboration by holding domain information in a logic-based language. The proposed architecture is illustrated in Figure 2 below.

Figure 2. Proposed multi-agent system architecture.



As shown in Figure 2, all agents are scattered in the search space at the initial phase. During each iteration, all agents evaluate their fitness functions to find their best makespan found so far. This is called the local best makespan for each agent. After the execution of each iteration, all agents share their knowledge about the local best makespan through the common ontology, and the shortest makespan amongst all agents will be chosen as the global best makespan. As the global best solution is chosen, all agents move towards the global best and try to perform their fitness function evaluation again. This process continues until the optimum makespan is found, such as that presented at the bottom of Figure 2. The process can be terminated by the number of iterations or by a time-elapsed threshold. As the JSSP consists of jobs and their respective start and finish time, the actual scheduling solution (Gantt Chart) can be shown in two dimensions. However, the three dimensional surface in Figure 2 is used to help in the illustration of the movement of agents towards the optimum solution.

The velocity of the agents is defined as the rate of change of permutation. As an example, if the solution in Figure 1 represents the global best for a problem instance, all agents will try to make themselves similar to that solution arrangement. For the ABZ6, where we have a 10×10 problem, the velocity will range from 1 to 100. A low velocity such as 1 will ask the moving agent towards the *gbest* to change the first cell of its array to the similar cell from the global best. In the case of Figure 1, the moving agent changes its first cell to number 3. The same logic, velocity 2 means changing the first cell to number 3 and the second cell to number 1. As the agents move towards the best solution, the velocity increases and that means imitating more cells like that in the best global position.

For the purpose of this research, there are two possible settings for the movements of the agents towards the best solution:

(1) Gradual approach of all agents towards the optimum solution, where position and velocity changes with respect to the *gbest* and velocity increases as agents get closer to the optimum.

(2) Having zero velocity, where in each iteration all agents jump to the position of the best-known agent and then try to perform a GA to optimise the solutions and to find the new *gbest*.

The GA operator performs optimisation on both settings. The only difference is that in the latter case, when all agents are generated randomly and the best known is calculated, all agents locate themselves in the position of the best-known solution and then they try to perform the GA individually. During the GA optimisation process, if an agent finds a better makespan, it will ask all agents to move to that position with a thread in the software, otherwise, it will go back to the previous mode and try to perform the GA on the previous generation. It is worth mentioning that agents are not moving in the space in the same manner as the classical PSO. When we are saying that agents are moving towards a better solution, it means that they try to change their arrays and make them more similar to the better solutions. In this instance, actual movement does not take place. Figure 3 shows the flow diagram of the sequence of operations.

3.5. Ontology for intelligent manufacturing scheduling

The literature on the subject shows a trend towards the provision of developing globally accepted ontology based on semantics' agreements, in order to align the terminologies for interoperable applications. This is described as being the backbone for an integrated manufacturing system. The ontology gives the appropriate meaning to stored data through constructing databases attached to standard terminologies. These can then be utilised to automate the process and create intelligent self-regulating systems. Such a semantics' web is used for connectivity of machines working in a facility using the standard protocols.

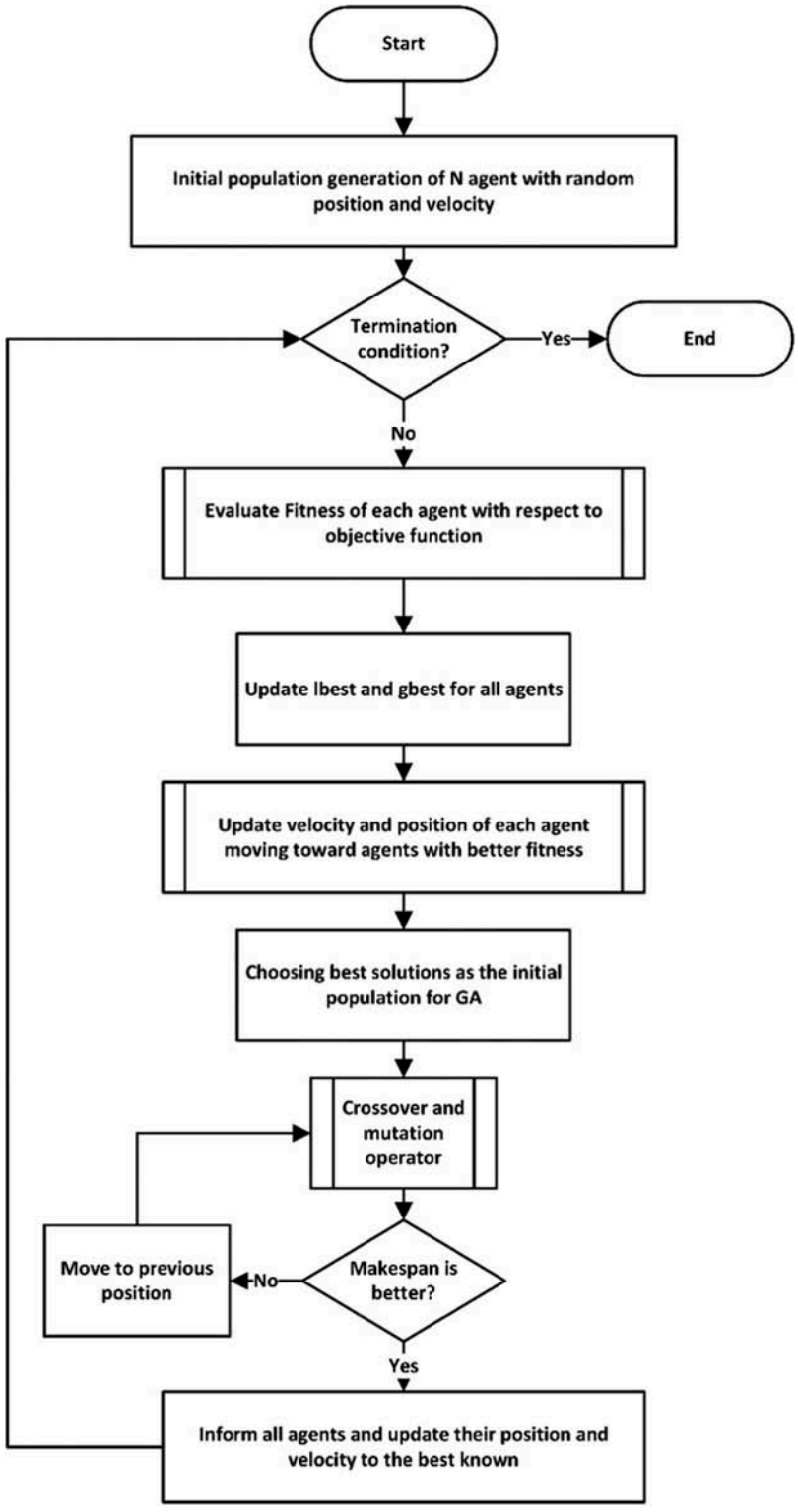
Knowledge management includes three main components, i.e. acquisition, presentation and examination. Knowledge management in the manufacturing ontology domain deals with the rules and ontology defined for interoperability. After acquisition of knowledge regarding a product's design and manufacture through multiple sources of human induction and literature, the defined ontology is used to present the acquired knowledge in a standard manner. Ontology Web Language (OWL)/Resource Description Framework (RDF) coding standards are used by the experts to deploy knowledge bases in Semantic Web Rule Language (SWRL). The creation and maintenance of ontologies is carried out through additional tools such as Protégé. Researchers have used Protégé to further create APIs (application programming interfaces) based on Java.

C.J. van Aart from the University of Amsterdam has introduced a novel concept, namely the Bean Generator, in order to implement a system which is capable of transforming the "manual method" of creating ontology definition classes. These include schemas, along with predicates, concept classes and agent actions with an "automated method" (Bellifemine, Caire, & Greenwood, 2007). The Bean Generator minimises the time required to produce the Java files, showing an ontology compatible with the JADE toolkit. It does this by acting as a plug-in for Protégé, whilst producing JADE compliant ontologies and Protégé projects, along with the import and export ability of RDF and RDFS.

Using ontologies to store data enables data re-usability and maintenance, which is a massive advantage as compared to normal databases; it also provides the opportunity to merge it with available ontologies for knowledge enhancement. This ontology alignment gives structure that is essential for compatibility with domain specific applications. This research made use of "OWLSimpleJADEAbstractOntology.owl" in order to generate FIPA compliant ontology for the multi-agent platform JADE, which is compiled using the beangenerator tool. The "swrla.owl" and "sqwrl.owl" ontologies enable realisation of the knowledge base through the use of Semantic Web Rule Language. Figure 4 shows the main structure of the aforementioned ontologies.

Under the structure of the existing ontologies, the proposed ontology is developed. First, the sub-classes of the class "Beangenerator: Concept" are identified using a top-down strategy. These

Figure 3. Flow diagram of main functions.



abstract concepts are identified as *Products*, which contain product information such as process plans; *Resources*, which include properties such as machines; *Orders*, which contain order details such as quantity or due time; *Beangenerator:AID* which represents Agent ID and indicates the

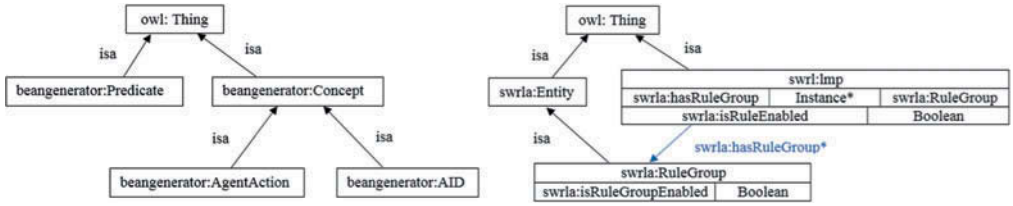


Figure 4. Bean generator OWL ontology and SWRL ontology.

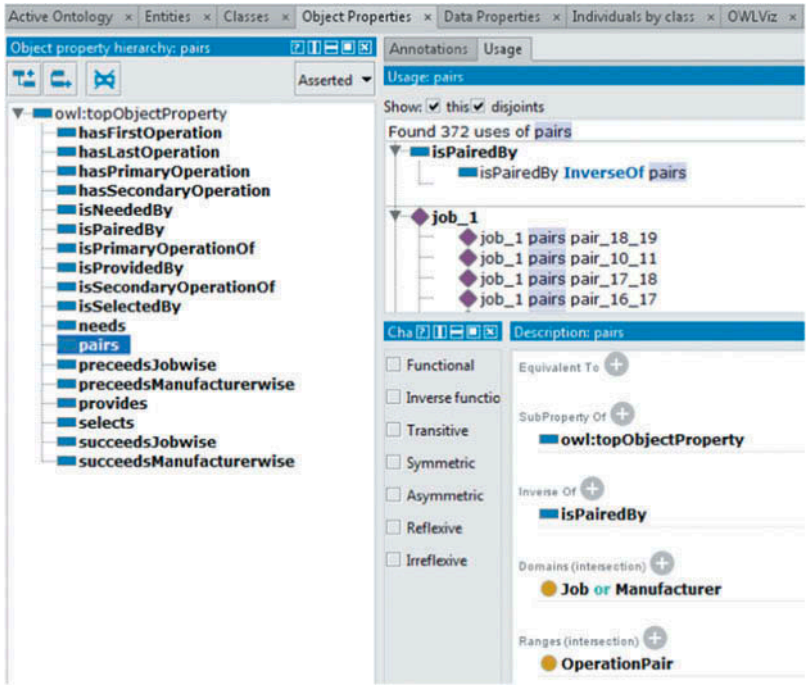


Figure 5. Object properties the ontology that describes the schedule of various jobs.

agent's details, such as its name and location in the network; and *Beangenerator:AgentAction*, which relates to the type of actions executed by agents in order to change the state of the environment. Then the main concepts are specialised into more specific concepts making sure that the ontology is not overspecialised.

The proposed ontology is developed in a systematic and scalable way in order to be re-usable for further development and implementation in a real case study of a manufacturing network. For the scope of this paper, only the operation research case studies are considered. However, scheduling data from the GFM Srl Company is extracted to further realise the application of the proposed ontology-based system on a real industrial case study for further research. GFM is an Italian company, who manage the production of gas and steam turbine parts for the power generator industry. Figures 5 and 6 illustrates the main object and data properties of the proposed ontology.

4. Results and discussions

The experimental results were obtained by executing the proposed multi-agent system platform based on 10 runs for each case study instance. The number of agents was set at 30 with reference to the work done by Lin et al. (2010) to enable precise comparison of the outcome data, while the number of iterations was set at 100 and 5000 for PSO and GA respectively. The results obtained from the proposed model were compared with benchmark instances from the operation research library (Lawrence, 1984). In order to evaluate the quality of the proposed model, two separate

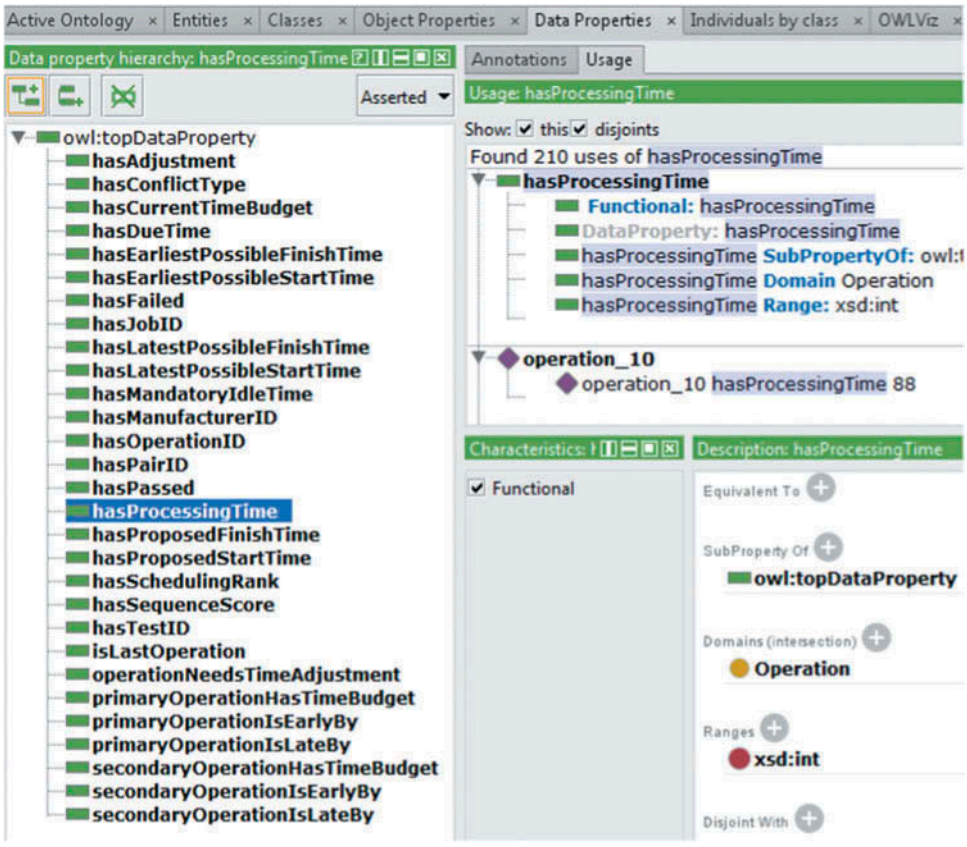


Figure 6. Data properties of the ontology that describes the schedule of various jobs.

algorithms were chosen from the literature to act as the benchmark. The first algorithm is extracted from the research on the use of a GA for solving JSSP (Hasan et al., 2009), and the second algorithm uses PSO for the same application (Lin et al., 2010).

Based on the extracted algorithms, three different models were developed and compared. The first model is designed to solve JSSP by implementing a GA as the only algorithm. The second model is designed to solve the same problem, but only using PSO; and the third model is the proposed hybrid model. Ten runs for each model were executed and the results were depicted as the average of all these tests. The optimum solution for the ABZ5 instance from the OR library is found to be 1234. The PSO solver achieved 1424.4; while the results for the GA and hybrid model were 1296.8 and 1240.5 respectively.

For the ABZ6 case study, the optimum solution was found to be 943 according to the literature. Accordingly, the PSO algorithm showed 1088.9; the GA model got 988.8; and the hybrid model achieved 947.9 units of time. Although the results from the hybrid agent-based system are not the same as the best known from the literature, the proposed model obtained near optimum solutions and it also outperforms individual GA and PSO models. The performance of the PSO over the GA shows that the PSO algorithm, which is mainly developed to solve continuous problems, cannot perform as efficiently as the GA for solving JSSP; which is a combinatorial problem with discrete solution space. The aim of this paper is not to optimise the PSO algorithm, but to demonstrate the feasibility of implementing multi-agent systems for intelligent decentralised scheduling. However, there are relevant studies on the improvement of PSO for JSSP in the literature (Lin et al., 2010) which can be further investigated. The comparison results for these three models are illustrated in Figure 7.

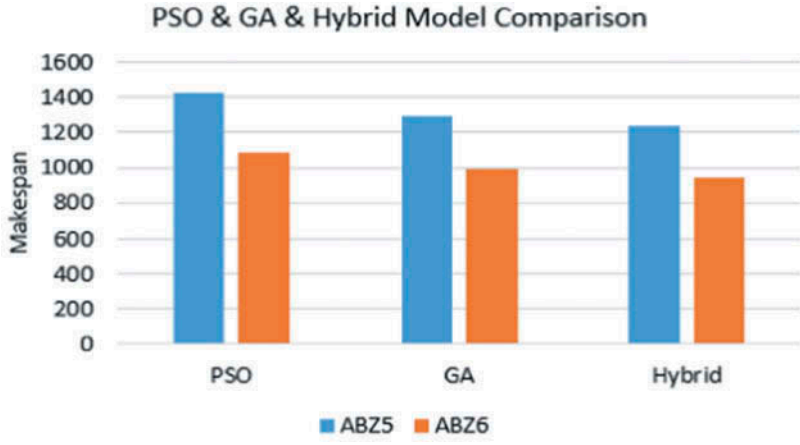


Figure 7. Achieved makespan for PSO, GA and hybrid models.

Furthermore, results from the proposed model are compared to some relevant benchmarks from the literature in order to evaluate the quality of our work. For this reason, results for ABZ5, ABZ6, FT10, LA16—LA20 and ORB1 case studies were extracted from the work done by Aydin and Fogarty (2004b), Meloni, Pacciarelli, and Pranzo (2004) and also from Kurdi (2017). In the first research study, the authors have considered several algorithms to solve JSSP. This includes SA and CSA models based on simulated annealing, TS, CTS and NT based on taboo search, HC and CHC based on hill climbing, GA based on genetic algorithms and ATeam's model based on the collaboration of distributed agents. In the second research study, the work focused on three heuristic algorithms, namely avoid maximum current completion time (AMCC); select most critical pair (SMCP); and select max sum pair (SMSP). And finally in the third study, an improved island model memetic algorithm (IIMMA) was investigated. The comparison of the results are tabulated below in Table 3.

The output data of the proposed model benchmarked and compared to the literature referenced above, illustrates a noticeable improvement in several cases, whilst competing with some of the other data within close proximity.

Figure 8 demonstrates how the use of multi agents leads to the convergence of solutions to near optimum as time elapses. As the number of iterations increases, more solution spaces are investigated by the agents and they move towards the best makespan. Looking at the rate of convergence, it is visible that agents are randomly scattered within the search space and they tend to approach the best solution. From Figure 8 it is evident that the use of a hybrid optimisation algorithm prevented a premature convergence. Consequently, as the simulation reaches its final generations, a slight improvement rate is achieved.

The effect of swapping operations within a schedule on the improvement of the makespan can be investigated thoroughly by reviewing two adjacent solutions from the software. For this purpose, the generated solutions for a random agent are extracted and the results are illustrated in Figure 9. If the two consequent solutions are considered from Figure 9, it is noticeable that by swapping four operations, the total makespan is reduced by 13 units of time. To be more precise, by changing the 1st operation of job 3 with the 4th operation of job 2, and also by changing the 8th operation of job 9 with the last operation of job 4, the makespan has changed from 1255 to 1242. This is demonstrated in Figure 9, which shows the contribution of each agent in the overall optimisation.

Table 3. Result of the proposed model versus existing optimal solutions

	Proposed Model	SA	CSA	NT	CHC	GA	ATeams	AMCC	SMSP	SMCP	IIMMA
ABZ5	1240.5	1274.25	1234	1241.60	1263	1254.33	1240.33	1346	1402	1541	-
ABZ6	947.9	980	946	945.00	948	958.33	946.33	985	978	1253	-
FT10	983.7	1041.33	954.74	967.20	1036.33	998.33	963.40	985	1013	1369	952.10
LA16	954.3	998.33	946.33	967.67	988	988.33	954.5	979	1060	1144	953.20
LA17	793.75	821.67	784	784.33	808.33	794	784.75	800	892	1041	784.50
LA18	855.50	904	848	857	880	874	855.75	916	976	1127	854.5
LA19	845.70	878.67	847	853	878.75	865.67	842.5	846	932	991	845.90
LA20	908	954.33	910	904.33	948.67	922.33	905.75	930	964	1053	909.30
ORB1	1087.33	1194.33	1102	1128.5	1167.33	1157	1112	1213	1264	1444	1094.20

cogent engineering

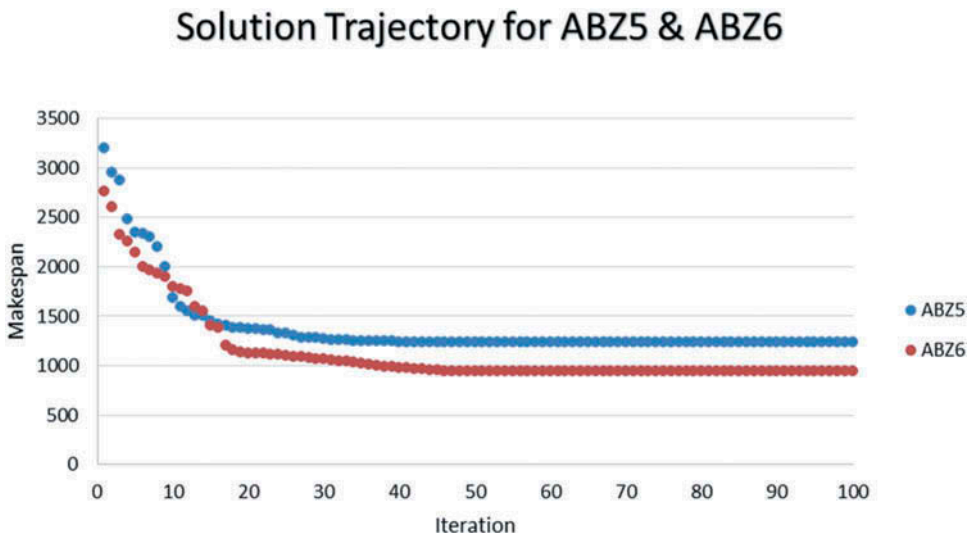


Figure 8. Convergence of the solution over 100 iterations for ABZ5 and ABZ6.

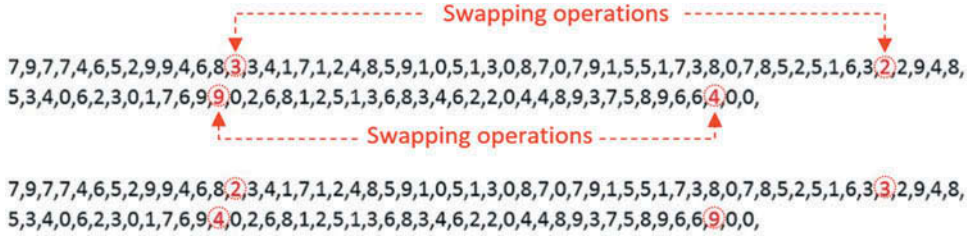


Figure 9. Generated solutions for an individual agent and effect of swapping 4 operations within one agent's solutions.

5. Conclusion

The aim of this research paper is to develop a multi-agent system for distributed manufacturing scheduling while the agents' communications are facilitated by an ontology. The physical dispersing nature of job scheduling is fundamental due to increased competition as manufacturing companies expand and witness an increase in suppliers, customers and partners. This growth is not matched by job shop scheduling in the manufacturing industry, which experiences a number of theoretical and practical problems. This research proposed that this could be resolved by a decentralised system allowing each individual company to decide and schedule plans with respect to their suppliers and sub-suppliers.

In this paper, a decentralised multi-agent system for the scheduling of manufacturing operations was presented in a JADE environment, where a MAS and GA are incorporated for searching the solution space. This combination improved the search diversification and optimised convergence. The use of simple algorithms, combined with distributed agents and an ontology as their core communication mechanism, leads to an intelligent manufacturing system. The underlying intelligence in such systems enables agents to sequence their own operation and acts autonomously while cooperating to achieve a common goal.

The development and implementation of an intelligent scheduling system has been investigated, which provides flexibility and scalability through the use of heterogeneous agents. Results obtained near optimum solutions when compared to published benchmarks for a number of case studies.

There is room for research in the combining of machine learning with multi-agent systems, as by "teaching" the machines how to react in certain situations, this could further enhance their

performance and bring the optimal solution closer. Additionally, heuristics can be combined with the multi-agent approach in order to obtain solutions that are close to the optimal. These concepts could also be applied to the flexible job shop scheduling problem, where customised goods are developed at the manufacturing plants. They are useful because the development of customised goods requires a high degree of adaptability. Finally, machine learning or heuristics in combination with multi-agent systems will add further adaptability and flexibility to the manufacturing process.

Acknowledgements
The authors would like to thank GFM S.r.l. for supporting this research by providing real industrial case study data.

Funding
The authors received no direct funding for this research.

Author details
Salman Saeidlou[1]
E-mail: sxs989@bham.ac.uk
Mozafar Saadat[1]
E-mail: M.SAADAT@bham.ac.uk
Ebrahim Amini Sharifi[2]
E-mail: Amini.sharifi@aut.ac.ir
Guiovanni D. Jules[1]
E-mail: gdj039@bham.ac.uk
[1] School of Mechanical Engineering, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK.
[2] School of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran.

Cover Image
Source:

Citation information
Cite this article as: Agent-based distributed manufacturing scheduling: an ontological approach, Salman Saeidlou, Mozafar Saadat, Ebrahim Amini Sharifi & Guiovanni D. Jules, *Cogent Engineering* (2019), 6: 1565630.

References

Adhau, S., Mittal, M. L., & Mittal, A. (2012). A multi-agent system for distributed multi-project scheduling: An auction-based negotiation approach. *Engineering Applications of Artificial Intelligence*, *25*(8), 1738–1751. doi:10.1016/j.engappai.2011.12.003

Alvarez, E. (2007). Multi-plant production scheduling in SMEs. *Robotics and Computer-Integrated Manufacturing*, *23*(6), 608–613. doi:10.1016/j.rcim.2007.02.006

Anand, N., van Duin, R., & Tavasszy, L. (2014). Ontology-based multi-agent system for urban freight transportation. *International Journal of Urban Sciences*, *18*(2), 133–153. doi:10.1080/12265934.2014.920696

Artto, K., Kulvik, I., Poskela, J., & Turkulainen, V. (2011). The integrative role of the project management office in the front end of innovation. *International Journal of Project Management*, *29*(4), 408–421. doi:10.1016/j.ijproman.2011.01.008

Aydin, M. E., & Fogarty, T. C. (2004a). A simulated annealing algorithm for multi-agent systems: A job-shop scheduling application. *Journal of Intelligent Manufacturing*, *15*(6), 805–814. doi:10.1023/B:JIMS.0000042665.10086.cf

Aydin, M. E., & Fogarty, T. C. (2004b). Teams of autonomous agents for job-shop scheduling problems: An Experimental Study. *Journal of Intelligent Manufacturing*, *15*(4), 455–462. doi:10.1023/B:JIMS.0000034108.66105.59

Bai, D., & Tang, L. (2013). Open shop scheduling problem to minimize makespan with release dates. *Applied Mathematical Modelling*, *37*(4), 2008–2015. doi:10.1016/j.apm.2012.04.037

Barbosa, J., Leitão, P., Adam, E., & Trentesaux, D. (2015). Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution. *Computers in Industry*, *66*, 99–111. doi:10.1016/j.compind.2014.10.011

Bellifemine, F. L., Caire, G., & Greenwood, D. (2007). *Developing multi-agent systems with JADE* (Vol. 7). Chichester, UK: John Wiley & Sons.

Bonabeau, E., Corne, D., Knowles, J., & Poli, R. (2010). Swarm intelligence theory: A snapshot of the state of the art. *Theoretical Computer Science*, *411*(21), 2081–2083. doi:10.1016/j.tcs.2010.03.001

Cowling, P. I., Ouelhadj, D., & Petrovic, S. (2003). A multi-agent architecture for dynamic scheduling of steel hot rolling. *Journal of Intelligent Manufacturing*, *14*(5), 457–470. doi:10.1023/A:1025701325275

Deepa, O., & Senthilkumar, A. (2016). Swarm intelligence from natural to artificial systems: Ant colony optimization. *Networks (GRAPH-HOC)*, *8*(1), 9–17.

Frey, D., Nimis, J., Wörn, H., & Lockemann, P. (2003). Benchmarking and robust multi-agent-based production planning and control. *Engineering Applications of Artificial Intelligence*, *16*(4), 307–320. doi:10.1016/S0952-1976(03)00075-7

Gabel, T. (2009). *Multi-agent reinforcement learning approaches for distributed job-shop scheduling problems* (Doctoral dissertation). Retrieved from https://repositorium.ub.uni-osnabrueck.de

Gaševic, D., Djuric, D., & Devedžic, V. (2009). *Model driven engineering and ontology development*. Berlin, Heidelberg: Springer Science & Business Media.

Gen, M., Zhang, W., & Hao, X. (2017). Advances in hybrid metaheuristics for stochastic manufacturing scheduling: Part II case studies. In *Proceedings of the Tenth International Conference on Management Science and Engineering Management* (pp. 1079–1094). Singapore: Springer.

Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., … Tu, S. W. (2003). The evolution of Protégé: An environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, *58*(1), 89–123. doi:10.1016/S1071-5819(02)00127-1

Giordani, S., Lujak, M., & Martinelli, F. (2013). A distributed multi-agent production planning and scheduling framework for mobile robots. *Computers & Industrial Engineering*, *64*(1), 19–30. doi:10.1016/j.cie.2012.09.004

Goldingay, H., & Van Mourik, J. (2013). The effect of load on agent-based algorithms for distributed task allocation. *Information Sciences*, *222*, 66–80. doi:10.1016/j.ins.2011.06.011

Gruber, T. (1993). What is an Ontology. Retrieved from http://www-ksl. stanford. edu/kst/whatis-an-ontology

Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology? In *Handbook on ontologies* (pp. 1–17). Berlin, Heidelberg: Springer.

Guo, Q., & Zhang, M. (2009). A novel approach for multi-agent-based intelligent manufacturing system. *Information Sciences*, *179*(18), 3079–3090. doi:10.1016/j.ins.2009.05.009

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., … Wassick, J. (2014). Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, *62*, 161–193. doi:10.1016/j.compchemeng.2013.12.001

Hasan, S. K., Sarker, R., Essam, D., & Cornforth, D. (2009). A genetic algorithm with priority rules for solving job-shop scheduling problems. In *Natural intelligence for scheduling, planning and packing problems* (pp. 55–88). Berlin, Heidelberg: Springer.

Jules, G., & Saadat, M. (2017). Agent cooperation mechanism for decentralized manufacturing scheduling. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*(12), 3351–3362. doi:10.1109/TSMC.2016.2578879

Jules, G. D., Saadat, M., & Li, N. (2013). On designing a unified ontology for holonic manufacturing networks. In *Integration of practice-oriented knowledge technology: Trends and prospectives* (pp. 207–220). Berlin, Heidelberg: Springer.

Jules, G. D., Saadat, M., & Saeidlou, S. (2015). Holonic ontology and interaction protocol for manufacturing network organization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *45*(5), 819–830. doi:10.1109/TSMC.2014.2387099

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*. (pp. IV, 1942–1948). Piscataway.

Komma, V. R., Jain, P. K., & Mehta, N. K. (2011). An approach for agent modeling in manufacturing on JADE™ reactive architecture. *The International Journal of Advanced Manufacturing Technology*, *52* (9–12), 1079–1090. doi:10.1007/s00170-010-2784-2

Kotulski, L., Sędziwy, A., & Strug, B. (2014). Translation of graph-based knowledge representation in multi-agent system. *Procedia Computer Science*, *29*, 1048–1056. doi:10.1016/j.procs.2014.05.094

Kulon, J., Broomhead, P., & Mynors, D. J. (2006). Applying knowledge-based engineering to traditional manufacturing design. *The International Journal of Advanced Manufacturing Technology*, *30*(9–10), 945–951. doi:10.1007/s00170-005-0067-0

Kurdi, M. (2017). An improved island model memetic algorithm with a new cooperation phase for multi-objective job shop scheduling problem. *Computers & Industrial Engineering*, *111*, 183–201. doi:10.1016/j.cie.2017.07.021

Lawrence, S. (1984). *Resouce constrained project scheduling: An experimental investigation of heuristic scheduling techniques (Supplement)*. Graduate School of Industrial Administration Pittsburgh, PA: Carnegie-Mellon University.

Leitão, P., Barbosa, J., & Trentesaux, D. (2012). Bio-inspired multi-agent systems for reconfigurable manufacturing systems. *Engineering Applications of Artificial Intelligence*, *25*(5), 934–944. doi:10.1016/j.engappai.2011.09.025

Li, W., Veliyath, R., & Tan, J. (2013). Network characteristics and firm performance: An examination of the relationships in the context of a cluster. *Journal of Small Business Management*, *51*(1), 1–22. doi:10.1111/jsbm.2013.51.issue-1

Lin, L. F., Zhang, W. Y., Lou, Y. C., Chu, C. Y., & Cai, M. (2011). Developing manufacturing ontologies for knowledge reuse in distributed manufacturing environment. *International Journal of Production Research*, *49*(2), 343–359. doi:10.1080/00207540903349021

Lin, T. L., Horng, S. J., Kao, T. W., Chen, Y. H., Run, R. S., Chen, R. J., … Kuo, I. H. (2010). An efficient job-shop scheduling algorithm based on particle swarm optimization. *Expert Systems with Applications*, *37*(3), 2629–2636. doi:10.1016/j.eswa.2009.08.015

Luck, M., McBurney, P., & Preist, C. (2003). *Agent technology: Enabling next generation computing (a roadmap for agent based computing)*. University of Southampton: AgentLink.

Martin, S., Ouelhadj, D., Beullens, P., Ozcan, E., Juan, A. A., & Burke, E. K. (2016). A multi-agent based cooperative approach to scheduling and routing. *European Journal of Operational Research*, *254*(1), 169–178. doi:10.1016/j.ejor.2016.02.045

Meloni, C., Pacciarelli, D., & Pranzo, M. (2004). A rollout metaheuristic for job shop scheduling problems. *Annals of Operations Research*, *131*(1–4), 215–235. doi:10.1023/B:ANOR.0000039520.24932.4b

Mohebbi, S., & Shafaei, R. (2012). e-Supply network coordination: The design of intelligent agents for buyer-supplier dynamic negotiations. *Journal of Intelligent Manufacturing*, *23*(3), 375–391. doi:10.1007/s10845-009-0377-4

Monostori, L., Váncza, J., & Kumara, S. R. (2006). Agent-based systems for manufacturing. *CIRP Annals-Manufacturing Technology*, *55*(2), 697–720. doi:10.1016/j.cirp.2006.10.004

Müller-Seitz, G., & Sydow, J. (2012). Maneuvering between networks to lead–A longitudinal case study in the semiconductor industry. *Long Range Planning*, *45* (2–3), 105–135. doi:10.1016/j.lrp.2012.02.001

Nguyen, S., Mei, Y., & Zhang, M. (2017). Genetic programming for production scheduling: A survey with a unified framework. *Complex & Intelligent Systems*, *3*(1), 41–66. doi:10.1007/s40747-017-0036-x

Noy, N. F., Sintek, M., Decker, S., Crubézy, M., Fergerson, R. W., & Musen, M. A. (2001). Creating semantic web contents with protege-2000. *IEEE Intelligent Systems*, *16*(2), 60–71. doi:10.1109/5254.920601

Owiti, O. H., Omulo, E. T. O., Okelo-Odongo, W., & Manderick, B. (2014). Game theoretic multi-agent algorithms for the job shop scheduling problem. *International Journal of Computer and Information Technology*, *3*, 06.

Owliya, M., Saadat, M., Jules, G. G., Goharian, M., & Anane, R. (2013). Agent-based interaction protocols and topologies for manufacturing task allocation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *43*(1), 38–52. doi:10.1109/TSMCA.2012.2192263

Rini, D. P., Shamsuddin, S. M., & Yuhaniz, S. S. (2011). Particle swarm optimization: Technique, system and challenges. *International Journal of Computer Applications*, *14*(1), 19–26. doi:10.5120/1810-2331

Saeidlou, S., Saadat, M., Amini Sharifi, E., & Jules, G. D. (2017). An ontology-based intelligent data query system in manufacturing networks. *Production & Manufacturing Research*, *5*(1), 250–267. doi:10.1080/21693277.2017.1374887

Schalkoff, R. J. (2011). *Intelligent systems: Principles, paradigms and pragmatics*. Boston, MA: Jones & Bartlett Publishers.

Shen, L., Dauzère-Pérès, S., & Neufeld, J. S. (2018). Solving the flexible job shop scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, *265*(2), 503–516. doi:10.1016/j.ejor.2017.08.021

Shen, W. (2002). Distributed manufacturing scheduling using intelligent agents. *IEEE Intelligent Systems*, *17* (1), 88–94. doi:10.1109/5254.988492

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *The Knowledge Engineering Review*, *11*(2), 93–136. doi:10.1017/S0269888900007797

Vallikavungal Devassia, J., Salazar-Aguilar, M. A., & Boyer, V. (2018). Flexible job-shop scheduling problem with resource recovery constraints. *International Journal of Production Research*, *56*, 1–18.

Vrba, P., & Marik, V. (2010). Capabilities of dynamic reconfiguration of multiagent-based industrial control systems. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *40*(2), 213–223. doi:10.1109/TSMCA.2009.2034863

Wang, L., Tang, D. B., Gu, W. B., Zheng, K., Yuan, W. D., & Tang, D. S. (2012). Pheromone-based coordination for manufacturing system control. *Journal of Intelligent Manufacturing*, *23*(3), 747–757. doi:10.1007/s10845-010-0426-z

Wang, K., & Choi, S. H. (2014). A holonic approach to flexible flow shop scheduling under stochastic processing times. *Computers & operations research*, *43*, 157–168. doi:10.1016/j.cor.2013.09.013

Wang, Z., & Liu, Y. (2006, October). A multi-agent agile scheduling system for job-shop problem. In *Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on* (Vol.2, pp. 679–683). Jinan, China: IEEE.

Wong, T. N., Leung, C. W., Mak, K. L., & Fung, R. Y. (2006). Dynamic shopfloor scheduling in multi-agent manufacturing systems. *Expert Systems with Applications*, *31*(3), 486–494. doi:10.1016/j.eswa.2005.09.073

Wu, Z. (2005). *Multi-agent workload control and flexible job shop scheduling* (Doctoral dissertation). Retrieved from https://scholarcommons.usf.edu

Yoo, M. J., & Müller, J. P. (2002, April). *Using multi-agent system for dynamic job shop scheduling*. 4th International Conference on Enterprise Information Systems : Universidad de Castilla-La Mancha, Ciudad Real, Spain. UCLM. s.l. : s.n., 8 .

Zhao, B., Gao, J., Chen, K., & Guo, K. (2018). Two-generation Pareto ant colony algorithm for multi-objective job shop scheduling problem with alternative process plans and unrelated parallel machines. *Journal of Intelligent Manufacturing*, *29*(1), 93–108. doi:10.1007/s10845-015-1091-z

Cogent Engineering **(ISSN: 2331-1916) is published by Cogent OA, part of Taylor & Francis Group.**