ELSEVIER

Computers in Industry 37 (1998) 213–231

COMPUTERS IN INDUSTRY

Holonic manufacturing scheduling: architecture, cooperation mechanism, and implementation

Ling Gou [a,1], Peter B. Luh [b,*], Yuji Kyoya [c,2]

[a] *Operations Research Department / 709, Delta Technology, 1001 International Boulevard, Atlanta, GA 30354-1801 USA*
[b] *Department of Electrical and Systems Engineering, University of Connecticut, Storrs, CT 06269-2157 USA*
[c] *Systems and Software Engineering Laboratory, Toshiba, 70 Yanagi-cho, Saiwaiku, Kawasaki 210, Japan*

Abstract

A Holonic Manufacturing System (HMS) is a manufacturing system where key elements, such as machines, cells, factories, parts, products, operators, teams, etc., are modeled as 'holons' having *autonomous* and *cooperative* properties. The decentralized information structure, the distributed decision-making authority, the integration of physical and informational aspects, and the cooperative relationship among holons, make the HMS a new paradigm, with great potential for meeting today's agile manufacturing challenges. Critical issues to be investigated include how to define holons for a given problem context, what should be the appropriate system architecture, and how to design effective cooperation mechanisms for good system performance. In this paper, holonic scheduling is developed for a factory consisting of multiple cells. Relevant holons are identified, and their relationships are delineated through a novel modeling of the interactions among parts, machines, and cells. The cooperation mechanisms among holons are established based on the pricing concept of market economy following 'Lagrangian relaxation' of mathematical optimization, and cooperation across cells is performed without accessing individual cells' local information nor intruding on their decision authority. The system also possesses structural recursivity and extendibility. Numerical testing shows that the method can generate near-optimal schedules with quantifiable quality in a timely fashion, and has comparable computational requirements and performance as compared to the centralized method following single-level Lagrangian relaxation. The method thus provides a theoretical foundation for guiding the cooperation among holons, leading to globally near-optimal performance. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Intelligent manufacturing systems; Holonic systems; Agent-based systems; Lagrangian relaxation; Manufacturing scheduling

1. Introduction

1.1. Holonic manufacturing systems (HMS)

The following trends have been widely documented [37,30,31]:

- Manufacturing is undergoing a 'paradigm shift' from mass production to *semi-customized* production to meet increasingly diverse demand.

[*] Corresponding author. E-mail: luh@brc.uconn.edu
[1] E-mail: ling.gou@delta-air.com
[2] E-mail: kyoya@ssel.toshiba.co.jp

- The 'made-in-house' mind-set is gradually changing to a more *collaborative mentality*, and various entities are teaming up to speed up product development and manufacturing, to reduce risk, and to penetrate local markets.
- Effective and efficient *cooperation* becomes critical to reap the full benefits of collaboration.
- Centralized control of various entities with different information, expertise, decision-making authorities, and objectives is almost impossible. Effective and efficient cooperation will be a key challenge for organizations who want to thrive in the increasingly competitive market.

In the new paradigm, technologies alone are not sufficient for superior system performance, but have to be integrated with organizational structures that can fully realize their benefits. The traditional structure with a fixed and vertically integrated hierarchy should be changed to a more agile structure featuring fewer levels and the distribution of authority [12,15,41]. For example, installing billions of dollars worth of robots and computer-aided manufacturing equipment at General Motors in early 1980s without appropriate organizational changes proved to be abortive for improving productivity [15]. In contrast, within 3 years of being taken out of Westinghouse hierarchy and integrated into ABB's decentralized structure in 1989, the U.S. Relays Unit doubled its operating profits, and became a growing company with significant potential [4]. Studies on flexible manufacturing systems (FMS) also indicate that hierarchical structures impose limitations on reconfiguration, reliability, and expansion because of the tight coupling among decision modules in the hierarchy. Furthermore, the complexity of hierarchical systems grows rapidly with size [19,13,1].

To meet the challenges posed by the new paradigm, some key requirements for manufacturing systems are identified as follows.

- *Localized information and decision-making authority*. Individual elements should have localized information and decision-making authority, and possess the capability to create and execute their plans and/or strategies.
- *Cooperativeness*. Individual elements should not operate with absolute autonomy. Rather, they should cooperate with others, operate within system constraints, and adjust their behaviors according to coordination information.
- *Integrability*. The combining of physical and informational aspects of elements is needed for ease of integration of these elements into a system. One should be able to set up a manufacturing system by integrating necessary elements, and be able to easily and quickly reconfigure the system by adding, adapting, adjusting, and deleting these elements.
- *Recursivity*. A manufacturing system may consist of elements at various levels, and elements at different levels should be structured along similar lines for meaningful integration from level to level. For example, cells and factories should be designed to share similar architectural features such as information structure and communication protocols to ensure proper cooperation among them.

A HMS is a way to organize manufacturing activities to meet the above requirements and to overcome the difficulties faced by conventional systems. In an HMS, key elements, such as machines, cells, factories, parts, products, operators, teams, etc., have *autonomous* and *cooperative* properties. These elements are called 'holons,' coming from the Greek holos, meaning whole, with the suffix 'on' which, as in proton or neutron, suggests a particle or part. The word was coined by Koestler [21] to describe the hybrid nature of sub-wholes in living organisms and social organizations. A holon in a hierarchy (holarchy) is characterized by *autonomy* and *cooperation* (capability for integration), and consists of a physical component and an informational component. The physical component corresponds to the holon's physical entity such as a product or a machine. The informational component contains information, has decision-making capabilities, and possesses mechanisms for managing the physical component and for cooperating with other holons. Holons may have informational components only. Holons also possess structural *recursivity*, i.e., a small holon can be part of a large holon. For example, a 'machine holon' is part of a 'cell holon,' and a cell holon is part of a 'factory holon'. The HMS concept has been investigated by the HMS Consortium [34], one of the six test cases of the Intelligent Manufacturing System (IMS) feasibility study program [20], and is now a full-scale IMS project. The objective of the

Consortium is to explore the benefits of holonic organizations, e.g., efficient use of available resources, flexibility, and adaptability in the face of changes, etc.

It is very easy to under-estimate the importance of the holonic concept. Is it new jargon that is fashionable for one or two years and will soon die out? A precious lesson can be learned by comparing conventional programming languages (e.g., Basic, FORTRAN, Pascal, and C) to object-oriented programming languages (e.g., C + + and Smalltalk). The encapsulation of data and methods within individual objects (autonomy) and the clear delineation of responsibilities and the relationships among them (cooperation) make object-oriented languages the overwhelmingly preferred languages for new software development in the past decade (before the rise of platform independent languages such as JAVA). The lack of proper cooperation within a shop or within an enterprise because of the intrinsic complexity has often been a nightmare for all who are involved. Holonic systems are built on whole-part relationships, and are managed in a distributed manner by system elements or holons, as opposed to being controlled by a centralized mechanism. In addition, the holonic concept matches the object-oriented programming concept, and requires the clear delineation of information, responsibilities, and interfaces of individual holons. It further pushes the integration of physical and informational aspects of system elements, enabling the easy creation, managing, and reconfiguration of systems. Therefore HMS points a promising direction for meeting today's manufacturing challenges, and it is conceivable that holonics will become the dominating trend for manufacturing and beyond, whatever name it is called. There are, however, major issues to be resolved, including how to define holons for a given problem context, what should be the appropriate holonic architecture, how to design effective cooperation mechanisms for good system performance, and implementation issues such as the plug-and-play capability for fast deployment.

1.2. Scope of this paper

This paper presents holonic scheduling for a factory consisting of multiple cells. The study was motivated by the design and implementation of a scheduling system for the production of Toshiba's gas insulated switchgears—circuit breakers used by electric utility companies to control the flow of high voltage electric current. The manufacturing is characterized by low-volume/high-variety products and multiple production stages. Although several simplifications have been made, the essential features of the manufacturing process are captured so that practical issues related to holonic scheduling are addressed and appropriate techniques developed. From the holonic viewpoint, scheduling involves cooperation among system elements or holons within the factory. This paper concentrates on developing the holonic models, delineating the coordination information, and establishing the cooperation mechanisms among holons for near-optimal system performance.

2. Literature review and paper overview

2.1. Related research

Manufacturing scheduling is very important because of its direct link to product delivery, inventory levels, and machine utilization. Effective scheduling, however, has been proved to be extremely difficult because of the combinatorial nature of integer optimization and the large size of practical problems [7]. In practice, material planning systems (e.g., MRP or MRP II) are often used for high level production planning and scheduling [17]. 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 utilization (e.g., Blackstone et al. [6]). Several artificial intelligence (AI) approaches are also built on scheduling rules (e.g., Kusiak [22]). Schedules obtained by heuristics, however, are often of questionable quality, and there is no good way to systematically improve the schedules generated.

As mentioned, the holonic manufacturing concept has been studied by the IMS-HMS consortium. Similar research has been performed under the banner of 'agent-based manufacturing.' A computer agent is a software object representing an entity, and can roughly be classified into three kinds: network agent, user agent, and Distributed Artificial Intelligence (DAI) agent. Network agents enable network navigation. A user agent observes the computer user, and acts on his/her behalf. DAI agents are distributed and sophisticated reasoning agents, and cooperate with others for problem solving. Many agent-based systems are heterarchical, and consist of groups of identical or complimentary agents acting together to solve problems based on their value systems and communication abilities [2]. 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 (especially a DAI agent) may have autonomous and cooperative properties, and is the building block of the system.

A popular approach is the ‘contract net protocol,’ where the protocol process involves bidding, negotiation, and cooperation [36]. A method for dynamic task assignment in a cellular manufacturing system was presented in Shaw [35], where ‘request for bid’ messages are broadcast to cells, and cells evaluate operation specifications, and prepare and submit bids. The cell that optimizes a predefined criterion is selected to perform the operation, without much consideration of overall system performance. Negotiation processes (inter-agent bidding) were developed using various heuristic rules to iteratively improve system performance (e.g., Refs. [13,23]). These processes, however, can be slow, and unresolved negotiation may lead to deadlocks [23]. A cooperative approach was presented in Duffie and Prabhu [14], where agents modify their local schedules using a look-ahead heuristic, and find a set of local schedules that may collectively achieve the global goals. Simulation studies with 2 machines and 12 parts for a total 16 operations were performed on an experimental system consisting of three microcomputers connected by a local area network. A multi-agent system AARIA (Autonomous Agents for Rock Island Arsenal) was developed for an Army manufacturing facility [32,3]. In the system, the foreground scheduler dispatches newly arrived parts and reacts to machine failures, and schedule optimization for the entire facility is performed in the background by using a cost-based method. Improved performance in terms of system agility and equipment utilization has been reported.

Recent research shows that agent-based systems are moving from the pure heterarchical structure to the so-called quasi-heterarchical structure or holarchy (e.g., Refs. [10,28]). In such a structure, agents, also called holons by many authors (e.g., Ref. [28]), can form hierarchies and adjust their behaviors according to the coordination information.

In the HMS literature, holonic architectures and their properties, including the autonomy, cooperativeness, and recursivity were discussed in Mathews [27]. The holonic manufacturing concept was compared with bionic and fractal manufacturing in Tharumarajah et al. [39], and the trend for an organization moving towards a set of autonomous and cooperative modules was concluded. Holons for real time scheduling were identified and implemented as objects using Smalltalk in Moriwaki et al. [29] and Sugimura et al. [38]. Beyond equipment and part objects, a ‘coordinator object’ was introduced to deal with conflicts among objects using heuristics. Simulation studies show that the system can cope with random disruptions such as machine breakdowns and arrivals of high priority parts. A schedule execution system for a pilot flexible assembly testbed was presented in Bongaerts et al. [8] and Valckenaers et al. [40], where three heuristic-based control modes including hierarchical, heterarchical, and holonic, were used to execute pre-generated schedules. Benchmark results show that holonic control has potential for improved system performance, and the holonic architecture can dynamically adapt during the life cycle of the system.

One of the key issues in an HMS is how to coordinate the activities of holons to improve overall system performance. Most of the research cited above uses various heuristic-based approaches for negotiation and cooperation, and system performance is generally difficult to evaluate. The only exception is the mapping of Lagrangian relaxation onto a holonic architecture for scheduling a simulated robotic assembly test bed [18,16]. Lagrangian relaxation is a mathematical optimization technique, and has recently been used to develop a decomposition and coordination approach for the near-optimal scheduling of several types of manufacturing systems, including identical machines and job shops [26,42]. The method has been mapped onto a holonic architecture in Gou et al. [16] and Hasegawa et al. [18], and testing of simplified cases shows that the method can generate near-optimal schedules with quantifiable quality in a timely fashion, and can effectively reconfigure a schedule in case of random disruptions.

2.2. *Overview of the paper*

The Lagrangian relaxation-based approach is further developed for holonic factory scheduling in this paper, where the decomposition and coordination nature of the approach is fully exploited to meet the holonic requirements of limited information accessibility and individual decision-making authority. In Section 3, a novel holonic model of factory scheduling is developed, where interactions among cells are modeled at the factory level, and detailed processing requirements reside within individual cells. In Section 4, the information necessary for effective coordination is delineated, and the cooperation mechanisms among holons are established based on Lagrangian relaxation for near-optimal system performance. Lacking a distributed computation platform and a physical factory environment, the communication protocols, related asynchronous implementation issues, and the integration of physical and informational components of holons will not be addressed.

The scheduling system is implemented by using the object-oriented language C + +, with holons' informational components implemented as software objects residing in a single computer. Testing has been performed using data drawn from Toshiba's HamaKawasaki Factory, where more than one hundred parts are processed by three cells with a total of 22 machine types and 87 machines. Numerical results presented in Section 5 show that the method can generate high quality schedules in a timely fashion, and has comparable computational requirements as compared to the single-level Lagrangian relaxation method. Computational speedup can be obtained by better initialization and distributed processing. The method developed also possesses structural recursivity, and combined with the integrability property of holons, enables high system flexibility and extendibility. Lagrangian relaxation thus provides a theoretical foundation for guiding the cooperation among holons, leading to near-optimal system performance.

3. Holonic modeling of factory scheduling

The factory under consideration has several cells, and each cell contains multiple types of machines. Products to be manufactured have multiple production stages. For simplicity of presentation, each production stage is assumed to be associated with a particular cell, and individual products at a particular cell are called 'parts.' The system architecture is shown in Fig. 1, where a factory with two cells and a product consisting of two parts are depicted. From the holonic viewpoint, each cell is responsible for scheduling machines and parts within the cell, and coordination across cells should be performed based on the interactions among cells without accessing individual cells' local information nor intruding on their decision authority. The factory- and cell-level coordinators are therefore introduced to generate coordination information at the factory- and cell-levels, respectively, as will be discussed later. The system is structurally recursive, i.e., the factory contains cells, products, and the factory coordinator; a cell contains machine-types, parts, and a cell coordinator; and a product contains parts. In addition, a part is a member of a product and a member of a cell, and this structure is called 'reticulated' (i.e., networked) hierarchy by Koestler [21]. By extending our previous work on job shop scheduling [26], a novel holonic model is developed in the following, where interactions among cells are modeled at the factory level, and detailed processing requirements reside within individual cells. For ease of reference, a list of symbols used in this paper is provided in Appendix A.

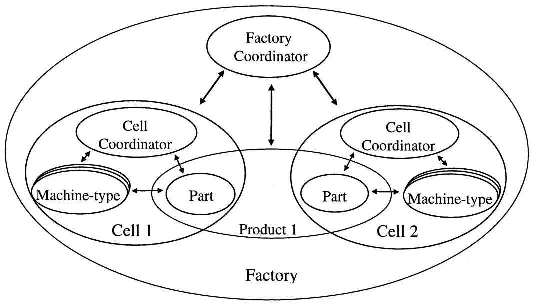


Fig. 1. System architecture.

3.1. Factory level

Assume that the factory has $|S|$ cells. There are $|I|$ products to be scheduled, and each product has to go through a series of production stages according to a specified routing. For simplicity, alternative routings and assembly are not considered, although there is no theoretical difficulty to extend the method to handle these features. As mentioned, individual products at a particular cell are called parts. Product i, $i \in I$, processed at cell s is denoted as part (i,s), and the set of possible immediate subsequent parts is denoted as I_i^s, and is a singleton or an empty set. An illustration is shown in Fig. 2. There is no deadlock since parts can be put into buffers of infinite capacity if machines are tied up. The scheduling horizon has K equal time intervals indexed by k, $k \in [0, K-1]$. For example, a time horizon of 2 weeks has $K = 80$, assuming 40 working hours per week and each time interval (time resolution) representing 1 h.

To integrate the physical and informational aspects of these elements and delineate their relationships, five types of holons at the factory level are identified, including *Product*, *Part*, *Cell*, *Factory Coordinator*, and *Factory* holons.

A *Product* holon corresponds to a product to be manufactured. An order for a particular product i enters the factory as a *Product* holon, with its informational component containing specifications such as production routing (which cells to go through, but does not include detailed processing requirements within individual cells), arrival time of raw material a_i, due date d_i, desired raw material release time $\bar{b}_i$, priority (earliness and tardiness weights β_i and ω_i to be explained later), and mechanisms for coordinating the manufacturing process. Other information such as processing status is added during the manufacturing process. The physical component of the holon grows from nothing to an intermediate product and then to the finished one. It disappears from the factory when the product is shipped to the customer. The informational component, however, is kept in archive for later reference.

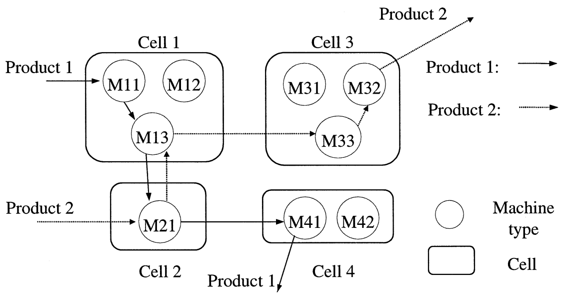


Fig. 2. Process plans for products 1 and 2.

A *Part* holon corresponds to a particular stage of a product, resides in the associated cell, and is a building block of the *Product* holarchy. Its physical component is an intermediate product. The informational component contains detailed process plan, processing status, and mechanisms for coordinating the manufacturing process within the associated cell.

A *Cell* holon corresponds to a cell, and contains machine types, parts, and a cell coordinator. Its functions include cooperating with *Product* and *Factory Coordinator* holons to fulfill the required production process, as will be elaborated later.

From the scheduling viewpoint, the interactions among the above *Product*, *Part* and *Cell* holons are specified by production routings, and can be described by the 'part precedence constraints' across cells as follows.

Part Precedence Constraints. A part cannot be started until its preceding part is finished. Equivalent, the beginning time of part (i,s') must be greater than or equal to the completion time of its preceding part (i,s) as delineated by the process plan, plus a required 'timeout,' i.e.,

$$c_i^s + S_i^{ss'} + 1 \le b_i^{s'}, \text{with } (i,s') \in I_i^s, \tag{1}$$

where $b_i^{s'}$ and c_i^s represent the beginning time of (i,s') and the completion time of (i,s), respectively. Parameter $S_i^{ss'}$ is the required 'timeout' for product i between cells s and s', representing processes not explicitly modeled in the problem formulation (e.g., the transportation time for product i in-between the two cells). In the above, '1' comes from the discretization process. For the first part of product i, the left-hand-side of (1) is degenerated to a_i, the raw material arrival time of the product.

The *Factory Coordinator* holon is introduced to coordinate the scheduling activities across cells, and has informational component only. It gathers the status of *Cell* and *Product* holons, and generates coordination information to guide these holons' scheduling activities for overall system performance.

The *Factory* holon corresponds to the entire factory, and is a holarchy consisting of *Cell*, *Product*, and the *Factory Coordinator* holons.

The objective of scheduling is to ensure on-time product delivery while keeping low work-in-process inventory. This is represented by minimizing ‘product penalties,’ i.e., the sum of weighted quadratic penalties for missing product delivery dates and penalties for releasing raw materials too early,

$$\mathcal{J} \equiv \sum_i \left\{ \omega_i T_i^2 + \beta_i E_i^2 \right\}. \tag{2}$$

In the above, T_i is the tardiness of product i defined as the amount the product completion time c_i (the completion time of product i at its last cell) passes its due date d_i, i.e., $T_i \equiv \max\{0, c_i - d_i\}$. As explained in Ref. [26], this weighted quadratic tardiness penalties account for the values of products, the importance of meeting due dates, and the fact that a product becomes more critical with each time unit passing its due date (the last cannot be reflected by linear tardiness penalties). The product earliness E_i is similarly defined as the amount the product beginning time b_i (the beginning time of product i at its first cell) leads the desired product release time $\bar{b}_i$, i.e., $E_i \equiv \max\{0, \bar{b}_i - b_i\}$. Parameters ω_i and β_i are weights associated with those penalty terms. The above penalties define a time window in which the product can be scheduled without penalty. The objective function (2) is to be minimized by selecting part beginning and completion times $\{b_i^s, c_i^s\}$ subject to part precedence (1) and raw material arrival time constraints (the processing of a product cannot be started before the arrival of raw materials). Part beginning and completion times are not totally independent decision variables, rather, they are determined by solving cell-level problems to be presented next.

3.2. Cell level

Similar to what was presented in Section 3.1 on the factory and products, a cell contains multiple machine types, and a part requires a series of operations for completion. For cell s, there are $|H_s|$ machine types, and each machine type may consist of one or multiple identical machines. The number of machines available for type $h_s \in H_s$ at each time period k is a given integer, denoted as M_{kh_s}. There are $|I_s|$, $I_s \subseteq I$, products to be scheduled in cell s, each requiring a series of operations for completion. The j-th operation of part (i,s) is denoted as operation (i,s,j), $1 \le j \le J_{is}$, where J_{is} is the total number of operations of part (i,s). Operation (i,s,j) has to be performed by a machine belonging to an eligible type $h_s \in H_{isj} \subset H_s$ with the given processing time t_{ijh_s}. Factory-level decision variables and cell-level decision variables are related through the following relationship: the first operation beginning time of part (i,s), $b_{i,1}^s$, is the beginning time of part (i,s), b_i^s. Similarly, the last operation completion time of part (i,s), $c_{i,J_{i,s}}^s$, is the completion time of part (i,s), c_i^s.

Five types of holons at the cell level are identified, including *Part*, *Machine-type*, *Machine*, *Cell Coordinator*, and *Cell* holons.

A *Part* holon corresponds to a particular production stage of a product and resides in the associated cell. Its informational component contains the detailed process plan of the part, the desired part completion time d_i^s (derived from the product due date and processing times of subsequent parts), earliest beginning time $a_{i,s}$ (derived from the arrival time of raw materials a_i and processing times of preceding parts), desired part beginning time $\bar{b}_i^s$, etc. The process plan is translated into the following constraints.

Operation Precedence Constraints. An operation cannot be started until its preceding operation is finished:

$$c_{ij}^s + S_{i,j,j+1}^s + 1 \le b_{i,j+1}^s, \forall i \in I_s \text{ and } 0 < j \le J_{is}, \tag{3}$$

where $b_{i,j+1}^s$ and c_{ij}^s represent the beginning time of $(i,s,j+1)$ and the completion time of (i,s,j), respectively, and $S_{i,j,j+1}^s$ is the required timeout in-between. For the last operation of part (i,s), this constraint reduces to the part precedence constraint of (1). An additional constraint can be added where the left-hand-side of (3) is replaced by to $a_{i,s}$, the earliest beginning time of the part.

Processing Time Requirements. Each operation must be assigned the required amount of time for processing on the specified machine type.

A *Machine-type* holon corresponds to a group of machines within the cell having identical functionality and capability from the scheduling viewpoint. Its informational component contains machines' functionality, capability, the availability of individual machines, and the aggregate availability M_{kh_s}. It also contains mechanisms for cooperating with *Part* and *Cell Coordinator* holons to schedule various operations assigned to the machine type. Its physical component contains those machines.

Each machine is represented by a *Machine* holon, which is nested in the *Machine-type* holon.

Individual part holons compete for machines to process their operations, and the competition is subject to the following machine capacity constraints.

Machine Capacity Constraints. The total number of operations active on Machine-type h_s at time k should be less than or equal to the number of type h_s machines available at time k, i.e.,

$$\sum_{i \in I_s, j} \delta_{ijkh_s}, \leq M_{kh_s}, 0 \leq k \leq K-1 \text{ and } \forall h_s \in H_{isj}, \tag{4}$$

where the operation-level variable δ_{ijkh_s} is one if operation (i,s,j) is active on machine type h_s at time k, and zero otherwise.

Similar to the *Factory Coordinator* holon presented in the previous subsection, a *Cell Coordinator* holon is created to gather the status of *Machine-type* and *Part* holons in the cell, and to coordinate their scheduling activities to achieve the cell's objective. The cell's objective function will be derived later in Section 4 based on the factory objective (2) and the coordination information established by the Factory Coordinator. It is sufficient to say at this point of time that the objective function is additive in terms of decision variables associated with various parts. This objective is to be minimized subject to the operation precedence (3), processing time, and machine capacity (4) constraints. The decision variables are operation beginning times $\{b_{ij}^s\}$ and the machine type to be used for each operation.

A *Cell* holon corresponds to a cell, and itself is a holarchy consisting of *Machine-type*, *Part*, and the *Cell Coordinator* holons. It defines how these holons cooperate with each other to generates local schedules, and how the holarchy cooperates with *Product* and the *Factory Coordinator* holons to coordinate activities across cells.

At the factory level, part precedence constraints couple decision variables associated with various cells. At the cell level, machine capacity constraints couple decision variables associated with different parts. Since the objective functions and the coupling constraints are all additive in terms of decision variables, the problem formulation is 'separable.' Lagrangian relaxation can therefore be effectively used to decompose the problem into cell-level subproblems and further down into part level subproblems, and to delineate the coordination information and establish the cooperation mechanisms among various holons for overall system performance as will be presented in Section 4.

Corresponding to the above two-level holonic model, a centralized or single-level model can be obtained by modeling the factory as a whole, containing all the information delineated above at a single level. This centralized model will be solved by using the method of Wang et al. [42], and results used as a benchmark to compare the performance of the holonic results.

4. Solution methodology and cooperation mechanisms

Lagrangian relaxation has been successfully used for the near-optimal scheduling of several types of manufacturing systems, including identical machines and job shops as presented in Refs. [26,42]. The key idea of the method, in a nutshell, is decomposition and coordination, where decomposition is based on the 'separability' of models, and coordination based on the 'pricing' concept of the *market economy*. The method begins with *creative modeling* that abstracts the essence of a real setting into a formulation that is separable; that is, its objective and the 'coupling constraints' are additive. The 'hard' coupling constraints are then 'softened' or 'relaxed' by using Lagrangian multipliers or 'shadow prices' in the economics literature. Since the original problem is separable, the 'relaxed problem' can be decomposed into many smaller subproblems. If decomposition is performed at the right level of granularity, these subproblems will be much easier to solve as compared to the original problem, and solutions can be efficiently obtained at the ‘low level’ for a given set of multipliers or prices. Multipliers are then iteratively adjusted at the ‘high level’ based on the degrees of constraint violation following the market economy concept: increase prices for over-subscribed constraints (e.g., over utilization of resources) and reduce prices otherwise. In the optimization terminology, the *concave* ‘dual function’ is iteratively *maximized* here. The method thus consists of the following three major steps: (1) Relax system-wide coupling constraints by using a set of Lagrange multipliers, and decompose the relaxed problem into simpler subproblems. (2) Solve the subproblems for the given set of multipliers. (3) Update the multipliers based on the degrees of constraint violation by using continuous variable optimization methods such as the subgradient method, the conjugate subgradient method, etc., and go back to (1) until certain stopping criteria are satisfied. The method is particularly powerful when the original problem is NP-hard whereas the decomposed subproblems are not.

Corresponding to the holonic model presented in Section 3, the above three steps are recursively used at both the factory level and the cell level for delineating the coordination information and establishing the cooperation mechanisms. At the factory level, part precedence constraints are relaxed and the problem is decomposed into individual cell subproblems. Within a cell, machine capacity constraints are relaxed and the cell subproblem is further decomposed into individual part subproblems. This two-level decomposition and coordination method naturally maps Lagrangian relaxation onto the recursive holonic structure.

4.1. Factory step 1: relax the factory-wide constraints

As mentioned, the objective of scheduling is to minimize the weighted product penalties as defined in Eq. (2). To avoid the oscillation of solutions with slight change of multiplier values when linear part precedence constraints (1) are relaxed resulting in pseudo-linear subproblem objectives as explained in Ref. [11], the objective (2) is modified to

$$\mathscr{J} \equiv \sum_{i,s} \left\{ \omega_{is} T_{is}^2 + \beta_{is} E_{is}^2 \right\}, \tag{5}$$

where the tardiness of part (i,s), T_{is}, is defined as the amount the part completion time c_i^s passes its desired completion time d_i^s, i.e., $T_{is} \equiv \max\{0, c_i^s - d_i^s\}$. The earliness of (i,s), E_{is}, is similarly defined as the amount the part beginning time b_i^s leads the desired beginning time $\bar{b}_i^s$, i.e., $E_{is} \equiv \max\{0, \bar{b}_i^s - b_i^s\}$. The desired completion time d_i^s and desired beginning time $\bar{b}_i^s$ can be derived from the product due date d_i and desired beginning time $\bar{b}_i$ using heuristics based on operation processing times. Parameters ω_{is} and β_{is} are weights associated with these penalties, with product weights ω_i and β_i appropriately imbedded in. Generally part tardiness weights are much smaller than product tardiness weights, and part earliness weights are much smaller than product earliness weights. These penalties therefore define a time window in which a part can be scheduled without penalty. The objective function (5) is to be minimized by selecting part beginning and completion times $\{b_i^s, c_i^s\}$ subject to part precedence (1) and the raw material arrival constraints.

The part precedence constraints (1) are first relaxed by using Lagrange multipliers $\{\eta\}$ (precedence prices), and the factory-level Lagrangian is formed as:

$$L^{\mathrm{F}} \equiv \sum_{i,s} \left\{ \omega_{is} T_{is}^2 + \beta_{is} E_{is}^2 \right\} + \sum_{i,s} \left\{ \eta_i^{ss'} \left(c_i^s + S_i^{ss'} + 1 - b_i^{s'} \right) \right\}. \tag{6}$$

With prices given, the ‘relaxed problem’ is to choose decision variables $\{b_i^s,\ c_i^s\}$ to minimize the Lagrangian L^{F} subject to part precedence (1) and the raw material arrival time constraints. Part beginning and completion times are not totally independent decision variables, rather, they are determined by solving cell subproblems formed by grouping terms in Eq. (6) related to individual cells:

$$\min_{b_i^s, c_i^s} L_s, \text{ with } L_s \equiv \sum_{i \in I_s} \left\{ \omega_{is} T_{is}^2 + \beta_{is} E_{is}^2 + \eta_i^{ss'} \left(c_i^s + S_i^{ss'} + 1 \right) - \eta_i^{\hat{s}s} b_i^s \right\}, \tag{7}$$

where $(i,\hat{s})$ is the part immediately preceding (i,s), i.e., $(i,s) \in I_i^{\hat{s}}$.

In the following subsections, the resolution of cell subproblems will be presented first, and the resolution of the factory problem will then be wrapped up. Information necessary for coordination will be delineated and cooperation mechanisms among holons established based on the solution process. The generation of feasible schedules will also be discussed.

4.2. Factory step 2: solve the cell subproblems

For cell s, the objective (7) is to be minimized subject to constraints within the cell. Similar to the factory level, the cell-wide machine capacity constraints are first relaxed, and the decomposed subproblems are then solved separately.

4.2.1. Cell step 1: relax cell-wide capacity constraints

The machine capacity constraints (4) within cell s are further relaxed by using Lagrange multipliers $\{\pi\}$ (capacity prices), and the cell-level Lagrangian is formed as:

$$L_s^{\mathrm{C}} \equiv \sum_{i \in I_s} \Big\{ \omega_{is} T_{is}^2 + \beta_{is} E_{is}^2 + \eta_i^{ss'} \times \big(c_{i,J_{is}}^s + S_i^{ss'} + 1 \big) - \eta_i^{\hat{s}s} b_{i1}^s \Big\} + \sum_{k,h_s \in H_s} \left\{ \pi_{kh_s} \left(\sum_{i \in I_s, j} \delta_{ijkh_s} - M_{kh_s} \right) \right\}. \quad (8)$$

In the above, the beginning and completion times of part (i,s), i.e., b_i^s and c_i^s, are now re-written as the beginning time b_{i1}^s of the part's first operation and the completion time $c_{i,J_{i,s}}^s$ of the part's last operation, respectively.

Collecting all the terms related to part (i,s) lead to the following part's subproblem:

$$\min_{b_{ij}^s, h_{isj}} L_{is}, \text{ with } L_{is} \equiv \omega_{is} T_{is}^2 + \beta_{is} E_{is}^2 + \eta_i^{ss'} \times \big(c_{i,J_{is}}^s + S_i^{ss'} + 1 \big) - \eta_i^{\hat{s}s} b_{i1}^s + \sum_{j=1}^{J_{is}} \sum_{k=b_{ij}^s}^{c_{ij}^s} \pi_{kh_{iaj}}, \quad (9)$$

where the fact that δ_{ijkh_s} is one if operation (i,s,j) is active on machine type h_s at time k (i.e., $b_{ij}^s \le k \le c_{ij}^s$), and zero otherwise is used. This subproblem is subject to operation precedence (3) and processing time constraints, and decision variables are the operation beginning time $\{b_{ij}^s\}$ and the machine type to be used for each operation of part (i,s).

4.2.2. Cell step 2: solve part-level subproblems

Solving a part subproblem (9) involves searching among all eligible beginning times and eligible machine types for the smallest cost, striving for a balance among tardiness and earliness penalties, proper positioning within the precedence structure, and machine utilization costs. This can be done by using dynamic programming (DP) as presented in Chen et al. [9] and Wang et al. [42]. Since the ‘backward’ dynamic programming (BDP) can be extended to deal with uncertainties as presented in Luh et al. [24], it is used in this study. With stages corresponding to operations and states corresponding to operation beginning times, the BDP starts with the last stage, and computes the tardiness penalties and machine utilization costs. As the algorithm moves backwards, cumulative costs of individual states belonging to a particular stage are computed based on the stage-wise costs and the minimum of the cumulative costs for the succeeding stage, subject to allowable state transitions as dictated by operation precedence constraints. This minimization can be very efficiently implemented by pair-wise comparisons, starting from the last state (largest possible operation beginning time) of the succeeding stage [42]. The optimal subproblem cost is then obtained as the minimum of the cumulative costs at the first stage, and the optimal beginning times for individual operations can be obtained by tracing the stages forwards. The computational complexity is $O(\sum_{j=1}^{J_{is}} |H_{isj}| K)$ as shown in Chen et al. [9], where $|H_{isj}|$ is the cardinality of H_{isj}.

The above solution process requires the cooperation among *Part* and related *Machine-type* holons. The *Part* holon selects machine types and coordinates operation beginning times within the BDP process, whereas *Machine-type* holons compute the machine utilization costs ($\sum_{k=b_{ij}^s}^{c_{ij}^s} \pi_{kh_{isj}}$) for various operations to be used by BDP.

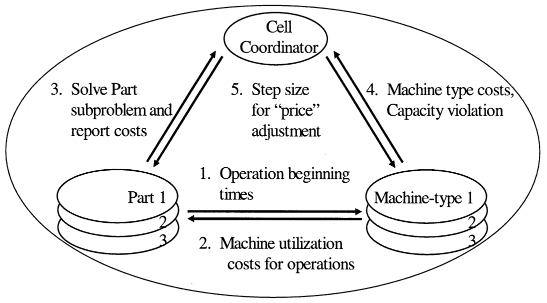

Cell Coordinator
3. Solve Part subproblem and report costs
5. Step size for "price" adjustment
4. Machine type costs, Capacity violation
Part 1
2
3
1. Operation beginning times
Machine-type 1
2
3
2. Machine utilization costs for operations

Fig. 3. Cell holon.

4.2.3. Cell step 3: update capacity prices

Let L_{is}^* denote the minimized L_{is}, then the dual problem is to maximize the cell dual function D_s^{C} with respect to capacity prices:

$$\max_{\pi \geq 0} D_s^{\mathrm{C}}, \text{ with } D_s^{\mathrm{C}} \equiv \sum_{i \in I_s} L_{is} * - \sum_{k, h_s \in H_s} \pi_{kh_s}, M_{kh_s}, \tag{10}$$

where $(\sum_k \pi_{kh_s} M_{kh_s})$ is defined as the 'machine type cost' for *Machine-type* $h_s \in H_s$.

Once part subproblems have been solved, the degrees of capacity constraint violation, i.e., the capacity subgradient, is evaluated by individual *Machine-type* holons. The component of the capacity subgradient for machine type h_s at time k is given by:

$$\mathscr{g}_{\pi_{kh_s}} \equiv \sum_{i,j} \delta_{ijkh_s} - M_{kh_s}. \tag{11}$$

Based on the subgradient information and the value of cell dual cost D_s^{C}, the coordination information, i.e., the price adjusting direction $d_{\pi_{kh_s}}$ and the step size $\alpha_{\pi_{kh_s}}$ can be obtained by using several methods. Since the dual function has many facets and tends to be smooth when the problem size is large, the conjugate subgradient method (CSG) is used in this study (same as the conjugate gradient method in Bertsekas [5] (pp. 122–132), with gradients replaced by subgradients). In each CSG iteration, the line search along direction $d_{\pi_{kh_s}}$ is needed, and several function evaluations (D_s^{C} in 10) are involved. This algorithm is embedded in the *Cell Coordinator* holon for generating the coordination information.

The capacity prices are updated by individual *Machine-type* holons according to the coordination information, i.e.,

$$(\pi_{kh_s})^{n+1} = (\pi_{kh_s})^n + (\alpha_{\pi_{kh_s}})^n (d_{\pi_{kh_s}})^n, \tag{12}$$

where superscripts n and $n+1$ refer to the iteration number.

The cooperation mechanisms are embedded in the cell holarchy as shown in Fig. 3, and the primary message passing among holons and its execution sequence are summarized in Table 1. From the factory viewpoint, *Cell* holons modify their local objectives (7) according to factory-level precedence prices, and coordinate schedules within individual cells.

4.3. Factory step 3: update precedence prices

Let L_s^* denote the minimized L_s, then the factory-level dual problem is to maximize the factory-level Lagrangian L^{F} in (6) with respect to precedence prices:

$$\max_{\eta \geq 0} D^{\mathrm{F}}, \text{ with } D^{\mathrm{F}} \equiv \sum_s L_s^*. \tag{13}$$

Based on cell-level solutions, the degrees of part precedence constraint violation, i.e., the precedence

Table 1
Message passing among holons and its execution sequence

Execution sequence	Functionality	Source	Message	Destination
1	Setup part subproblem	Part	$\{b_{ij}^s, t_{ijh_{s_s}}\}$	Machine-type
2	Compute utilization cost	Machine-type	$\sum_{j=1}^{J_{is}} \sum_{k=b_{ij}^s}^{c_{is}^s} \pi_{kh_{isj}}$	Part
3	Solve subproblem	Part	$L_{is} *$	Cell Coordinator
4	Monitor machine status	Machine-type	$\sum_k \pi_{kh_s} M_{kh_s}, g\pi_{khs},$	Cell Coordinator
5	Construct coordination information	Cell	$d_{\pi_{kh_s}}, \alpha_{\pi_{kh_s}}$	Machine-type
		Cell Coordinator	Iteration start signal	Part

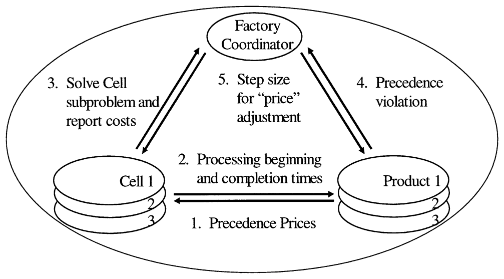

Factory Coordinator
3. Solve Cell subproblem and report costs
5. Step size for "price" adjustment
4. Precedence violation
2. Processing beginning and completion times
Cell 1
2
3
Product 1
2
3
1. Precedence Prices

Fig. 4. Factory holon.

subgradient, is evaluated by individual *Product* holons. The component of the precedence subgradient as related to two consecutive parts (i,s) and (i,s') is given by:

$$\mathscr{g}_\eta \equiv c_i^s + S_i^{ss'} + 1 - b_i^{s'}. \qquad (14)$$

The prices are updated by individual *Product* holons similar to (12), i.e.,

$$(\eta_i^{ss'})^{n+1} = (\eta_i^{ss'})^n + (\alpha_\eta)^n (d_\eta)^n. \qquad (15)$$

Many times the minimized cell costs L_s in (13) may not be available in view of the combinatorial nature of the subproblems or limited by available computational times. Since the maximum dual function cost D_s^C is a lower bound to L_s^*, D_s^C or its approximation can be used to replace L_s^* in (13).

Similar to *Cell Coordinator* holons presented in Section 4.2, the *Factory Coordinator* holon generates factory-level coordination information, i.e., the precedence price adjusting direction d_η and the corresponding step size α_η, to guide the updating of precedence prices. The information needed includes cell subproblem costs (cell-level dual costs can be used instead) and part precedence subgradients. The cooperation mechanisms are embedded in the factory holarchy as shown in Fig. 4, and the primary message passing among holons and the execution sequence are summarized in Table 2.

4.4. Generating feasible schedules

As can be seen from the above, the two level cooperation mechanisms are an iterative optimization process. After precedence prices are updated at the factory level, local objectives of relevant cells are modified. Cell-level solutions are then iteratively updated for the modified objectives. In practice, this optimization is performed at given 'snapshots,' either periodically or after the occurrence of a major event (e.g., the arrival of a major order or the breakdown of a key machine). As will be illustrated in Example 2 of Section 5, the prices obtained from one optimization process can be used to initialize the next process to speed up convergence. For each process, optimization is stopped after a fixed amount of computation time has elapsed, after a fixed number of iterations has been executed, or after the optimal solution has been detected. The operation beginning times, machine types selected, and the precedence and machine prices are embedded in *Part*, *Product*, and *Machine-type* holons. Since part precedence constraints across cells and machine capacity constraints within each cell have been relaxed, solutions of subproblems, when put together, generally do not constitute a feasible schedule. The solutions, however, provide valuable information for operation dispatching as presented in Ramaswamy and Joshi [33]. In doing this, *Product* holons send part information to *Cell* holons, *Part* holons send operation information to *Machine-type* holons, and *Machine-type* holons dispatch operations based on the machine prices and parts' status. In this way, opti-

Table 2
Message passing among holons and its execution sequence

Execution sequence	Functionality	Source	Message	Destination
1	Modify cell objective	Product	η	Cell
2	Coordinate cell subproblem	Cell	$\{b_i^s, c_i^s\}$	Product
3	Solve subproblems	Product	$\mathscr{g}_\eta$	Factory Coordinator
4	Report cell subproblem cost	Cell	D_s^*	Factory Coordinator
5	Construct coordination information	Factory	d_η, α_η	Product
		Coordinator	Iteration start signal	Cell

mization is performed in the background to support the foreground operation dispatching.

Currently, a heuristic similar to what was presented in Luh and Hoitomt [26] is used at individual cells to modify subproblem solutions for operation dispatching. Each cell has a list U_s of ‘assignable’ operations created at time 0 based on subproblem solutions, and this list is updated at subsequent time slots according to the realizations at individual cells. Operations in each U_s are scheduled in the ascending order of their beginning times as the required machines become available. If capacity constraint is violated at a particular time, a greedy heuristic is used to determine which operations should be scheduled and which ones should be delayed. Assignable lists are then updated based on the completion of operations within the cell as well as from other cells. This procedure is then repeated at the next time slot until all operations are dispatched.

An important by-product of the method is that the dual value is a lower bound to the optimal cost. The quality of a schedule obtained can thus be quantitatively evaluated by its relative duality gap which is the relative difference between the minimum feasible schedule cost $\mathcal{J}$ of Eq. (5) and the maximum dual value obtained D^{F*} of Eq. (13), i.e., Duality Gap $\equiv (\mathcal{J}* - D^{F*})/(D^{F*})100\%$.

Lagrangian relaxation can also be applied to the centralized formulation as presented in Refs. [26,42]. Depending on what constraints are relaxed, various subproblems can be formed with corresponding coordination methods. In Appendix B, the computational complexity under a simplifying assumption is analyzed to roughly compare the holonic method with the single-level method.

5. Implementation and testing results

A scheduling system based on the above architecture has been implemented to examine the performance of the holonic method developed. Lacking a distributed computation platform and a physical factory environment, issues related to communication protocols, asynchronous implementation, and the integration of physical and informational components of holons are not addressed. The implementation is on a single Sun ULTRA-1 workstation using the object-oriented language C + +, with holons’ informational components implemented as software objects consisting of encapsulated information and functions. These functions support the processing of information, and the cooperation with other holons.

Three examples are presented here to demonstrate the performance of the holonic method developed. These examples draw data from Toshiba’s HamaKawasaki Factory, which produces gas insulated switchgears for electric utilities. There are three cells of 22 machine types with a total of 87 machines. Machines may not always be available, and products may have different earliest beginning times, due dates, and weights. In view of the long time horizon needed to schedule all products (e.g., 590 h) and the small resolution required to handle certain short operations (e.g., 1 h), the ‘time step reduction technique’ presented in Luh et al. [25] is used. In this technique, multiple ‘resolution increments’ are aggregated into an ‘enumeration step’ and represented by a single machine price (e.g., 10 one-hour resolution increments are aggregated into a 10-h enumeration step for the following examples). Multiple ‘short’ operations are thus allowed to ‘share’ a machine within an enumeration step, and a part with several ‘short’ operations is allowed to flow through the machines. This technique also decreases the number of prices needed while maintaining a certain degree of modeling accuracy. As presented in Appendix B, each factory-level CSG iteration involves N_1 (≈ 3.5) factory-level dual function evaluations, and each factory-level dual function evaluation has N_3 cell-level CSG iterations. In the following examples, $N_3 = 5$ is used for individual cells, and prices are initialized at zero unless specified otherwise.

Example 1: This example is to demonstrate that the holonic method can obtain high quality schedules

Table 3
Description of example 1

Case	No. of products	No. of operations	Time horizon (h)
1	100	633	590
2	150	922	770

Table 4
Numerical results for example 1

Case	Method	Feasible cost $\mathcal{J}$ defined in Eq. (5)	Feasible cost $\mathcal{J}$ defined in Eq. (2)	Duality gap	Time (min:s)
1	Holonic	84 167	74 955	15.4%	6:20
	Single-level	/	74 820	12.9%	5:00
2	Holonic	92 033	82 798	17.0%	10:25
	Single-level	/	82 297	16.9%	8:56

without accessing local information nor intruding on the decision authority of individual cells. Two test cases summarized in Table 3 are used.

For comparison purpose, the 'single-level method' is also applied for these cases with Eq. (2) as the objective function. Optimization iterations for the holonic method are stopped after $(N_4 =)$ 25 factory-level CSG iterations, and optimization iterations for the single-level method are stopped after $(N_2 =)$ 450 CSG iterations. Based on the analysis of Appendix B (Eq. (16)), $\gamma \approx 1$, implying similar computational requirements for these two methods.

Numerical results are summarized in Table 4. It can be seen that with a similar computation effort, the two methods generate results of similar quality. In addition, when the number of products and consequently the time horizon increase, computation requirements needed to carry out a fixed number of CSG iterations increase. Computation times for both methods can be reduced by solving individual subproblems in parallel.

Example 2: This example is to show that the computation time can be reduced by initializing prices at values obtained from the previous scheduling process [26]. Based on the schedule obtained in Case 1 of Example 1, assume that 17 products have been completed in the first three days (6 ten-hour shifts), and 22 new products have arrived. The schedule is reconfigured at the beginning of the fourth day by initializing the prices at values obtained from Case 1 of Example 1, with 105 products for a total of 660 operations to be scheduled. For comparison purpose, optimization is also performed when prices are initialized at zero.

Numerical results are summarized in Table 5. To obtain a schedule with a comparable quality, the CPU time can be decreased to one-third by using the prices obtained from the previous scheduling process (as opposed to initializing the prices at zero). From a different perspective, a better schedule can be obtained within a fixed amount of computation time.

Example 3: Most manufacturing systems are heterogeneous. This example is to examine the behavior of the holonic method in a heterogeneous environment, which is simulated by assuming that Cell 3 has a 'slow' local scheduler, i.e., it can only perform a limited number of CSG iterations (N_3) in-between two factory-level price updates. The same data set as in Case 1 of Example 1 is used, and optimization

Table 5
Numerical results for example 2

Price initialization	Feasible cost $\mathcal{J}$ defined in Eq. (5)	Feasible cost $\mathcal{J}$ defined in Eq. (2)	Duality gap	CPU time (min:s)
at 0	84 942	75 615	18.9%	5:00
from previous process	84 942	75 615	18.7%	1:35
from previous process	84 389	75 177	14.6%	5:00

Table 6
Numerical results for example 3

No. of CSG iter. at cell 3	Feasible cost $\mathcal{J}$ defined in Eq. (5)	Feasible cost $\mathcal{J}$ defined in Eq. (2)	Duality gap
1	85 018	75 678	20.6%
2	84 788	75 540	19.6%
5	84 167	74 955	15.4%
10	84 286	75 021	15.7%

iterations are stopped when the number of the factory-level CSG iterations (N_4) reaches 25.

Numerical results are summarized in Table 6, where results for the normal case $(N_3 = 5)$ and a 'fast' scheduler case $(N_3 = 10)$ are also included. It can be seen that the 'slowness' of the local scheduler $(N_3 = 1$ or $2)$ may affect the quality of schedules, whereas a single fast local scheduler $(N_3 = 10)$ may not significantly improve the overall scheduling quality. This example thus implies that as long as the necessary coordination information as delineated in Section 4 is provided, the holonic method has the potential to be extended to coordinate activities across heterogeneous subsystems having different local scheduling mechanisms.

6. Conclusion

In this paper, holonic scheduling is developed for a factory consisting of multiple cells. Through a novel modeling of the interactions among cells and a decomposition-coordination solution method based on Lagrangian relaxation, coordination across cells is performed without accessing individual cells' local information nor intruding on their decision authority. Numerical testing shows that the method can generate high quality schedules in a timely fashion, and has comparable computational requirements as compared to the single-level Lagrangian relaxation method. This study therefore demonstrated that the Lagrangian relaxation technique can be naturally mapped onto the holonic concept to meet the holonic requirements of limited information accessibility and individual decision-making autonomy, and provides a theoretical foundation for guiding the cooperation among holons, leading to globally near-optimal performance.

Acknowledgements

This work was supported in part by United Technologies Research Center, the National Science Foundation under Grants DDM-9500037, Toshiba, and the Advanced Technology Center for Precision Manufacturing, the University of Connecticut. The authors would like to thank Dr. Shinsuke Tamura, Dr. John M. Oblak, Mr. Tetsuo Hasegawa, and Mr. Yuanhui Zhang for their valuable suggestions and effort. The authors are grateful to Prof. A.D. Baker of University of Cincinnati and referees for their valuable comments.

Appendix A. A list of symbols

$\lvert x \rvert$	Cardinality of x
a_i	Arrival time of raw materials for product i
$\bar{b}_i^s$	Desired release time of part (i,s)
b_i	Beginning time of product i
b_i^s	Beginning time of part (i,s)
b_{ij}^s	Beginning time of operation (i,s,j)
c_i	Completion time of product i
c_i^s	Completion time of part (i,s)
c_{ij}^s	Completion time of operation (i,s,j)
D_s^{C}	Cell-level dual
D^{F}	Factory-level dual
$D^{\mathrm{F}*}$	Maximum factory-level dual value obtained till current factory-level iteration
d_i	Due date of product i
d_η	Direction for updating multiplier η
d_π	Direction for updating multiplier π

E_i	Earliness of product i
E_{is}	Earliness of part (i,s)
$\mathscr{G}_\eta$	Subgradient for η
$\mathscr{G}_\pi$	Subgradient for π
H_s	Set of machine types in cell s
H_{isj}	Eligible machine type set for processing operation (i,s,j)
h_s	Machine type index
I	Set of products to be scheduled
I_s	Set of products to be processed in cell s
I_{is}	Set of possible immediate subsequent parts of (i,s)
i	Product index, $i \in I$
$\mathscr{J}$	Feasible schedule cost
$\mathscr{J}^*$	Minimum feasible cost obtained up to and include the current iteration
(i,s,j)	j-th operation of part (i,s), $1 \le j \le J_{is}$
J_{is}	Total number of operations of part (i,s)
j	Operation index
K	Time horizon
k	Time index, $0 \le k \le K-1$
L_s	Objective function for cell subproblem s
L_{is}	Objective function for Part subproblem (i,s)
L^{F}	Factory-level Lagrangian
L_s^{C}	Cell-level Lagrangian for cell s
M_{kh_s}	Number of machines of type h_s available at time k
N_1	Number of function evaluations in one CSG iteration
N_2	Number of CSG iterations for the single-level method
N_3	Number of cell-level CSG iterations
N_4	Number of factory-level CSG iterations
S	Set of cells in the factory
$S_i^{ss'}$	Timeout for product i between cells s and s'
$S_{i,j,j+1}^s$	Timeout between operations (i,s,j) and $(i,s,j+1)$
s	Cell index, $s \in S$
T_i	Tardiness of product i
T_{is}	Tardiness of part (i,s)
t_{ijh_s}	Processing time of operation (i,s,j) on machine type h_s
α_η	Step size for updating multiplier η
α_π	Step size for updating multiplier π
β_{is}	Weight of earliness penalty for part (i,s)
β_i	Weight of earliness penalty for product i
β_{is}	Weight of earliness penalty for part (i,s)
δ_{ijkh_s}	Operation-level variable equal to one if (i,s,j) is active on machine type h_s at time k, and zero otherwise
$\eta_i^{ss'}$	Lagrangian multiplier relaxing the precedence constraint between parts (i,s) and (i,s')
π_{kh_s}	Lagrange multiplier relaxing the machine capacity constraint for machine type h_s at time k
ω_i	Weight of tardiness penalty for product i
ω_{is}	Weight of tardiness penalty for part (i,s)

Appendix B. Complexity analysis

In the following, computational complexity under a simplifying assumption is analyzed to roughly compare the holonic implementation with the single-level method. Suppose that there are $|H|$ machine types in the factory, and $|I|$ products are to be scheduled. Each product has $|J|$ operations, and each operation having only one eligible machine type. For simplicity of discussion, one function evaluation here refers to solving all the related subproblems for a given set of Lagrange multipliers (prices).

B.1. Complexity of the single-level method

For the single-level method, the complexity for one CSG iteration is first discussed, and the complexity for the overall solution process is then presented.

Complexity of One CSG Iteration. The complexity for solving a product subproblem using DP is $O(K|J|)$. Since one function evaluation involves solving $|I|$ product subproblems, the complexity is $O(K|J||I|)$. The complexity for adjusting prices in (12) is $O(K\ |H|)$. Assuming the average number of function evaluations required for each CSG iteration is N_1 (N_1 approximately 3.5 based on our testing experience), the complexity for one CSG iteration is thus $O(K|J||I|N_1 + K|H|N_1)$.

Complexity of the Solution Process. For a quadratic function, the number of CSG iterations needed to arrive at the optimal is the dimension of decision variables. The function considered here generally has a large dimension and is not quadratic. For example, for Case 2 of Example 1 in Section 5, the dimension is the total number of machine capacity prices $K|H|$, i.e., 1694. It is unrealistic in practice to wait for so many iterations to generate a schedule. A fixed number of CSG iterations N_2 is therefore used, and the complexity is thus given by $O(K|J||I|N_1 N_2 + K|H|N_1 N_2)$.

B.2. Complexity of the implementation in this study

To analyze the complexity of holonic implementation presented in this study, assume for simplicity that there are $|S|$ cells, each with the same number of machine types $(|H|)/(|S|)$. Individual products have to go through every cell once for $(|J|)/(|S|)$ operations. In the following, the complexity of one factory-level CSG iteration is first discussed, and the complexity for the overall solution process is then presented.

Complexity of One Factory-Level CSG Iteration. A CSG iteration, as mentioned earlier, has N_1 function evaluations, and one cell-level function evaluation involves solving $|I|$ part subproblems. The complexity of one cell-level CSG iteration is thus $O(K(|J|)/(|S|)|I|N_1 + K(|H|)/(|S|)N_1)$ following the above derivation. Since the time between two factory price updates is limited, a fixed number N_3 ($N_3 = 5$ from our testing experience) of cell-level CSG iterations is assumed for each factory-level function evaluation. Considering the complexity $O(|I||S|)$ for adjusting precedence prices (15), the complexity of one factory-level CSG iteration is $O(K|J||I|N_1^2\ N_3 + K|H|N_{12}\ N_3 + |I||S|N_1)$.

Complexity of the Solution Process. Similar to the single-level case, the factory-level problem has a large dimension $|I|(|S| - 1)$, and a fixed number of factory-level CSG iterations N_4 is assumed. The complexity is thus given by $O(K|J||I|N_1^2 N_3 N_4 + K|H||I|N_1^2 N_3 N_4 + |I||S|N_1 N_4)$.

For our testing, $|I||J|$ is generally much larger than $|H|$, and the dominating terms in the complexity analysis of both methods are the first ones. With given parameters (N_1, N_2, N_3, and N_4), the complexity of these two methods increases linearly with time horizon K and the number of products $|I|$. To roughly compare these two methods, the ratio of the single-level complexity to the holonic complexity implemented is given by:

$$\gamma \equiv \frac{K|J||I|N_1 N_2}{K|J||I|N_1^2 N_3 N_4} = \frac{N_2}{N_1 N_3 N_4}. \tag{16}$$

It can be seen that γ is a constant depending on the parameters selected but not explicitly on the problem size.

B.3. Complexity of a distributed holonic implementation

Finally, the complexity of a distributed holonic implementation ignoring the communication overhead is examined. Assume that each cell has one computer for its subproblem solving, and each machine type has one computer for its scheduling. Since the complexity for each cell under our simplifying condition is the same, the complexity of one factory-level CSG iteration is $O((K|J||I|N_1^2 N_3)/(|S|) + KN_1^2 N_3 + |I|N_1)$ according to the previous analysis, and the complexity of the solution process is $O((K|J||I|N_1^2 N_3 N_4)/(|S|) + K|I|N_1^2 N_3 N_4 + |I|N_1 N_4)$.

It can be seen that computational speedup can be obtained, and the complexity is not related to the number of machine types. The communication overhead, however, will increase with the number of computers, and will finally limit the computational speedup.

References

[1] P. Anstiss, The implementation and control of advanced manufacturing systems, Control and Programming in Advanced Manufacturing, IFS Publications, Springer-Verlag, Berlin, UK, 1988.

[2] A.D. Baker, A survey of factory control algorithms which can be implemented in a multi-agent heterachy, submitted to the Journal of Manufacturing Systems, 1997.

[3] A.D. Baker, H.V.D. Parunak, E. Erol, Manufacturing over the Internet and into Your Living Room: Perspectives from the AARIA Project, submitted to the IEEE Internet Computing, 1997.

[4] C.A. Bartlett, S. Ghoshal, Changing the Role of Top Management: Beyond Structure to Process, National Productivity Review, January–February, 1995, 86–96.

[5] D.P. Bertsekas, Nonlinear Programming, Athena Scientific, 1995.

[6] J.H. Blackstone, D.T. Phillips, G.L. Hogg, A state-of-the-art survey of dispatching rules for manufacturing job shop operations, International Journal of Production Research 20 (1982) 27–45.

[7] J. Blazewicz, G. Finke, R. Haupt, G. Schmidt, New trends in machine scheduling, European Journal of Operational Research 36 (1988) 303–316.

[8] L. Bongaerts, P. Valckenaers, H. Van Brussel, P. Peeters, 1997, Scheduling execution in holonic manufacturing systems, in: Proc. 29th CIRP International Seminar on Manufacturing Systems, 209–214.

[9] H. Chen, C. Chu, J.M. Proth, A more effective Lagrangian relaxation approach, in: Proc. IEEE Int. Conf. on Robotics and Automation, 1995, 496–501.

[10] T.J. Crowe, E.J. Stahlman, A proposed structure for distributed shop floor control, Integrated Manufacturing Systems 6 (6) (1995) 31–36.

[11] C.S. Czerwinski, P.B. Luh, Scheduling products with bills of materials using an improved Lagrangian relaxation technique, IEEE Trans. Robotics and Automation 10 (2) (1994) 99–111.

[12] P.F. Drucker, The Coming of the New Organization, Harvard Business Review, January–February, 1988, 45–53.

[13] N.A. Duffie, R.S. Pipier, Non-hierarchical control of a flexible manufacturing cell, Robotics Computer-Integrated Manufacturing 3 (2) (1987) 175–179.

[14] N.A. Duffie, V.V. Prabhu, Real-time distributed scheduling of heterarchical manufacturing systems, Journal of Manufacturing Systems 13 (2) (1994) 175–179.

[15] S.L. Goldman, R.N. Nagel, Management, technology and agility: the emergence of a new ear in manufacturing, International Journal of Technology Management 8 (1/2) (1993) 18–38.

[16] L. Gou, T. Hasegawa, P.B. Luh, S. Tamura, J.M. Oblak, Holonic planning and scheduling for a robotic assembly test bed, in: Proc. 4th International Conference on Computer Integrated Manufacturing and Automation Technology, 1994, 142–149.

[17] J. Harhen, MRP/MRP II, in: A. Rolstadas (Ed.), Computer-Aided Production Management, Springer-Verlag, 1988, 23–35.

[18] T. Hasegawa, L. Gou, S. Tamura, P.B. Luh, J.M. Oblak, Holonic planning and scheduling architecture for manufacturing, in: Proc. 2nd International Working Conference on Cooperating Knowledge Based Systems, Keele, UK, 1994, 125–139.

[19] J. Hatvany, Intelligence and cooperation in heterarchic manufacturing systems, Robotics Computer-Integrated Manufacturing 2 (2) (1985) 101–104.

[20] H. Hayashi, The IMS international collaborative program, in: Proc. 24th International Symposium on Industrial Robots, Japan Industrial Robot Association, 1993.

[21] A. Koestler, The Ghost in the Machine, The Macmillan, 1968.

[22] A. Kusiak, Intelligent Manufacturing Systems, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[23] G.Y. Lin, J.J. Solberg, Integrated shop floor control using autonomous agents, IIE Transaction 24 (3) (1992) 57–71.

[24] P.B. Luh, D. Chen, L.S. Thakur, Modeling uncertainty in job shop scheduling, in: Proc. 1st International Conference on Operations and Quantitative Management Jaipur, India, 1997, 490–497.

[25] P.B. Luh, L. Gou, Y. Zhang, T. Odahara, M. Tsuji, K. Yoneda, T. Hasegawa, Y. Kyoya, Job shop scheduling with group-dependent setups, finite buffers, and long time horizon, Annals of Operations Research 76 (1998) 233–259.

[26] P.B. Luh, D.J. Hoitomt, Scheduling of manufacturing systems using the Lagrangian relaxation technique, IEEE Trans. Automatic Control 38 (6) (1993) 1066–1080.

[27] J. Mathews, Organization foundations of intelligent manufacturing systems—The Holonic Viewpoint, Computer Integrated Manufacturing Systems 8 (4) (1995) 237–243.

[28] F.P. Maturana, D.H. Norrie, Multi-agent mediator architecture for distributed manufacturing, Journal of Intelligent Manufacturing 7 (1996) 257–270.

[29] T. Moriwaki, N. Sugimura, Y.Y. Matawirya, S.H. Wirjomartono, Production Scheduling in Autonomous Distributed Manufacturing Systems, in: Quality Assurance Through Integration of Manufacturing Processes and Systems, PED-Vol. 56, ASME, 1993, 175–186.

[30] R.N. Nagel, R.D. Dove, 21st Century Manufacturing Enterprise Strategy, Iacocca Institute, Lehigh University, Bethlehem, PA, 1991.

[31] NGM Report, 1997, Next Generation Manufacturing Project: A Framework for Action, Agility Forum.

[32] H.V.D. Parunak, A.D. Baker, S.J. Clark, The AARIA agent architecture: an example of requirements-driven agent-based system design, in: Proc. 1st International Conference on Autonomous Agents, 1997, 482–483.

[33] S.E. Ramaswamy, S. Joshi, Distributed control of automated manufacturing systems, in: Proc. 27th CIRP Seminar on Manufacturing Systems, 1995.

[34] D. Seidel, M. Hopf, J.M. Prado, E. Garcia-Herreros, T.D. Strasser, J.H. Christensen, J.M. Oblak, HMS—Strategies, Vol. 0, The Report of HMS Consortium, 1994.

[35] M.J. Shaw, Dynamic scheduling in cellular manufacturing systems: a framework for networked decision making, Journal of Manufacturing Systems 7 (2) (1988) 83–94.

[36] R.G. Smith, The contract net protocol: high-level communication and control in a distributed problem solver, IEEE Trans. Computer 29 (12) (1980) 1104–1113.

[37] H. Suda, Future Factory System Formulated in Japan, Techno Japan, Part 1: Vol. 22, No. 10, 15–25; Part 2: Vol. 23, No. 3, 1989 and 1990, 51–61.

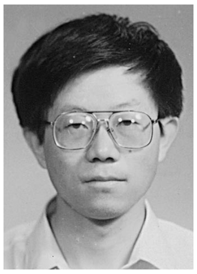
[38] N. Sugimura, Y. Tanimizu, T. Yoshioka, A study on object oriented modeling of holonic manufacturing system, in: Proc. 29th CIRP International Seminar on Manufacturing Systems, 1997 215–220.

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

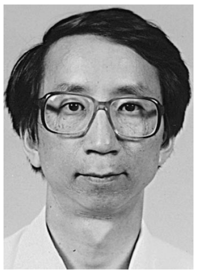
[40] P. Valckenaers, H. Van Brussel, L. Bongaerts, J. Wyns, IMS test case 5, holonic manufacturing systems, to appear in Journal of Integrated Computer-Aided Engineering 4 (3) (1997) 123.

[41] H.J. Warnecke, 1993, The Fractal Company: A Revolution in Corporate Culture, Springer-Verlag.

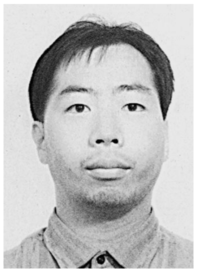
[42] J. Wang, P.B. Luh, X. Zhao, J. Wang, An Optimization-Based Algorithm for Job Shop Scheduling, SADHANA, a Journal of Indian Academy of Sciences, a Special Issue on Competitive Manufacturing Systems, Vol. 22, Part 2, April 1997, pp. 241–256.



Ling Gou received his BS and MS degrees, both in Control Engineering from Chongqing University, China in 1983 and 1986, respectively. He received his MS and PhD degrees, both in Electrical Engineering from the University of Connecticut, Storrs, CT in 1995 and 1997, respectively. From September 1986 to June he was with the Robotics Institute of Beijing University of Science and Technology, working as a Research Engineer. He was a Research Assistant of Manufacturing Systems Laboratory at the University of Connecticut from August 1992 to July 1997. He joined the Operations Research Group of US Airways, as a Senior OR Analyst in August 1997. Since January 1998, he has been with the Operations Research Department of Delta Technology, a wholly owned subsidiary of Delta Air Lines as a Senior OR analyst. His research interests include operations research methods as applied to manufacturing and airline industries; manufacturing planning and scheduling; robotics; holonic systems, agent-based systems and distributed processing; and airline crew scheduling and crew rerouting.



Peter B. Luh received his BS degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, Republic of China in 1973, MS degree in Aeronautics and Astronautics Engineering from MIT, Cambridge, MA in 1977, and PhD degree in Applied Mathematics from Harvard University, Cambridge, MA in 1980. Since 1980 he has been with the University of Connecticut, and currently is the Director of Booth Center for Computer Applications and Research, and a Professor in the Department of Electrical and Systems Engineering. He is interested in planning, scheduling, and coordination of design and manufacturing activities, and has developed a near-optimal and efficient schedule generation and reconfiguration methodology to improve on-time delivery of products and reduce work-in-progress inventory. He is also interested in schedule and transaction optimization for power and has developed near-optimal and efficient unit commitment, hydro-thermal coordination and power transaction methodologies to minimize total fuel costs. He owns one US patent and three Best Paper Awards, and co-authored eight book chapters, 58 journal articles, and 154 conference papers. Dr. Luh is a Fellow of IEEE, and a member of Connecticut Academy of Science and Engineering. He has been the Editorial Board of IEEE Transactions on Robotics and Automation since 1990 (Technical Editor, Associate Editor, and now Editor from 1995 on), is an Editor of IEEE Robotics and Automation Magazine (1996–), and was an Associate Editor for IEEE Transactions on Automatic Control (1989–91).



Yuji Kyoya received the BS and MS degrees in Information and Computer Sciences from Osaka University, Japan in 1992 and 1994, respectively. He is currently the research scientist at Systems and Software Research Lab., Toshiba. He had been working on the development of Job Shop Scheduling system for the first 4 years in Toshiba. Currently he is engaged in the research of Super Design Technology. His major research interest are schedule generation for manufacturing systems, information management in design and design automation of products.