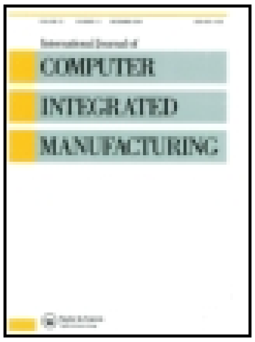


International Journal of
COMPUTER
INTEGRATED
MANUFACTURING

Taylor & Francis
Taylor & Francis Group

International Journal of Computer Integrated Manufacturing

ISSN: 0951-192X (Print) 1362-3052 (Online) Journal homepage: https://www.tandfonline.com/loi/tcim20

A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems

Fuqing Zhao , Yi Hong , Dongmei Yu , Yahong Yang & Qiuyu Zhang

To cite this article: Fuqing Zhao , Yi Hong , Dongmei Yu , Yahong Yang & Qiuyu Zhang (2010) A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems, International Journal of Computer Integrated Manufacturing, 23:1, 20-39, DOI: 10.1080/09511920903207472

To link to this article: https://doi.org/10.1080/09511920903207472

International Journal of Computer Integrated Manufacturing
Vol. 23, No. 1, January 2010, 20–39

A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems

Fuqing Zhao[a]*, Yi Hong[a], Dongmei Yu[a], Yahong Yang[b] and Qiuyu Zhang[a]

[a]*School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, Gansu, P.R. China;* [b]*College of Civil Engineering, Lanzhou University of Techchnology, Lanzhou 730050, Gansu, P.R. China*

(*Received 29 January 2009; final version received 25 July 2009*)

Modern manufacturing systems have to cope with dynamic changes and uncertainties such as machine breakdown, hot orders and other kinds of disturbances. Holonic manufacturing systems (HMS) provide a flexible and decentralised manufacturing environment to accommodate changes dynamically. HMS is based on the notion of holon, an autonomous, co-operative and intelligent entity which is able to collaborate with other holons to complete the tasks. HMS requires a robust coordination and collaboration mechanism to allocate available resources to achieve the production goals.

In this paper, a basic integrated process planning and scheduling system, which is applicable to the holonic manufacturing systems is presented. A basic architecture of holonic manufacturing system is proposed from the viewpoint of the process planning and the scheduling systems. Here, the process planning is defined as a process to select suitable machining sequences of machining features and suitable operation sequences of machining equipments, taking into consideration the short-term and long-term capacities of machining equipments. A fuzzy inference system (FIS), in choosing alternative machines for integrated process planning and scheduling of a job shop in HMS, is presented. Instead of choosing alternative machines randomly, machines are being selected based on the machine's capacity. The mean time for failure (MTF) values are input in a fuzzy inference mechanism, which outputs the machine reliability. The machine is then being penalised based on the fuzzy output. The most reliable machine will have the higher priority to be chosen. In order to overcome the problem of un-utilisation machines, sometimes faced by unreliable machine, the hybrid particle swarm optimisation (PSO) with differential evolution (DE) has been applied to balance the load for all the machines. Simulation studies show that the proposed system can be used as an effective way of choosing machines in integrated process planning and scheduling.

Keywords: holonic manufacturing systems (HMS); process planning, production scheduling; particle swarm optimisation; differential evolution (DE)

1. Introduction

A holonic manufacturing system (HMS)(HMS Consortium) (Van Brussel *et al.* 1998) is a manufacturing system where key elements, such as machines, cells, factories, parts, products, operators, teams, etc., are modelled as 'holons' having autonomous and co-operative properties.

The decentralised information structure, the distributed decision-making authority, the integration of physical and informational aspects, and the cooperative relationship among holons, make HMS a new paradigm to meet today's agile manufacturing challenges (Valckanaers *et al.* 1997, Giret and Botti 2006).

Manufacturing scheduling is very important because of its direct link to product delivery, inventory levels, and machine utilisation. Effective scheduling, however, has been proven to be extremely difficult because of the combinatorial nature of integer optimisation and the large size of practical problems (Cooke 2004). In practice, material planning systems, e.g. MRP or MRP II are often used for high-level production planning and scheduling (Turgay and Taskin 2007). MRPII system is a hierarchically structured information system which is based on the idea of controlling all flows of materials and goods by integrating the plans of sales, finance and operation, in applying MRPII in practice, one of the main problems is that there is little help with the necessary aggregation and disaggregation process, especially when uncertain demand exists. Because these systems generally ignore resource capacities, resulting plans or schedules are usually infeasible. Many heuristic methods have been developed to dispatch parts at the local (resource or machine) level based on due dates, criticality of operations, processing times, and machine utilisation (Malakooti 2004, Axsater 2007, Monch Lars *et al.* 2007,

*Corresponding author. Email: fzhao2000@hotmail.com

ISSN 0951-192X print/ISSN 1362-3052 online

DOI: 10.1080/09511920903207472
http://www.informaworld.com

Pedersen *et al.* 2007). Artificial intelligence (AI) approaches have also been proposed based on the application of scheduling rules (Jawahar *et al.* 1998, Masood 2004, Yin Xiao-Feng and Wang Feng-Yu 2004). Schedules obtained by heuristics, however, are often of questionable quality, and there is no good way systematically to improve the schedules generated.

The concept of holonic manufacturing has been studied by IMS-HMS consortium. Similar research has been performed under the banner of 'agent-based manufacturing.' When applied to manufacturing, an agent is a software object representing an element in a manufacturing system such as a product or a machine. Similar to a holon, an agent may have autonomous and cooperative properties, and is the building block of the system.

In practice, scheduling and planning problems are considered together as manifested in classic combinatorial problems (Wang *et al.* 2008, Chan *et al.* 2009, Shao Xinyu *et al.* 2009). The practical problems always involve multiple objectives that need to be addressed simultaneously. Hence, the actual optimisation objective is to determine the process plan and schedule concurrently. Owing to the complexity of manufacturing systems, process planning and scheduling are often carried out sequentially with little communication. Process planning seldom considers job shop capacity and scheduling information, such as resource capacity and availability. Production scheduling, on the other hand, is performed under fixed parameters without alternatives which provide alternative production flows. Re-planning is frequently required by improvisation with a long throughput time and other unexpected problems. During the last decade, several integration approaches have been proposed, but most integrated process planning and scheduling methods focused on the time aspects of alternative machines when conducting scheduling.

HMS is a holarchy which integrates all procedures of manufacturing activities from order management through design, production and marketing to fulfill the agile manufacturing enterprise. An HMS is therefore a manufacturing system where key elements, such as raw materials, machines, products, parts, and AGVs, have autonomous and cooperative properties (Deen 2003.).

Currently, HMS consortium partners have developed their own testbeds using their existing software and hardware environments under the same concepts and similar system architectures. Most of the early research results on HMS were reported only internally in the HMS consortium. However, some results have been published, (McFarlane and Bussman 2000) give a review of existing work in holonic manufacturing systems relevant to production planning and control, and provide an analysis of the scope and applicability for specific application domain. Heragu, *et al.* proposed a framework, which models the entities (e.g., parts) and resources (e.g., material handling devices, machines, cells, departments) as holonic structures, and introduces real-time negotiation mechanisms to solve task allocation and planning problems (Heragu *et al.* 2002). Leitao, Paulo and Restivo, Francisco present a holonic approach to manufacturing scheduling which combines centralised and distributed strategies to improve responsiveness of manufacturing systems to emergence (Leitao and Restivo 2002, Leitao and Restivo 2008). A decentralised holonic approach in manufacturing planning and control is presented to allocate material handling operations to the available system resources (Babiceanu and Chen 2009). Shrestha *et al.*, adopt genetic algorithm (GA) and dispatching rule (DR) in an integrated process planning and scheduling system which is applicable to HMS (Shrestha *et al.* 2008). Rais *et al.*, applied the GA and the dynamic programming (DP) methods to select suitable machining sequences and sequences of machining equipment in holonic manufacturing control and planning systems (Rais *et al.* 2002). It has become a very active research area with a large number of publications including several related books (Deen 2003, Vladimír Mařík 2007, Vicente and Adriana 2008).

Applications of HMS have been at the inter-enterprise level on holonic collaborative enterprises, and mostly at the enterprise and manufacturing system level, and at the manufacturing execution system (MES) level (Vladimír Mařík *et al.* 2007, Vicente and Adriana 2008).

The objective of this paper is to present an integrated process planning and scheduling system which aims at realising a flexible production control in holonic manufacturing systems. This paper deals mainly with the process planning of product machining process, taking into consideration the future schedules of machining equipments. The following issues are discussed in the paper: (a) basic architecture of target HMS and process planning systems; (b) formulation of an objective function based on job time and machining cost of products; and (c) procedure to select suitable machining sequences and sequences of machining equipment for integrating the process planning task with the scheduling task.

In this paper, a fuzzy logic (FL) is proposed to decide alternative machines for integrated process planning and scheduling. The FL is introduced for the purposes of choosing appropriate machines based on the machines' reliability characteristics. This ensures the capability of the machine in fulfilling the production demand. In addition, based on the capability information, the load for each machine is balanced by using the hybrid particle swarm optimisation (PSO) with differential evolution (DE). The remainder of the paper is arranged as follows: Section 2 describes a holonic architecture for manufacturing systems and common scheduling problems and challenges. A proposed method to face these challenges is proposed in Section 3. Analysis results and discussions are provided in Section 4. Finally, the conclusions and future researches are given out in Section 5.

2. A holonic architecture for manufacturing systems and common scheduling problems

Figure 1 illustrates a holonic architecture that we are proposing for manufacturing systems with the focus on the system parts of process planning and production planning. The figure illustrates components of a manufacturing system holarchy. In Figure 1, resource holons (e.g. machines, robots) and task holons (product orders) are grouped in scheduling holons. Scheduling holons are grouped with other holons (e.g. stock management holon, etc) and form production planning holons. One holon may be part of more than one holarchy. For example, a resource holon may be the member of a scheduling holon and of a process planning holon at the same time. This kind of architecture provides more flexibilities than static and traditional computer integrated manufacturing (CIM) architectures do.

2.1. *The initial planning holon*

In the holonic manufacturing system, each holon cooperates with other holons and makes use of external resource to accomplish the task delivered by the system. So the chains from customer requirement, through internal production planning holon and cooperative production task holon to supplier delivered task are established in the system, as shown in Figure 2.

After analysing the relationship and activity of each pair of entities, we find that we can model the relations between each pair of entities as the connection of customers and suppliers. The role of the initial planning holon is that of matching the task and resource capacity to accomplish the task that is fulfilled by the supplier. The decision processes are based on a supply and demand relationship and the outcomes are decided by selling and buying activity in respect of the resource and by market behaviour. As shown in Figure 2, there are four basic elements in the market behaviour: supplier, customer, item required and rules of transaction.

Suppliers are the holders of resource and the providers of the requirement of the project. The customer is the consumer of the requirement. Supplier and customer can be the rational entities of supplier, manufacturer, subseller, section in manufacturing enterprise and customer. The project requirement is the stuff which has value for the customer, including processed materials, semi-finished articles, final

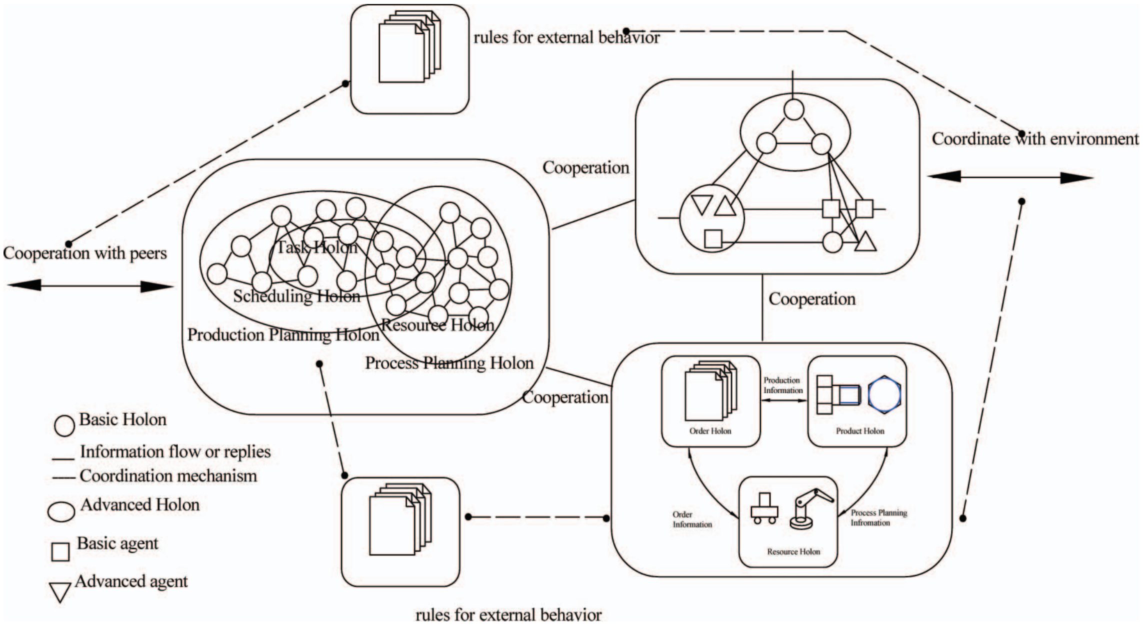


Figure 1. The holonic architecture.

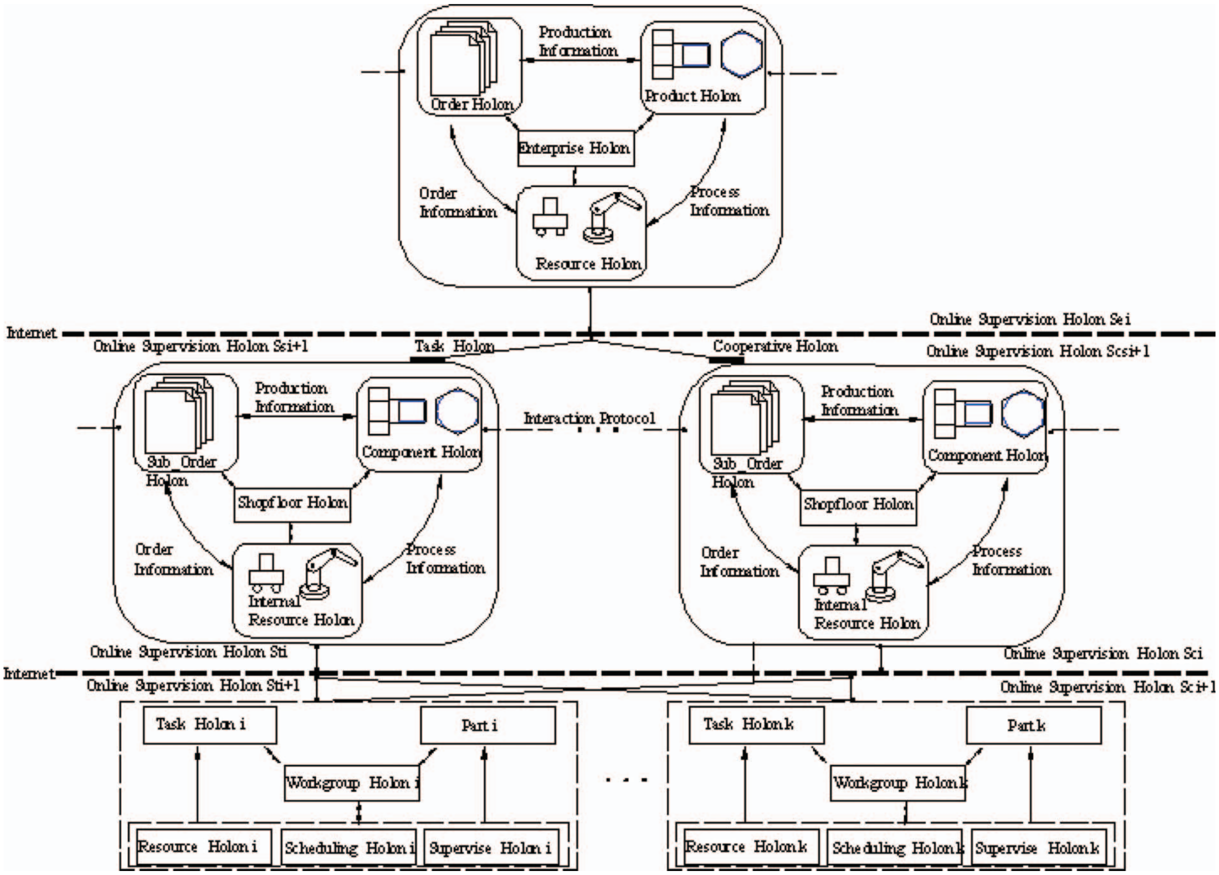


Figure 2. Initial planning holon.

product, services, and so on. Transaction rules are the criteria of transfer of project ownership from supplier to customer. That is to say, the rules for transaction and trade for resource capacity are in different time.

In the initial planning holon, all the relations between suppliers and customers are dynamic. Customer orders first enter into the system, according to the item required by customers, and the initial order details are generated by the order holon. Product holons generate the detailed information for the concrete production according to the collaboration information delivered by the order holon, then the product holons need to select the suppliers according to the different component combination to fulfil the customer order. In addition, product holons simultaneously need to collaborate with resource holons according to the capacity of the resource combination. All the processes in the collaboration are guided by certain rules in the concrete transaction.

In the application of the system, roles can change. For example, when one holon provides the resource to another holon, its role is as supplier. It will, however, play the customer role when capacity cannot meet the requirement and it turns for help to another holon. In the market mechanism all we need to do is to define the connectivity and activity of the entities in the holonic manufacturing system. So, when an enterprise holon lays out its production planning, it can use market mechanism method based on market equilibrium theory, and apply it to the modification feature that matches order/task to resource in the market, and consequently lay out the appropriate strategy, rule, or optimisation method to realise the optimum allocation of resource. The production planning and control system can respond to the market quickly through the holonic manufacturing system. The responsiveness can be seen at several points: 1) when the production task and resource experience changes, the system can reconfigure to set a new production planning and control system. 2) the system can collect, save, fetch and keep track of the information online. 3) it can respond quickly when the resource encounters any problems.

2.2. The detailed planning holon

A scheduling holon includes two kind of holon: resource holons and task holons, as shown in Figure 3.

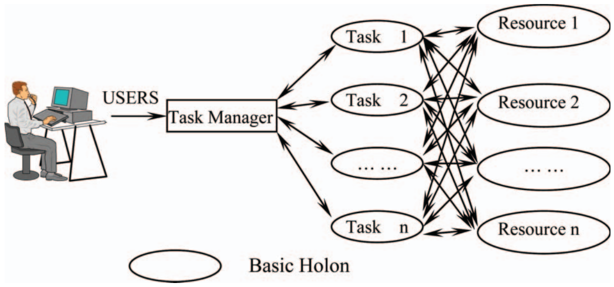


Figure 3. Task and resource holons for scheduling.

The resources are basic components of a manufacturing system (robots, NC machine, conveyors) or they can be cells made up of basic resources. Each resource is presented by a holon. The number of resources does not vary, except when resources are introduced or removed from the manufacturing system. A resource holon represents one resource with status, such as delivered activities and activities to be carried out. The activity of a resource is represented in an agenda. The agenda is the sequence of operations to be carried out and it specifies the expected durations for these operations as well as the free time intervals of the resource (Cheng 2004). The activity of a resource changes according to the operation of the resource and the dynamic situation of the manufacturing system (e.g. new orders, failures and delays in other resources).

The function of a task holon is managing the task system undertaken. It accepts the static programming and overall monitoring from the supplier; on the other hand, it will feedback the status of task which is carried on to the supplier. In some cases, if the activities which are implemented by task holons are the activities corresponding with input/output interface, task holons will communicate with the suppliers by input message/output message. In addition, task holons will collaborate with other task holons to propel the finish of the task under the satisfactions of priority. Furthermore, task holons will communicate with production holons and resource holons to instruct the task allocation according to the capacity constriction of the resources. Last but not least, task holons will monitor the task progress online and adjust according to the timely status collected in the system.

The 'task manager' interfaces with the user receiving orders for new tasks for the manufacturing system. This holon is responsible for launching 'task holon' whenever a new task is ordered. Besides, the task manager is responsible for dealing with dynamic changes of task conditions (e.g. when the user changes the deadline of a task). Task holons represent the possibilities to execute a plan for a task into a plan structure (Stahmer 2004, Babiceanu *et al.* 2005). The task manager has the knowledge about the resources that each task may need. Once launched, they directly negotiate with appropriate resource holons. The task manager is responsible for launching task holons, however, it will not launch task holons immediately after receiving requests from users. the task manager maintains a priority list of waiting tasks. The next task holon to be launched is the one with more priority and that does not conflict with other tasks that are still negotiating with resource holons. The possibility of conflict is detected when a task needs a resource that has received task announcement messages but has not received the correspondent acknowledgements or 'give up' messages. This means that this resource is still establishing a contract with other previously launched task holons.

The task holon and initial planning holon exchange production operation information which includes the information and method on how to accomplish certain processes in corresponding resources with constriction of time and cost, i.e. the process can be fulfilled by the certain resources combination with the required machining parameters and quality. The production knowledge, which includes information and methods on how to utilise certain resources to produce a certain type production, is communicated between production holon and task holon. In the process of communication with production holon and task holon, the production processes certain resource, the data structure for expressing production results and evaluation methods for production planning belong to production knowledge. The machining operation knowledge flows between production holon and operation holon. The procedure in the collaboration is monitored by the on-line supervision holon, which can also make suggestions for all the holons involved.

In an HMS, holons may form holarchies whose members collaborate through Cooperation Domains. Using the mechanism of virtual clustering, holons can be dynamically involved in different clusters (holarchies) and cooperate through a Cooperation Domain. The cluster exists for the duration of its cooperation tasks and disappears when the tasks are completed. A cooperation domain can be implemented and maintained through the creation of a mediator holon (as shown in Figure 4).

The process of the operation is as follows.

(1) A new order/task is generated in the market during the system operation. The order holon will send a bidding request to a

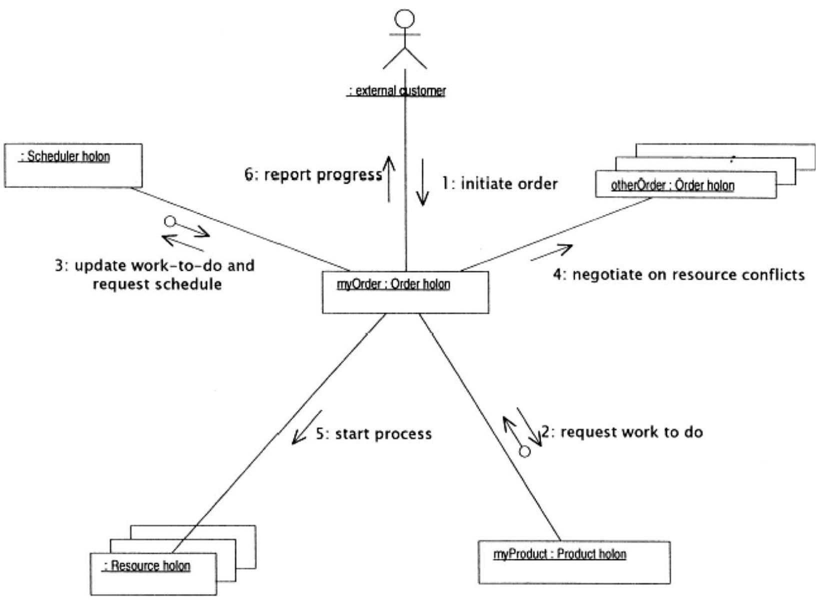


Figure 4. The scheduling holon's interaction with other holons.

resource holon. The information format can be defined as

$$\begin{aligned}
&Bidding = \{order/taskID,\ title,\ Number,\\
&\quad Time, Cost,\ \mathrm{Re}source,\ order\_Description\}\\
&order\_Description = \{order\_Deadline, order\_\\
&\quad Constriction, order\_\mathrm{Pr}iority, order\_\ Status\}\\
&order\_Status = \{finished,\ performing,\\
&\quad waiting, unscheduling\}
\end{aligned}$$

The functions of order holon include the following aspects.

(a) Assess order: analyse the feasibility of the order, obtain the validity of the order, and compute the possibility of the candidate order.

(b) Generate order, after the process of the assessment to the order, the detailed order documentary, which has legality in the business transaction, is generated. Meanwhile, the uniform data format of the generated order which is convenient to be processed is needed.

(c) Reply to the collaboration request. In this procedure, order holon handles the collaboration request during the process of order assessment and communication with outer environment to facilitate the success of the order assessment.

(d) Reply to the request information from customer, order holon handle the request information when customers ask for a check on the progress of the order and then feedback the search information to customers.

(2) The resource holon will communicate with the product holon and make sure whether it can satisfy the technical requirement for the production order such as machining methods, machining precision etc. Then it will decide whether to bid for the order/task according to capacity and benefit to the process. When the resource holon obtains the order/task it will create an order/task holon to group the virtual order/task clustering and start up the planning holon.

(3) The planning holon will set up production planning for the order having the highest priority. It communicates with the production holon, breaks down the order/task according to the design, process and related information, and method, extracts the resource requirement model, acquires the scheme of resource requirement, and confirms the objective of the subtask/order to generate the information that constitutes the subtask/order.

(4) The planning holon invokes the mediator holon to deliver the subtask/order holon to the mediator holon to apply the resource. The mediator holon sends the subtask/order information to subholon of local holon or candidate holon partner according to history and register information. The planning holon also receives bids from other holons.

(5) The subresource holon and other resource holons bid to undertake the task/order according to how they evaluate their interest in it. The mediator holon classifies the market and selects the resource holon according to task application result, resource storage, priority etc. When there is any collision in the system, it negotiates with other connected holons to adopt a different negotiation strategy. Finally, the mediator holon informs the planning holon of the candidate holon for carrying out the sub task/order.

(6) The planning holon invokes the scheduling holon. The scheduling holon allocates the task to a candidate resource holon by applying a particular optimisation algorithm. The scheduling holon creates a subtask holon which connects with the corresponding resource holon to analyse the task into its basic elements.

(7) If the task has not been broken down into its irreducible elements, basic tasks which can not be divided, and the resource has not been laid out into the basic resource holarchy constituted by the manufacturing entities, then the above procedure repeats and recursion operates. The subtasks for the realisation of the whole task can be isolated. The resource is also designated for each subtask. Different layer and task oriented dynamic virtual clustering for the task/order is built up.

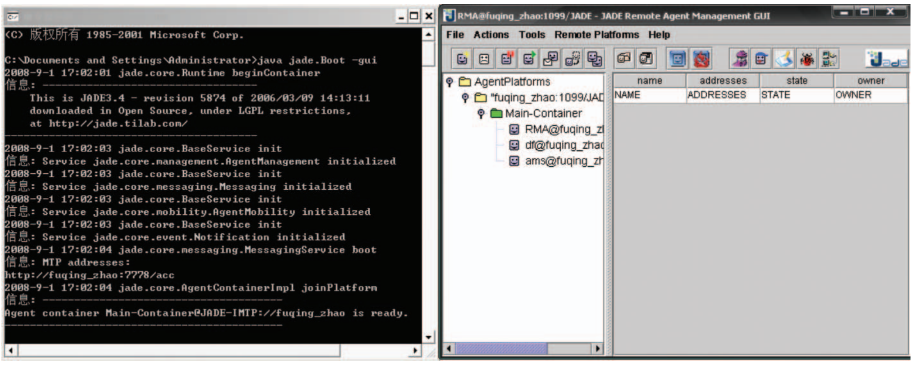
The proposed holonic system and the interaction process are implemented on the JADE (Java Agent Development Environment) platform. JADE is a multi-agent system platform which conforms strictly to FIPA criteria. The JADE programmer can use JAVA to exploit systems when the agent is built (Administrator Guide, Programmer Guide). Meanwhile, because JADE simplifies the communication process between agents by delivering messages which abide by FIPA criteria (FIPA), the message can also be inserted into the sequenced object to realise the standardisation parameter delivery. Furthermore, the yellow function can be directly used because DF function is provided by JADE to guarantee the register for customer systems. With AMS and Sniffer tools provided by JADE, users can debug the implementation platform and easily achieve the total functioning of the system. The startup interface is shown in Figure 5.

2.3. *Expanded job-shop scheduling problem (EJSSP) (Li and Pei-Huang 2004)*

EJSSP is a deterministic and static scheduling problem. There are *m* distinct machines to process *n* jobs that have their specific processing routines. Each job's operation has its precedence and takes up a deterministic time period at a specific machine. At one time, there is only one operation at a machine and the job does not leave this machine until the operation is completed. The operation starting time of each job must be within predefined regions, which are subject to the available time and due dates of jobs. Enabling conditions of an operation are prescribed by technological planning including job and resource requirement such as machines and fix tools, as well as cutting tools. It can be seen easily that EJSSP is significantly more general than the standard job-shop scheduling problem (JSSP).

2.4. *Modelling and analysis of EJSSP*

Expanded job-shop scheduling problem (EJSSP) is a deterministic and static scheduling problem. Each job's operation has its precedence and takes up a deterministic time period at a specific machine (Yu Haibin 2001, Petrovic Sanja and Petrovic 2008).



```
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>java jade.Boot -gui
2008-9-1 17:02:01 jade.core.Runtime beginContainer
信息: ----------------------------------
    This is JADE3.4 - revision 5874 of 2006/03/09 14:13:11
    downloaded in Open Source, under LGPL restrictions,
    at http://jade.tilab.com/
----------------------------------------
2008-9-1 17:02:03 jade.core.BaseService init
信息: Service jade.core.management.AgentManagement initialized
2008-9-1 17:02:03 jade.core.BaseService init
信息: Service jade.core.messaging.Messaging initialized
2008-9-1 17:02:03 jade.core.BaseService init
信息: Service jade.core.mobility.AgentMobility initialized
2008-9-1 17:02:03 jade.core.BaseService init
信息: Service jade.core.event.Notification initialized
2008-9-1 17:02:04 jade.core.messaging.MessagingService boot
信息: MTP addresses:
http://fuqing_zhao:7778/acc
2008-9-1 17:02:04 jade.core.AgentContainerImpl joinPlatform
信息: --------------------------------------
Agent container Main-Container@JADE-IMTP://fuqing_zhao is ready.
--------------------------------------------
```

RMA@fuqing_zhao:1099/JADE - JADE Remote Agent Management GUI

name	addresses	state	owner
NAME	ADDRESSES	STATE	OWNER

Figure 5. Startup interface of JADE platform.

2.4.1. Notations for EJSSP

The symbols for modelling scheduling problems are as follows:

n number of jobs;
n_i number of operations of job i;
m type number of various resources;
r_s number of resources of type s, $s \in [1,2, \ldots ,m]$;
R_i set of pairs of operations $\{k,l\}$ belonging to job i, operation k precedes l;
Q_i set of pairs of operations $\{k,l\}$ belonging to job i;
N_q set of operations requiring resource q, $q \in [1,2, \ldots r]$;
H large enough positive number;
t_{il} processing time of operation l of job i, $l \in [1, \ldots , n_i]$;
x_{ik} starting time of operation k of job i, $k \in [1, \ldots , n_i]$;
x_{si} starting time of the first (or free) operation of job i;
x_{ie} completion time of the last (or free) operation of job i;
a_i availability time of job i;
d_i delivery due date of job i;
$[i,k]$ the kth operation of job i, also called operation k in short if no confusion is caused;

$$y_{kl} = \begin{cases} 1 \text{ if operation } k \text{ precedes operation } l \\ 0 \text{ otherwise} \end{cases}$$

where $\{k,l\} \in Q_i$, $i = 1,2, \ldots ,\text{n}$;

$$z_{ij} = \begin{cases} 1 \text{ if operation } i \text{ precedes operation } j \\ 0 \text{ otherwise} \end{cases}$$

where $\{i,j\} \in N_q$, $q = 1,2, \ldots , r_s$, $s = 1,2, \ldots , m$.
Note: $i \in [1, \ldots , n]$; $s \in [1, \ldots , m]$; free operation means operation without precedence restrictions from technological planning.

2.4.2. Modelling and analysis of EJSSP

A feasible solution means that the scheduling satisfies all constraint conditions. There are mainly four types of major constraints for any operation as follows:

(1) *Precedence constraint*. Precedence constraint means that some jobs must be processed at different machines in fixed precedence sequence defined by technological planning. Concretely, the lth operation of job i must be before the kth operation of the same job, t_{ik} means processing time of operation of k of job i, if $\{k,l\} \in R_i$, i.e.

$$\begin{aligned} &\sum_{i=1}^{n}\sum_{l=1}^{n_i} x_{il} - \sum_{i=1}^{n}\sum_{k=1}^{n_i} x_{ik} \\ &\geq \max\left\{\sum_{j=1}^{m}\sum_{t=1}^{di}(t+d_{ij})y_{ij}(t),\right. \\ &\left.\sum_{j=1}^{m}\sum_{t=1}^{d_i} t y_{kj}(t) x_{il} - t_{ik}\right\} \end{aligned} \tag{1}$$

(2) *Resource constraint*. Resource constraint means that any resource can only provide service for one operation at a time; for example, resource q can only be selected by one job waiting to be processed in the queue at any time.

$$\begin{aligned} &\sum_{i=1}^{n}\sum_{j=1}^{n_i} x_{ij} - \sum_{j=1}^{ni}\sum_{k=1}^{m} x_{jk} \\ &+ \prod_{i=1}^{n}\sum_{j=1}^{m_i}\sum_{t=1}^{d} y_{ij}(t)(x_{ij} - x_{jk}) \\ &+ \sum_{k=1}^{n}\sum_{l=1}^{n_i} H(1 - z_{kl}) \geq 0 \\ &if\{k,l\} \in N_q \end{aligned} \tag{2}$$

(3) *Job (hidden) constraint*. Although there may be no precedence constraint among some operations of a job, the constraint that the operations l and k could not be processed at the same time still exists because these two operations were done by the same job, i.e.

$$\begin{aligned} &\sum_{i=1}^{n}\sum_{j=1}^{n_i} x_{ij} - \sum_{i=1}^{n}\sum_{l=1}^{n_i} x_{il} \\ &+ \prod_{k=1}^{n}\sum_{i=1}^{n}\sum_{t=1}^{n_i}(z_{ij}(t)(x_{ij} - x_{jk}) \\ &+ H(1 - y_{kl})) \geq \sum_{i=1}^{n}\sum_{k=1}^{n_i} t_{ik} \\ &if\{k,l\} \in Q_i \end{aligned} \tag{3}$$

(4) *Starting and completion time constraint*. In practice, the starting time and that the completion time of a job are restricted by the available time and the due date of delivery. Mathematically, it can be depicted by Equations (4) and (5).

$$\begin{aligned} &\sum_{k=1}^{n}\sum_{t=1}^{d}\{x_{si}(y_{ik}(t) - z_{ik}(t)) - a_i\} \geq 0 \\ &i \in [1, \cdots , n] \end{aligned} \tag{4}$$

$$\sum_{i=1}^{n}\sum_{t=1}^{d}\{x_{ie}(y_{ik}(t)-z_{ik}(t))-d_i-t_{ie}\}\geq 0 \quad i\in[1,\cdots,n] \tag{5}$$

Minimising the total penalty for early and tardy jobs

$$Min\ Z=\sum_{i=1}^{n}\sum_{t=1}^{d_i}\sum_{k=1}^{n_i} [z_{ik}(t)\times \max(0,d_i-x_{ie}) + y_{ik}(t)\times \max(0,x_{ie}-d_i)] \tag{6}$$

2.5. Problems of traditional process planning

While 'manufacturing' refers to producing parts that meet specifications, process planning refers to the set of instructions that is used to make parts that meet specifications. In other words, process planning determines how a part will be manufactured and acts as a bridge between design and manufacturing. In process planning, the objectives would be to minimise the number of rejects, processing cost and manufacturing lead time. Therefore, process planning is a major determinant of the profitability of a product. There are several variables associated with raw materials: e.g. unit cost of raw material, amount of raw material required and salvage value of scrap. In machining processes, the variables are processing cost, processing time, set-up time and standard deviation (which determine the percentage of rejects).

The traditional and most widely used method of process planning is based on human experience. The disadvantages of this approach are the time required to acquire the expertise and that the plans developed may not be consistent due to the element of human judgment. The feasibility of a process plan is relying on the quality of design, availability of machine tools and other influences such as allocation of machine tools. Therefore, it is necessary to develop a process plan that has sufficient knowledge of both upstream (design) and downstream (scheduling) activities. The methods described below have been developed to overcome these problems as much as possible.

2.6. Integration of process planning and scheduling

Sormaz *et al.* defined conventional scheduling as a task that uses input from rigid process plans (Sormaz *et al.* 2004). These plans specify a unique choice of machines for each operation, and a unique sequence of operations, whereas integrated process planning and scheduling (or integrated scheduling) uses more flexible process plans as input. The flexible plans allow a choice of operation sequences and/or machines.

An increasing number of published papers have focused on this subject. Various approaches to the integrated scheduling problem have been introduced. Literature (Li and Mcmahon 2007, Shukla Sanjay Kumar *et al.* 2008) has shown that there exists a need for integrating the process planning and scheduling functions in manufacturing in order to achieve productivity improvements. Zhang *et al.* used heuristic procedures as the integrated approach. Examples of other approaches are integer programming (Zhang *et al.* 2003), rule-based approach (Li Jianxiang *et al.* 2004), simulated annealing approach, tabu search (Reddy and Ponnambalam 2003) and GA approach (Chyu Chiuh-Cheng and Wei-Shung 2008, Kim Haejoong and Park Woo 2008).

The concurrent integrated process planning and scheduling model mainly consists of an initial planning holon and a detailed planning holon. The initial planning holon is the core at the initial planning. It cooperates with other holons to finish the task at the initial planning. The input data to initiate planning holon can be classified into two types. One is the product model from a design system which represents the geometrical shape of the product, design tolerances, surface requirement and the properties of the material; the other is the resource information from the production scheduling holon. The task of the initial planning holon is to find all feasible operations as determined by part information and resource information. The initial planning holon does not carry out the complete process planning, but it can generate many feasible blocks made up of feasible processes based on individual operations. A block is generally machined on an individual machine unless it is mass production. The output of the initial planning holon is a series of blocks; these blocks are assigned to each machine based on feedback from production scheduling. In fact, the alternative process plans are made up of all these alternative blocks. Another task of the initial planning holon is to provide manufacturing evaluation for the designer.

The designer can use the tool to check the manufacturing feasibility and cost of alternative designs. Thus, designers can modify their designs so that they are manufacturable and cost-effective according to the current shop floor environment. The manufacturing evaluations produced by concurrent integrated process planning system should consider other approaches, in that its evaluations are based not only on the design (namely process capabilities that can be described by the part shape, dimensions, tolerances, surface finish, geometric and technological constraints, and economics of a process), but also on information on the shop floor resources. Shop floor resources greatly affect the manufacturability analysis. A design may be manufacturable under one set of shop floor resources, but not under another.

The detailed planning holon is executed just before the beginning of manufacturing. It will generate the detailed process planning, which includes selecting the cutting speed, feed rate, etc., and calculating machining time, programming the NC program for each operation of the processing route generated by the decision making holon. The output of the detailed planning holon is an entire document of process planning to be sent to the shop floor to guide the production.

In HMS, holons make autonomous decisions based on guidelines or system-wide constraints provided by high-level holons. Each holon can participate in the construction of different virtual clusters. During the evolution of the holarchy, one holon can be included in different holonic society to form a dynamic virtual cluster, which is based on multiple dimension task driven, as shown in Figure 7.

In addition to having some common characteristics such as distribution, autonomy, interaction, and openness, the architecture of the holonic system possesses some other characteristics as follows:

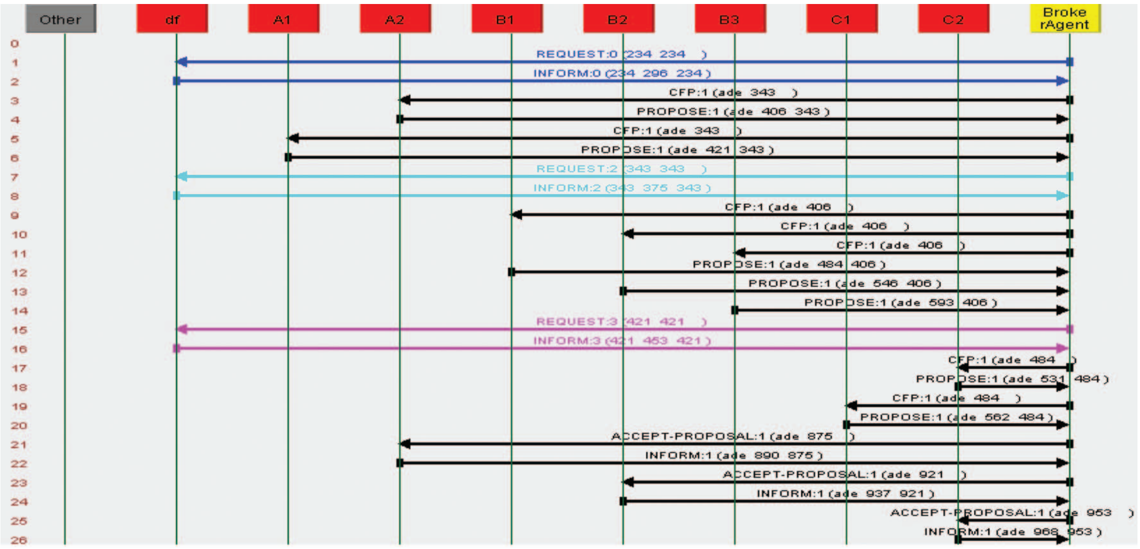
(1) Reconfiguration. It supports easy reconfiguration to accommodate the introduction of new manufacturing environments.
(2) Customisation. It supports customisable functionalities that enable users at all three planning levels to interactively manipulate process planning.
(3) Hybridisation. It has a hybrid architecture that is composed of the peer-to-peer relationships (task holon with initial planning holon) and the hierarchical relationships between holons (such as between planning holons).

3. A proposed approach

In HMS, any holon in the holon society may be involved in more than one cluster. With ongoing clustering and holons becoming involved in multiple compositions, a multi-dimensional cluster negotiation process is illustrated in Figure 6. There are three kinds of holons in the system: each type represents certain functions such as scheduling holon, resource holon and supervision holon. The interaction can be traced in JADE platform.

In a situation where an agent requires more assistance or cooperation from outside its virtual cluster, it will create further clusters for its cooperation subtasks. The subclustering process is then repeated, with subclusters and then subsubclusters etc. being formed as needed, resulting in a dynamically interlinked and overlapped cluster society. In this virtual cluster society, in principle, all tasks are distributed and solved cooperatively. Such an architecture is recursive and scalable. The simulation result interface is shown in Figure 7.

An integrated process planning and scheduling system, which is applicable for scheduling holon of the HMS, is vital for high efficiency communication and execution in the engineering application.



```
0
1  REQUEST:0 (234 234 )
2  INFORM:0 (234 296 234 )
3  CFP:1 (ade 343 )
4  PROPOSE:1 (ade 406 343 )
5  CFP:1 (ade 343 )
6  PROPOSE:1 (ade 421 343 )
7  REQUEST:2 (343 343 )
8  INFORM:2 (343 375 343 )
9  CFP:1 (ade 406 )
10 CFP:1 (ade 406 )
11 CFP:1 (ade 406 )
12 PROPOSE:1 (ade 484 406 )
13 PROPOSE:1 (ade 546 406 )
14 PROPOSE:1 (ade 593 406 )
15 REQUEST:3 (421 421 )
16 INFORM:3 (421 453 421 )
17 CFP:1 (ade 484 )
18 PROPOSE:1 (ade 531 484 )
19 CFP:1 (ade 484 )
20 PROPOSE:1 (ade 562 484 )
21 ACCEPT-PROPOSAL:1 (ade 875 )
22 INFORM:1 (ade 890 875 )
23 ACCEPT-PROPOSAL:1 (ade 921 )
24 INFORM:1 (ade 937 921 )
25 ACCEPT-PROPOSAL:1 (ade 953 )
26 INFORM:1 (ade 968 953 )
```

Figure 6. Multi-agent negotiation and interaction process based CNP (simulated result).

3.1. PSO and its corresponding mapping mechanism

The proposed approach is exhibited in Figure 8, where the modules of PSO and FL (available in Matlab Fuzzy Logic Toolbox) are the primary components.

PSO simulates a social behaviour such as bird flocking to a promising position for certain objectives in a multidimensional space (Coello Coello *et al.* 2004, van den Bergh *et al.* 2004). Like evolutionary algorithms, PSO conducts search using a population (called swarm) of individuals (called particles) that are updated from iteration to iteration. Each particle represents a candidate position (i.e. solution) to the problem at hand, resembling the particle of GA. A particle is treated as a point in an M-dimension space, and the status of a particle is characterised by its position and velocity (Izui Kazuhiro *et al.* 2008, Li *et al.* 2008). Initialised with a swarm of random particles, PSO is achieved through particles flying along the trajectory that will be adjusted based on the best experience or position of the one particle (called local best) and the best experience or position ever found by all particles (called global best). The M-dimension position for the ith particle in the tth iteration can be denoted as $x_i(t) = \{x_{i1}(t), x_{i2}(t), \ldots x_{iM}(t)\}$. Similarly, the velocity (i.e. distance change), also an M-dimension vector, for the ith particle in the tth iteration can be described as. $v_i(t) = \{v_{i1}(t), v_{i2}(t), \ldots, viM(\mathrm{t})\}$. The particle-updating mechanism for particle flying (search process) can be described as

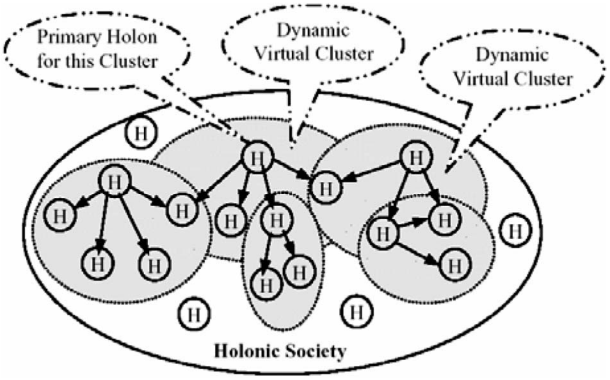


Figure 7. Dynamic virtual cluster in holonic society.

$$v_{id} = \chi(\varpi v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id})) \quad (7)$$

$$x_{id} = x_{id} + v_{id} \quad (8)$$

where p_i Pbest of agent i at iteration k, p_g gbest of the whole group χ compress factor, r_1,r_2 random numbers in (0,1), c_1, c_2 weighting factor, ϖ inertia function, in this paper, the inertia weight is set to the following equation. PSO algorithm is problem-independent, which means little specific knowledge relevant to a given problem is required. What we have to know is just the fitness evaluation for each solution. This advantage makes PSO more robust than many other search algorithms.

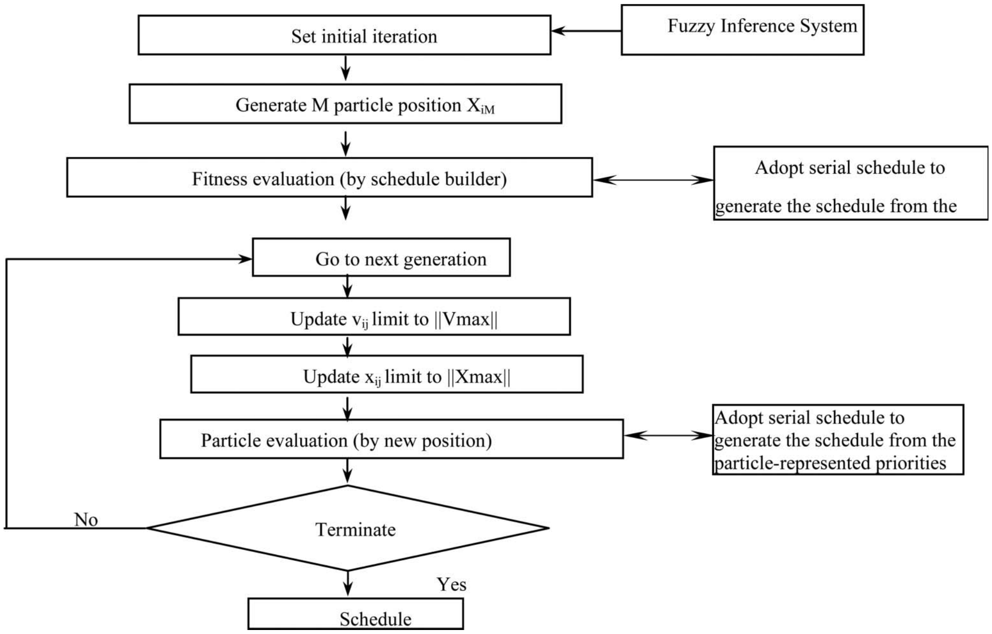
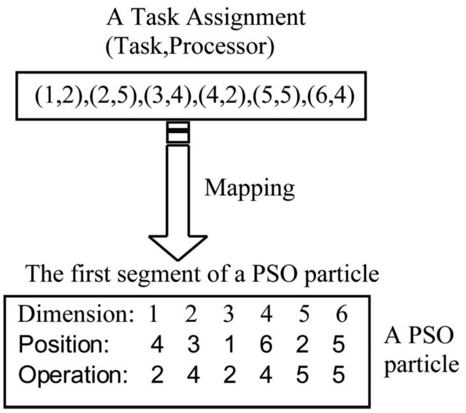

Set initial iteration
Fuzzy Inference System
Generate M particle position X_{iM}
Fitness evaluation (by schedule builder)
Adopt serial schedule to generate the schedule from the
Go to next generation
Update v_{ij} limit to ||Vmax||
Update x_{ij} limit to ||Xmax||
Particle evaluation (by new position)
Adopt serial schedule to generate the schedule from the particle-represented priorities
No
Terminate
Yes
Schedule

Figure 8. The schematic diagram of the proposed approach.

The algorithm keeps an updated version of two special variables throughout the course of its execution. Generally, initial swarm and initial particle velocities are generated randomly. In order to reduce the iterative time of PSO, a new method to generate initial swarm is introduced in this paper. Notice that most feasible solutions in task allocation are arranged according to the increasing order of the task and only a few tasks are reversed. Then, we arrange task order in our work according to the increasing order of task. If one task's execution order is the same as the other, the two task's orders are arranged randomly. For example, considering the tasks and operation on processor 1 in Figure 9, Figure 9 shows a mapping between one possible assignment instance to particle position coordinates in the PSO domain. Using such particle representation, the PSO population is represented as a P × M two-dimensional array consisting of N particles, each represented as a vector of M tasks. Thus, a particle flies in an M-dimensional search space. A task is internally represented as an integer value indicating the processor number to which this task is assigned to during the course of PSO. In our PSO algorithm, we map an M-task assignment instance into corresponding M-coordinate particle position. The algorithm starts by generating randomly as many potential assignments for the problem as the size of the initial population of the PSO. It then measures particles' fitness. We used Equation (10) as our fitness function. The algorithm keeps an updated version of two special variables throughout the course of its execution: 'global-best' position and 'local-best' position. It does that by conducting two ongoing comparisons: First, it compares the fitness of each particle being in its 'current' position with fitness of other particles in the population in order to determine the global-best position for each generation. Then, it compares different visited positions of a particle with its current position, in order to determine a local-best position for every particle. These two positions affect the new velocity of every particle in the population according to Equation (7). Figure 10 shows two possible expressions of the first segment of an initial particle that can be generated according to the new method. If tasks on every processor are arranged in this way, the probability that the initial particle may be a feasible solution, i.e. a feasible schedule, increases greatly.



Dimension:	1	2	3	4	5	6
Position:	4	3	1	6	2	5
Operation:	2	4	2	4	5	5

Figure 9. Tasks allocation to PSO particle mapping.

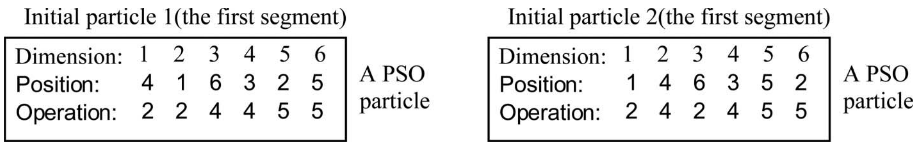
In Equation (7), inertia weight (w) is an important parameter to search ability of PSO algorithm. A large inertia weight facilitates searching new area while a small inertia weight facilitates fine-searching in the current search area. Suitable selection of the inertia weight provides a balance between global exploration and local exploitation, and results in less iterations on average to find a sufficiently good solution. Therefore, considering linearly decreasing the inertia weight from a relatively large value to a relatively small value through the course of PSO run, PSO tends to have more global search ability at the beginning of the run while having more local search ability near the end of the run. In this paper the inertia weight is set according to the following equation

$$\varpi = \varpi_{\max} - \frac{\varpi_{\max} - \varpi_{\min}}{I_{\max}} \times I$$

where $\varpi_{\max}$ = initial value of weighting coefficient
$\varpi_{\min}$ = final value of weighting coefficient
$I_{\max}$ = maximum number of iterations or generation
I = current iteration or generation number

In numerical simulations, the inertia weight is set to 1.4 to guarantee a wider search area and linearly decreasing to 0.3 to assure a finer search area in near optimisation point.

The acceleration constants c_1 and c_2 in Equation (7) adjust the amount of 'tension' in PSO system. Low


Initial particle 1(the first segment)
A PSO particle
Initial particle 2(the first segment)
A PSO particle

Dimension:	1	2	3	4	5	6
Position:	4	1	6	3	2	5
Operation:	2	2	4	4	5	5

Dimension:	1	2	3	4	5	6
Position:	1	4	6	3	5	2
Operation:	2	4	2	4	5	5

Figure 10. Two possible expressions on processor 1.

value allow particles to roam far from target regions before being tugged back, while high values result in abrupt movement toward, or past, target regions. According to experiences of other researchers, acceleration constants c_1 and c_2 are set to 2.0 for all following examples.

The parameter χ is the compress factor (CF), and it is used as mechanisms for the control of the velocity's magnitude. The basic system equation of PSO can be considered as a kind of difference equations. Therefore, the system dynamics, namely, search procedure, can be analysed by the eigen value analysis. Namely, the velocity of CF can be expressed as follows

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}$$

where $\varphi = c_1 + c_2$, $\phi \geq 4$.

For example, if $\varphi = 4.0$, then $\chi = 1.0$. As φ increases above 4.0, χ gets smaller. For example, if, $\varphi = 5.0$ then $\chi = 0.38$, and the damping effect is even more pronounced. The convergence characteristic of the system can be controlled by φ. Namely, Clerc (1999) found that the system behaviour can be controlled so that the system behaviour has the following features:

(a) The system does not diverge in a real value region and finally can converge.
(b) The system can search different regions efficiently by avoiding premature convergence.

Unlike other EC methods, CF of PSO ensures the convergence of the search procedures based on the mathematical theory. CF can generate higher-quality solutions than PSO with IWA (Eberhart 2000).

However, CF only considers dynamic behaviour of one agent and the effect of the interaction among agents. Namely, the equations were developed with fixed best positions (p_i and p_g) although p_i and p_g can be changed during search procedure in the basic PSO equations. The effect of p_i and p_g in the system dynamics is one for future works.

3.2. Hybrid PSO and differential evolution algorithm(HPSO)

PSO, which is a swarm evolution algorithm developed by the simulation of the food searching process in the natural world, has the ability to remember the best position of particles and has available a communication mechanism among the swarm, namely: through the cooperation and competition between individuals in the population to obtain the optimum of the problem. PSO and artificial life, as well as evolution algorithms have a close relationship, but compared with evolution algorithms, PSO keeps the global searching based strategy in the population. It adopts a simple velocity-position model to avoid complex genetic operations, and thanks to its special memories it can keep track on the current searching status and adjust the corresponding searching strategy.

PSO algorithm is problem-independent, which means little specific knowledge relevant to a given problem is required. What we have to know is just the fitness evaluation for each solution. This advantage makes PSO more robust than many other search algorithms. However, as a stochastic search algorithm, PSO is prone to lack global search ability at the end of a run. PSO may fail to find the required optima in case the problem to be solved is too complicated and complex. DE (Chang 2007, Qian Bin 2008) has the merits of memorising the best solution and sharing the group information in the candidate. We can control the search process and avoid individuals being trapped in local optimum more efficiently. Thus, a hybrid algorithm of PSO and DE, named HPSO, is presented as follows.

```
Begin
  STEP 1 Initialisation
    Initialise swarm population, each particle's position and velocity;
    Evaluate each particle's fitness;
    Initialise gbest position with the lowest fitness particle in swarm;
    Initialise pbest position with a copy of particle itself;
    Initialise $\varpi_{\max}$, $\varpi_{\min}$, $c_1$, $c_2$, maximum generation, and generation = 0.
    Determine $T_0$,$T_{end}$, $B$.

STEP 2 Operation
For PSO
do {
    generate next swarm by Equation (7) to Equation (8);
    find new gbest and pbest;
    update gbest of swarm and pbest of the particle;
    generation ++;
    }
    while (generation < maximum generation)
For DE
    For gbest particle S of swarm
    { BEGIN
Initialise: generate the DE population randomly
    Evaluate the individual
    Get the best solution $X_{best}$
While (the end qualification is not met)
```

```
        Exert mutation operation to each individual to acquire corresponding mutation individual

            $V_i(t+1) = X_{best}(t) + F(X_{r1}(t) - X_{r2}(t))$

      Exert mutation operation, then crossover operation to get the experiment individual

            $ui,j(t+1) = \begin{cases} v_{i,j}(t+1), & if(rand(j) \leq CR) \\ & or\, j = randn(i) \quad j = 1,2,\cdots,d \\ x_{i,j}(t) & otherwise. \end{cases}$

      Evaluate the object value of experiment individual
        Exert selection operation between individual and individual to generate new individual

            $Xi(t+1) = \begin{cases} U_i(t+1), & if \quad F(U_i(t+1)) < F(X_i(t)) \\ Xi(t), & otherwise \end{cases}$

        Update $X_{best}$
      End While
  Output $X_{best}$ and objective value
  END
```

Where, $r1,\ r2 \in \{1,2, \ldots N\}$ are different to each other;
$F \in [0,2]$ is zoom scale factor which is used to control the ratio of father generation differential vector $X_{r1}(t) - X_{r2}(t)$
$X_{best}(t)$ is the basic vector after exerting disturbance. The generation will share the information of best solution;
rand(*j*) is the random variable of *j*th separate random in [0,1]
rand(*i*) is the random number in aggregation $\{1,2, \ldots , d\}$
$CR \in [0,1]$ is crossover parameter to control scatter dimension of population.

```
  STEP 3 Output optimisation results.
  END
```

As a new evolutionary technique, differential evolution (DE) was mainly proposed for continuous optimisation. DE is a population-based globally evolutionary algorithm, which uses a simple operator to create new candidate solutions and one-to-one competition scheme to select new candidates greedily. DE has the merits of memorising the best solution and sharing the group information in the candidate. DE adopts the differential operation to generate new candidates. The compete mechanism DE used is the one to one greedy selection (Liu Bo and Ma Hannan 2008). The real number code is used in the algorithm, so the complicated gene operation in GA algorithm is avoided.

In the first place, DE generates the initialisation population random in the feasible solution in the problem space. The current populations are exerted mutation and crossover operation to generate a new population. Second, two populations are selected with one to one mechanism by greedy theory to gain the ultimate new generation. The best solution in DE is obtained by the distance of individuals in population and direction information. So each individual in population changes the search action by altering the direction of differential item and step. In each generation, mutation and crossover operation is exerted to individuals to generate a new population. Finally, the next generation is generated by selection in the old generation.

From the chart, we can see that PSO provides initial solution for DE during the hybrid search process. Such hybrid algorithms can be converted to traditional DE by setting swarm size to one particle. HPSO implements easily and reserves the generality of PSO and DE. Moreover, such HPSO can be applied to many combinatorial optimisation problems by simple modification.

Three objective values are used to measure the effectiveness of the schedule. Those objectives are:

(1) Minimise total completion time of all jobs or the makespan;
(2) Minimise total number of rejects – the total number of rejects can be calculated by adding all the number of scrap produced:

$$F1 = \sum_{l=1}^{n} Y_l^s = \sum_{l=1}^{n} Y_l^o \left(\sum_{j=l}^{op} k_j^s \right) \tag{9}$$

(3) Minimise total processing cost – the total processing cost can be calculated by summing all the processing costs for all the operations;

$$F2 = \sum_{l=1}^{n} N_l^i \sum_{j=1}^{op} \left[k_{lj}^i X_{lj}^i - k_{lj}^s X_{lj}^s + k_{lj}^i f\left(Y_{lj}^i \right) \right] \tag{10}$$

For Equations (9) and (10): Y_l^s is the scrap unit, Y_i^o is the output unit, k_j^i is the input technological coefficient per unit output, k_j^s is the scrap technological coefficient per unit, X_j^i, X_j^s are the unit average cost of input and scrap respectively, $f\left(Y_j^i\right)$ is the processing cost per unit, n is the total number of jobs, 1 is the job number, op is the total number of operations and N^i is the number of input units. Equations (9) and (10) are detailed in Singh (1996).

3.3. *Fuzzy interference system for machine balance*

Each machine will be assigned an imaginary MTF value randomly. The values will be in a range between

0 to 50 units. These values are represented by the triangular membership function as shown in Figure 11. Where, (c_{+a},c_{+b}), (c_{-a},c_{-b}) and (c_{ia},c_{ib}), ($i = 0,1, \dots ,4$) is the border value for corresponding membership function.

A set of fuzzy sets is set up when the input is fuzzified. It is denoted by $\{NB,NM,NS,Z,PS,PM,PB\}$, where, NB-Negative big, NM-negative middle, NS-negative small, Z-zero, PS-positive small, PM-positive middle and PB-positive big.

The fuzzy membership function *f* for each fuzzy set is represented as below.

$$f_{NB} = \begin{cases} 1 & if \quad x < c_{-a} \\ \frac{c_{-b}-x}{c_{-b}-c_{-a}} & if \quad c_{-a} \leq x \leq c_{-b} \\ 0 & if \quad x > c_{-b} \end{cases}$$

$$fi, \Delta = \begin{cases} 0 & if \quad x < c_{ia} \\ 2\frac{x-c_{ia}}{c_{ib}-c_{ia}} & if \quad c_{ia} \leq x \leq \frac{c_{ia}+c_{ib}}{2} \\ 2\frac{c_{ib}-x}{c_{ib}-c_{ia}} & if \quad \frac{c_{ib}+c_{ia}}{2} < x \leq c_{ib} \\ 0 & if \quad x > c_{ib} \end{cases}$$

$$f_{PB} = \begin{cases} 0 & if \quad x < c_{+a} \\ \frac{x-c_{+b}}{c_{+b}-c_{+a}} & if \quad c_{+a} \leq x \leq c_{+b} \\ 1 & if \quad x > c_{+b} \end{cases}$$

The input for the inference process is a single number set given by the antecedent and the output by a fuzzy set. The fuzzy reasoning methods that we used are built-in methods supported by Matlab Fuzzy Logic Toolbox (The Mathworks 1998), which truncates the output of fuzzy sets.

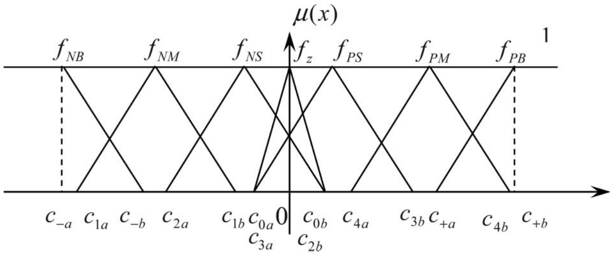


Figure 11. Membership functions.

The fuzzy rules are in the format as follows.

if <antecedent, related to the MTF> then <consequent, the machine reliability>

Rules used in the FL are:

If (MTF is very low) then (machine reliability is very low)
If (MTF is low) then (machine reliability is low)
If (MTF is medium) then (machine reliability if standard)
If (MTF is high) then (machine reliability is high)
If (MTF if very high) then (machine reliability is very high)

A decision is based on the testing of all the rules in an FL. A combination of the rules is necessary for decision making. Aggregation is the process by which the fuzzy sets that represent the outputs of each output are combined into a single fuzzy set. The input of the aggregation process is the list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable.

The last step is to defuzzify the aggregate output fuzzy set. The result from this step is a single number. In this work the result is the reliability index. Figure 12 shows the example of the output of FL. The first column shows how the input variable is used in the rules. The input variable is shown at the top, i.e. MTF = 21.2. The second column shows how the output variable, i.e. machine reliability (mac_reliability), is used in the rules. Each row of plots represents one rule. The five plots (i.e. the triangle plots) in the input column show the membership functions referenced by the antecedent, or if part of each rule. The second column of plots shows the membership functions referenced by the consequent, or the then part of each rule. The shaded input plots represent the MTF value, 21.2 belongs to the membership function A2 and A3. The truncated output plots show the implication

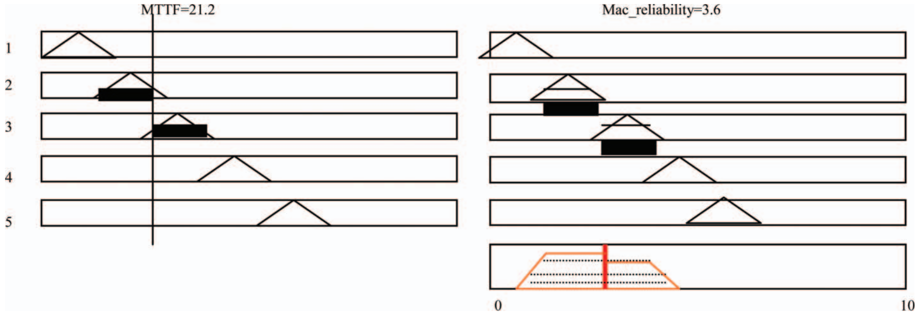


Figure 12. Example of the input-output of the fuzzy inference system.

process. It has been truncated to exactly the same degree as the antecedent. The lower plot of the output column is the resultant aggregate plot. A defuzzified output value is shown by the line passing through the aggregate fuzzy set. In this example, if the MTF is 21.2, the machine reliability index is 3.6. The index is the defuzzification result of the FL. It is in the range of 0 to 10.

In a case where numeric input is not available, FL can be modified to accept linguistic or fuzzy input (such as 'low'). This is one of the advantages of using FL in integrating machine capability to scheduling.

4. Results and discussion

In our experiment, utilising test datum used by Morad (Morad and Zalzala 1997), the acceptable value for the total completion time is 1287 units of time, the total number of rejects is 12 units and total processing cost is 678 units of dollar. Figure 13 represents the final schedule in a Gantt chart form.

In the above experiment, it can be seen that machine 23 is left empty without doing any processing. In this case the unreliable machine is not given a job. In a real manufacturing environment, this case is unfavourable. Even though machine 23 is unreliable, in this case, it should not be left without doing any jobs. At least some operations can be assigned to this machine, in order to avoid any wasting resources.

To overcome this problem, another approach is proposed in this work. The HPSO will be used to balance the load for each machine. The load will be distributed based on the machine capability, which is measured by the reliability index. For the most reliable machine, the load given to the machine will be more compared to the unreliable one. The load on each machine is measured by the machine utilisation, i.e. the percent of time the machine is being utilised. For this purpose, the ranking values from the FL are being grouped into three levels for penalty purposes:

(1) Unreliable when the machine reliability index is less or equal to 2;
(2) Standard when the machine reliability index is equal to 3;
(3) Reliable when the machine reliability index is more than 3;

A list of if-then rules has been developed to penalise machines based on its utilisation. For unreliable machines, the machine utilisation should be less or around 20, for the standard machine the machine utilisation should be less or around 40, and for the reliable machine, the machine utilisation should be around total utilisation. The utilisation is measured in percentage. It can be summarised as:

```
If machine reliablity $\leq 2$
    If machine utilisation $> 20\%$ or machine utilisation $= 0$
        Penalty(x) $= 10$
    Else if machine reliability $= 3$
    If machine utilisation $> 40\%$ or machine utilisation $= 0$
        Penalty(x) $= 70$
    Else
    If machine utilisation $>$ total utilisation or machine utilisation $= 0$
        Penalty $= 100$
End
```

If these machines exceed the utilisation limits, it will be penalised. This criteria will be checked for each particle in each generation. The HPSO will try to minimise the total penalty value until the best particles which present the most balanced load is achieved.

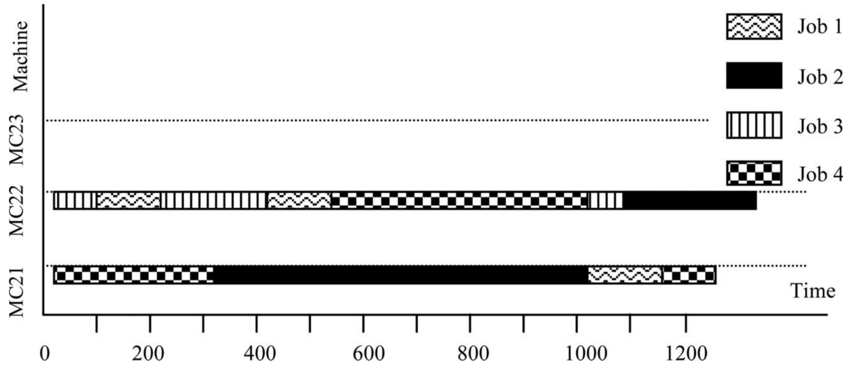


Figure 13. Gantt chart of the schedule.

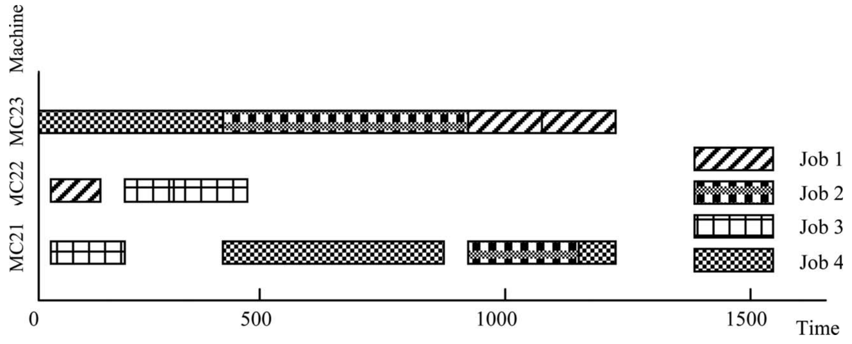


Figure 14. Gantt chart for improved method.

Another simulation, using the same data set, has been conducted for the proposed approach. Instead of assigning the MTF values randomly, in this simulation each machine is being assigned an imaginary value of MTF. The values are 30, 10 and 50 for machine 21, machine 22 and machine 23, respectively. Based on the FL, the ranking for the machines are 3, 1 and 5 for machine 21, machine 22 and machine 23, respectively. After 50 generations, the optimised schedule is shown in Figure 14.

From the Gantt chart shown in Figure 14, the machine utilisation can be calculated as:

$$\begin{aligned}&\mathit{Machine\ 21\ utilisation} = 970/2910 = 33.3\%\\&\mathit{Machine\ 22\ utilisation} = 415/2910 = 14.3\%\\&\mathit{Machine\ 23\ utilisation} = 1525/2910 = 52.4\%\end{aligned}$$

From the calculation, the unreliable machine (i.e. machine 22) has the least load compared with other machines. This machine is being utilised up to 14.2% of the time. The standard machine, that is machine 21, uses 33.3% of its time processing the jobs. Lastly, the most reliable machine, that is machine 23, has the most loads compared to other machines. From the calculation, the utilisation is 52.4%. From the results, it shows that each machine received loads within a range respective to its capability.

In this experiment, the total penalty value becomes the objective function of the HPSO. The HPSO is minimising the objective individually. The details of the schedule are shown in Table 1. It shows the order of the jobs, the machines involved in processing operations, operation sequence, start time of the operation and finish time of the operation.

Compared with the solution obtained by the simple PSO and Simulated annealing (SA) (Table 2), it can be seen that the solution is an optimal one and that it uses less time than the simple PSO and SA. The generation process is shown in Figure 15. From this figure, it can be found that the evolution process of the HPSO tends to be stable when the generation reaches more than 50.

Table 1. The detailed information of the schedule.

Job	Machine	Operation	Start time	Finish time
4	3	1	1	406
1	2	1	4	109
3	1	1	2	157
2	3	1	406	702
4	1	2	406	811
1	3	2	702	859
3	2	2	157	362
2	1	2	811	986
4	1	3	859	1021
1	3	3	986	1112
3	2	3	362	467
2	3	3	1021	1180

Table 2. The comparison of different algorithms.

Best rate (100%)	HPSO result	PSO result	SA result
Best	1180	1295	1378
Mean	1221	1396	1482
Maximum	1262	1497	1586
CPU time	48.00"	79.00"	1029.0"

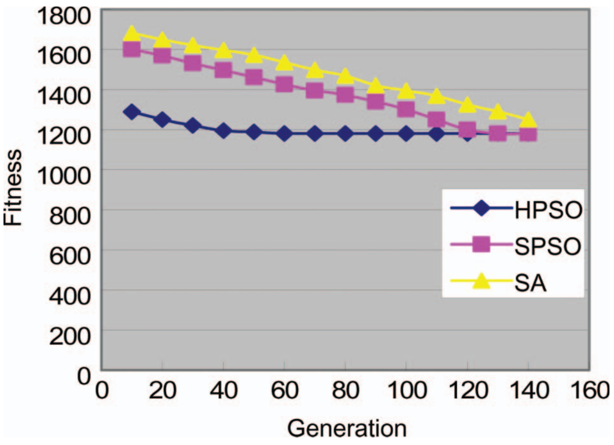


Figure 15. The generation process of the different algorithm.

Table 3. Comparison of the result of different problems with different sizes.

	HPSO		GA	
Tasks	Time	Fitness	Time	Fitness
10	0.030678	602.035	0.048535	637.6933
20	0.145147	1128.599	0.170788	1276.966
30	0.444139	1515.008	0.54533	2152.045
40	1.107601	2591.969	1.395605	3363.448
50	2.487637	4119.71	3.184524	4956.859
60	5.103938	6053.59	6.713829	6861.95
70	6.596156	8991.452	12.86447	9786.616
80	12.23077	10 444.1	23.73169	13 438.71
90	24.20787	15 639.47	41.0847	11 356.78
100	42.45422	19 369.5	57.9913	24 542.87
Total	97.808156	70 455.433	147.730771	82 173.9409
Improvement in fitness (%) 16.63				
Normalised running time 1.511				

When considering the larger size problem, the PSO and SA usually find it hard to obtain the optimal solution in the desired time, so the HPSO is introduced. To test the performance of the HPSO, we randomly produced some problems with different sizes. The result together with the comparison of the PSO without the embedded rule and genetic algorithms (GA) are shown in Table 3.

We compare our results with the GA for 10 fully connected homogeneous processors. Two types of comparisons are carried out based on the results obtained by running each algorithm on this suite of 150 graphs. First, we compare the computational cost for all the random graphs generated by the technique and the average running times of these algorithms. The cost generated by GA technique and HPSO technique for randomly generated process graphs is shown in Figure 15. Each point in the figure is the average of 25 test cases. PSO solution quality is on average 16.63% better than GA. On average, the GA algorithm is 1.511 times slower than PSO technique. These results indicate that the proposed HPSO algorithm is a viable alternative for solving the process planning and scheduling problem.

5. Conclusion and future work

This paper has presented a holonic architecture for dynamic scheduling of manufacturing systems. An integrated process planning and scheduling system of machine product, aimed at realising a flexible production control in holonic manufacturing systems is presented. Instead of choosing alternative machines randomly, this paper proposed the usage of FL to choose the most reliable machine. This is an alternative way to integrate the production capability during scheduling. This paper shows some promising results in integrating production capability and load balancing during scheduling activity. There are a few objectives that could be optimised individually or simultaneously. This will give a choice to the scheduler in determining which objective is the most important.

The great benefit of a holonic architecture for dynamic scheduling of manufacturing systems is the graphical and concise representation of activities, resources and constraints of a operation of scheduling holon in a single coherent formulation, which is very helpful for designers to better understand and formulate scheduling problems. Moreover, it is understood from this research that the FL and HPSO approach has a great potential for solving a variety of complicated scheduling problems in scheduling holon. In this regard, further research is also being undertaken to accommodate complicated situations such as variable task size, variable cycle time, scheduling of mixed batch/continuous plants and implementation of integrated supervisory control and scheduler using the model and algorithm.

Acknowledgements

This research is supported by 863 High Technology Plan Foundation of China (grant NO 2002AA415270), National Natural Science Foundation of China (Contract No. 2001BA201A32) and Natural Science foundation of GANSU province (grant NO3ZS062-B25-033).

References

Administrator, Guide. *Jade administrator's guide* [online]. Available from: http://sharon.cselt.it/projects/jade/doc/administratorsguide.pdf

Axsater, S., 2007. A heuristic for triggering emergency orders in an inventory system. *European Journal of Operational Research*, 176 (2), 880–891.

Babiceanu, R.F. and Chen, F.F., 2009. Distributed and centralised material handling scheduling: Comparison and results of a simulation study. *Robotics and Computer-Integrated Manufacturing*, 25 (2), 441–448. Available from: http://dx.doi.org/10.1016/j.rcim.2008.02.007

Babiceanu, R.F.C., Frank, F., and Sturges, R.H., 2005. Real-time holonic scheduling of material handling operations in a dynamic manufacturing environment. *Robotics and Computer-Integrated Manufacturing*, 21 (4–5), 328–337.

Chan, F.T.S., Kumar, V., and Tiwari, M.K., 2009. The relevance of outsourcing and leagile strategies in performance optimization of an integrated process planning and scheduling model. *International Journal of Production Research*, 47 (1), 119–142.

Chang, W.-D., 2007. Parameter identification of Chen and Lu systems: A differential evolution approach. *Chaos, Solitons and Fractals*, 32 (4), 1469–1476.

Cheng, F.-T.L. and Ching-Tien, 2004. A holonic information exchange system for e-manufacturing. *30th Annual Conference of IEEE Industrial Eelctronics Society,* 2377–2382.

Chyu Chiuh-Cheng, C. and Wei-Shung, 2008. A genetic-based algorithm for the operational sequence of a high speed chip placement machine. *International Journal of Advanced Manufacturing Technology*, 36 (9–10), 918–926.

Clerc, M., 1999. The swarm and the queen: Toward a deterministic and adaptive particle swarm optimization. *IEEE International Congress on Evolutionary Computation*, Vol. 3, 1951–1957.

Coello Coello, C.A., *et al.*, 2004. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8 (3), 256–279.

Cooke, D.L.R., Thomas, R., and Silver, E.A., 2004. Finding effective schedules for the economic lot scheduling problem: A simple mixed integer programming approach. *International Journal of Production Research*, 42 (1), 21–36.

Deen, S.M., 2003. Agent-based manufacturing – advances in the holonic approach. Springer-Verlag, Heidelberg, Germany.

Eberhart, Y.S., 2000. Comparing inertia weights and constriction factors in particle swarm optimization. *Congress on Evolutionary Computation*, 84–88.

FIPA. *Foundation for Intelligent Physical Agents* [online]. Available from: http://www.fipa.org/

Giret, A. and Botti, V., 2006. From system requirements to holonic manufacturing system analysis. *International Journal of Production Research*, 44 (18–19), 3917–3928.

Heragu, S.S., *et al.*, 2002. Intelligent agent based framework for manufacturing systems control. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 32 (5), 560–573. Available from: http://dx.doi.org/10.1109/TSMCA.2002.804788

Hms, Consortium. Holonic Manufacturing System (HMS). Web Site. Available from: http://hms.ifw.uni.hannover.de/

Izui Kazuhiro, N.S., *et al.*, 2008. Enhanced multiobjective particle swarm optimization in combination with adaptive weighted gradient-based searching. *Engineering Optimization*, 40 (9), 789–804.

Jade, Jade. *Jave Agent DEvelopment Framework* [online]. Available from: http://jade.cselt.it/

Jawahar, N., Aravindan, P., Ponnambalam, S., and Etal, G., 1998. Knowledge-based workcell attribute oriented dynamic schedulers for flexible manufacturing systems. *International Journal of Advanced Manufacturing Technology*, 14 (7), 514–538.

Kim Haejoong, J.H.-I. and Park, Jinwoo, 2008. Integrated model for production planning and scheduling in a supply chain using benchmarked genetic algorithm. *International Journal of Advanced Manufacturing Technology*, 39 (11–12), 1207–1226.

Leitao, P. and Restivo, F., 2002. *Holonic adaptive production control systems*. Sevilla, Spain: Institute of Electrical and Electronics Engineers Computer Society, 2968–2973.

Leitao, P. and Restivo, F., 2008. A holonic approach to dynamic manufacturing scheduling. *Robotics and Computer-Integrated Manufacturing*, 24 (5), 625–634. Available from: http://dx.doi.org/10.1016/j.rcim.2007.09.005

Li, J., Lou, and Pei-Huang, 2004. Study on strategies of overcoming deadlocks in ga-based solution for jssp. *Journal of Nanjing University of Aeronautics and Astronautics*, 36 (3), 317–321.

Li, J.G., *et al.*, 2008. Cutting parameters optimization by using particle swarm optimization (pso). *Applied Mechanics and Materials*, 10–12, 879–883.

Li Jianxiang, T., *et al.*, 2004. Scheduling the production of hot rolling steel tube: A rule-based heuristics. *Iron and Steel (Peking)*, 39 (9), 39–45.

Li, W.D. and Mcmahon, C.A., 2007. A simulated annealing-based optimization approach for integrated process planning and scheduling. *International Journal of Computer Integrated Manufacturing*, 20 (1), 80–95.

Liu Bo, Z.X. and Ma Hannan, 2008. Hybrid differential evolution for noisy optimization. *IEEE Congress on Evolutionary Computation,* 587–592, Location/publisher.

Malakooti, B.M., Nima, R., and Yang, Ziyong, 2004. Integrated group technology, cell formation, process planning, and production planning with application to the emergency room. *International Journal of Production Research*, 42 (9), 1769–1786.

Mařík, V., Vyatkin, V., and Colombo, A.W., 2007. Holonic and multi-agent systems for manufacturing. *3rd International Conference on Industrial Applications of Holonic and Multi-agent Systems, HoloMAS 2007*, Springer.

Masood, T., 2004. Applications of artificial intelligence in industrial engineering. *Proceedings of the 5th International Conference on Quality, Reliability, and Maintenance,* 217–220.

Mcfarlane, D.C. and Bussman, S., 2000. Developments in holonic production planning and control. *Production Planning and Control*, 11 (6), 522–536. Available from: http://dx.doi.org/10.1080/095372800414089

Monch Lars, S.R., *et al.*, 2007. Genetic algorithm-based subproblem solution procedures for a modified shifting bottleneck heuristic for complex job shops. *European Journal of Operational Research*, 177 (3), 2100–2118.

Morad, N. and Zalzala, A.M.S., 1997. *Optimisation of cellular manufacturing systems using genetic algorithms*. University of Sheffield.

Pedersen, C.R., Rasmussen, R.V., and Andersen, K.A., 2007. Solving a large-scale precedence constrained scheduling problem with elastic jobs using tabu search. *Computers and Operations Research*, 34 (7), 2025–2042.

Petrovic Sanja, F.C. and Petrovic, D., 2008. Sensitivity analysis of a fuzzy multiobjective scheduling problem. *International Journal of Production Research*, 46 (12), 3327–3344.

Programmer Guide. Jade programmer's guide [online]. Available from: http://sharon.cselt.it/projects/jade/doc/programmersguide.pdf

Qian Bin, W.L., *et al.*, 2008. A hybrid differential evolution method for permutation flow-shop scheduling. *International Journal of Advanced Manufacturing Technology*, 38 (7–8), 757–777.

Rais, S., Sugimura, N., and Kokubun, A., 2002. Study on process planning system for holonic manufacturing – (selection of machining sequences and sequences of machining equipment). *JSME International Journal Series C-Mechanical Systems Machine Elements and Manufacturing*, 45 (2), 527–533. Available from: <Go to ISI>://000177338900018.

Reddy, M.M. and Ponnambalam, S.G., 2003. A ga-sa multiobjective hybrid search algorithm for integrating lot sizing and sequencing in flow-line scheduling. *International Journal of Advanced Manufacturing Technology*, 21 (2), 126–137.

Shao Xinyu, L.X., *et al.*, 2009. Integration of process planning and scheduling-a modified genetic algorithm-based approach. *Computers and Operations Research*, 36 (6), 2082–2096.

Shrestha, R., *et al*., 2008. A study on integration of process planning and scheduling system for holonic manufacturing with modification of process plans. *International Journal of Manufacturing Technology and Management*, 14 (3–4), 359–378. Available from: http://dx.doi.org/10.1504/IJMTM.2008.017733

Shukla, S.K., Tiwari, M.K., and Son, Y.J., 2008. Bidding-based multi-agent system for integrated process planning and scheduling: A data-mining and hybrid tabu-sa algorithm-oriented approach. *International Journal of Advanced Manufacturing Technology*, 38 (1–2), 163–175.

Singh, N., 1996. *Systems approach to computer-integrated design and manufacturing*. New York: John Wiley.

Sormaz, D.N., Arumugam, J., and Rajaraman, S., 2004. Integrative process plan model and representation for intelligent distributed manufacturing planning. *International Journal of Production Research*, 42 (17), 3397–3417.

Stahmer, B.P. and Schwaiger, A., 2004. Holonic probabilistic agent merging algorithmed. *IEEE/WIC/ACM International Conference on Intelligent Agent Technology,* 409–412.

The Mathworks, I., 1998. *Fuzzy logic toolbox user's guide*. Available from http://www.pdfee.com/fuzzy-logic-toolbox-users-guide.html

Turgay, S.K. and Taskin, H., 2007. Modelling and simulation of mrp ii activities in multi agent systems. *Production Planning and Control*, 18 (1), 25–34.

Valckanaers, P., *et al*., 1997. Holonic manufacturing system. *Integrated Computer-Aided Engineering*, 4 (3), 191–201.

Van Brussel, H., Wyns, J., and Valckenarers, P., 1998. Reference architecture for holonic manufacturing systems:Prosa. *Computer in Industry*, 37 (2), 255–274.

Van Den Bergh, F. and Engelbrecht, A.P., 2004. A cooperative approach to participle swam optimization. *IEEE Transactions on Evolutionary Computation*, 8 (3), 225–239.

Vicente, B. and Adriana, G., 2008. *Anemona: A multi-agent methodology for holonic manufacturing systems* (Springer series in advanced manufacturing). New York: Springer.

Wang, G.L., Lin, L., and Zhong, S.S., 2008. Task-oriented hierarchical planning and scheduling for discrete manufacturing enterprise. *Applied Mechanics and Materials*, 10–12, 114–120.

Yin Xiao-Feng, C.T.-J. and Wang, Feng-Yu, 2004. A rule-based heuristic finite capacity scheduling system for semiconductor beckend assembly. *International Journal of Computer Integrated Manufacturing*, 17 (8), 733–749.

Yu Haibin, L.W., 2001. Neural network and genetic algorithm-based hybrid approach to expanded job-shop scheduling. *Computers and Industrial Engineering*, 39 (3–4), 337–356.

Zhang, Y.F., Saravanan, A.N., and Fuh, J.Y.H., 2003. Integration of process planning and scheduling by exploring the flexibility of process planning. *International Journal of Production Research*, 41 (3), 611–628.