Dynamic composition of holonic processes to satisfy timing constraints with minimal costs

Fu-Shiung Hsieh

1This paper is supported in part by NSC under grant NSC97-2410-H-324-017-MY3
2Department of Computer Science and Information Engineering
3Chaoyang University of Technology, Taiwan
4e-mail: fshsieh@cyut.edu.tw

Abstract

The flexible architecture provided by holonic manufacturing systems (HMS) poses challenges in planning and control of production processes. The challenges are due, in part, to the loosely coupled structure of holons and also to the complex interactions among holons. Development of new methodologies is required to optimize the holonic processes in HMS to achieve the objectives. In this paper, we concentrate on the development of method for the composition of holonic processes. We consider the holonic processes composition (HPC) problem to synthesize processes with minimal costs while meeting the timing constraints in HMS. We formulate this problem based on a hybrid model in which contract net protocol is adopted as the negotiation protocol and timed Petri net is used to analyze the timing and resource constraints. To specify the costs of operations, we augment the timed Petri net with a cost function. We formulate an optimization problem to minimize the cost while meeting the timing constraints based on the Petri net models. A solution to HPC can be represented by a collaborative Petri net. Our methodologies include a condition to check whether the timing constraints can be met, a condition for the existence of an optimal solution to the HPC problem and a multi-layer contract net protocol to find the minimal cost solution.

Key words: Holonic manufacturing systems, contract net, Petri-nets, process composition.

1. Introduction

The manufacturing sector has been facing major challenges as it undergoes revolutionary changes fuelled by new and sophisticated demands from customers, global competition, distribution of manufacturing and marketing activities and technological advances. Due to the dynamically changing characteristics of manufacturing environments, static control structures are not suitable anymore. Bousbia and Trentesaux reviewed state-of-the-art and future trends of self-organization in distributed manufacturing control (Bousbia and Trentesaux, 2002). The increasing versatility in product demands calls for a system architecture able to evolve in time (Trentesaux and Tahon, 1995). In order to address these challenges, manufacturing enterprises need to change the way they do business and adopt innovative organizational forms, technologies and solutions to increase their responsiveness and production efficiency. To cope with the increased rate of changes that was affecting the entire business world, the idea of using the holonic concept in the design of manufacturing systems emerged in the early 1990s (Agility Forum, 1997). Holonic manufacturing systems (HMS) (Balasubramanian 2001; Wyns, 1999; Brussel, 1998; Christensen, 1994) provide a reconfigurable, flexible and decentralized manufacturing environment to accommodate changes and meet customers’ requirements dynamically based on the notion of holon (Koestler, 1967), an autonomous, co-operative and intelligent entity able to collaborate with other holons to process the tasks.

In HMS, a holon can be part of another holon and a system of holons that can autonomously cooperate to achieve a goal forms a holarchy. A challenge is to design a mechanism to guide the holons such that the decisions made by the individual holons as a whole composite a holarchy that achieves the objectives such as meeting customers’ demands and due dates. In existing literature, there are many studies on HMS (McFarlane and Bussmann, 2000; Gou et al., 1998; Ramos, 1996; Brennan and Norrie, 2001, Wullink, Giebels & Kals, 2002; Giebels, Kals and Zijm, 2001; Neligwa and Fletcher, 2003; Leitão, Colombo and Restivo, 2003; Hsieh, 2006; Leitão & Restivo, 2006; Hsieh, 2008). There are many works on planning and optimization in HMS. Giebels, Kals and Zijm (Wullink, Giebels & Kals, 2002; Giebels, Kals and Zijm, 2001) proposed the system architecture of a flexible manufacturing planning and control system, named EtoPlan, for concurrent manufacturing planning and control system. Existing works on how to specify formally the dynamic behaviour of holonic systems appear in (Leitão, Colombo and Restivo, 2003; Johnson, 2003; Hsieh, 2006). In (Karageorgos, Mehandjiev, Weichhart, and Hämmerle, 2003), the authors present an approach based on agent negotiation with an extended contracting protocol for supporting logistics and
production planning, taking into account not only availability but also cost of logistic service providers. Studies on how should the production control structure evolve to adapt to changes have been made by Leitão et al. (Leitão & Restivo, 2006). Leitão et al. present an agile and adaptive manufacturing control architecture that addresses the need for the fast reaction to disturbances at the shop floor level, increasing the agility and flexibility of the enterprise. Existing results on how to achieve global optimisation in decentralised systems can be found in (Hsieh 2008).

In a real-world manufacturing environment, finding the right sequences and associated schedules with resource, precedence and timing constraints is a difficult task. One approach to overcome the limitations of classical scheduling is the use of distributed schemes such as agent or holonic-based control architectures (Babiceanu, Chen & Sturges, 2005). Babiceanu, Chen & Sturges presents a solution for scheduling using the holonic control approach (Babiceanu, Chen & Sturges, 2005) in which a feasible solution emerges from the combination of individual material handling holons’ solutions. Leitão and Restivo (Leitão and Restivo, 2007) presents a holonic approach to manufacturing scheduling, where the scheduling functions are distributed by several entities, combining their calculation power and local optimization capability.

In (Hsieh, 2008), the author defined a holarchy formation problem to lay a foundation to propose models and develop collaborative algorithms to guide the holons to form a holarchy that coherently moves toward the desired goal state with minimal costs. However, timing constraints have not been taken into account in (Hsieh, 2008). In this paper, we will concentrate on the development of methodology to compose holonic processes with minimal costs while satisfying timing constraints. Instead of applying the complex and optimized scheduling algorithms found in traditional approaches, we extend the distributed collaborative mechanism proposed in (Hsieh, 2008).

A holonic process is dynamically formed by a set of product holons and resource holons based on a certain task distribution protocol such as contract net protocol (CNP) to execute a task. CNP (Smith, 1980) is a well known protocol for distributing tasks. Application of CNP for task allocation in HMS is found in (Neligwa and Fletcher, 2003; Gou et al., 1998; Ramos, 1996). Formation of holonic processes in HMS based on CNP has been studied in (Hsieh, 2008, Hsieh, 2006; Hsieh, 2004), where Petri net (Murata, 1989) models have been proposed to capture the interactions between resource holons and product holons. These results pave the way for the development of methodology for composing holonic processes that satisfy timing constraints. To achieve the objectives, we first formulate a holonic process composition problem based on Petri nets and propose a method to find an optimal solution.

This paper differs from (Giebels, Kals and Zijm, 2001) in that a formal approach based on Petri nets is adopted. Although contract net protocol has been used in this paper, our work is differentiated from (Karageorgos, Mehandjiev, Weichhart, and Hännmerle, 2003) as we focus on optimization of holonic production processes with timing constraints. This paper is differentiated from the works of Leitão et al. (Leitão, Colombo and Restivo, 2003; Leitão and Restivo, 2006) in several aspects. Although all these works intend to contribute to the development of a dynamic and adaptive control approach that improves the agility and reaction to unexpected disturbances by taking advantage of the flexibility offered by HMS without compromising the global optimisation, the approaches are different. The self-organization adaptation mechanism proposed by Leitão et al. introduces the concept of autonomy factor and a pheromone-like spreading mechanism to propagate emergence and reorganize the system. Our approach combines a multi-layer contract net protocol with timed Petri net models augmented with costs to achieve optimization based on collaboration of holons. The problem considered in this paper is more general than that of (Hsieh, 2008) as the timing constraints have been considered in this work.

The remainder of this paper is organized as follows. In section 2, we describe and state the holonic processes composition problem (HPC). To formulate the HPC problem, we introduce timed Petri net models in section 3. We propose a collaborative Petri net model and formulate an optimization problem for the HPC problem. In section 4, we propose a condition for collaborative workflows to satisfy given time constraints. In section 5, we study the condition to determine whether a holon is feasible and also the condition for the existence of an optimal solution to the optimization problem. In section 6, we illustrate how to apply a multi-layer contract net protocol to find the optimal solution by using an example. Section 7 concludes this paper.

2. Composition of Holonic Processes

An HMS consists of three types of holons: resource holons, product holons and order holons (Wyns, 1999). A resource holon consists of production resources with relevant components to control the resources. A product holon contains the production process information to manufacture products. An order holon represents an order. Individual product holons or resource holons cannot process a complex task alone. To process a task, a set of resource holons and product holons form a composite holon called a holarchy.

![Fig. 1 Holonic process formation](image)
formed in HMS with seven product holons \( h_1, h_2, \ldots, h_7 \) and ten resource holons \( r_1, r_2, \ldots, r_{10} \) to accomplish a task with timing constraints. Holonic processes are production processes dynamically created based on the collaboration of product holons. Each product holon has an internal process flow. Execution of the internal process of a product holon may rely on the outputs from the internal processes of one or more upstream product holons. For example, product holon \( h_5 \) and \( h_6 \) depends on either \( h_1 \) or \( h_2 \) to provide the type-one parts and also depends on either \( h_3 \) or \( h_4 \) to provide the type-two parts. Product holon \( h_7 \) depends on either \( h_5 \) or \( h_6 \) to provide the type-three parts.

Execution of a task \( \tau \) requires collaboration of a set of product holons and resource holons. To process a task, the internal processes of a minimal set of product holons need to be connected. One way to form a holarchy is based on the contract net protocol (CNP). Formation of holonic processes in HMS based on CNP has been studied in (Hsieh, 2008), where CNP is applied to form a holarchy base on establishment of contracts between a set of product holon and a set of resource holons. In this paper, we consider the composition of minimal cost holonic processes that execute a task \( \tau \) while satisfying timing constraints.

To describe the dependency between product holons in forming a holarchy for executing a task \( \tau \), we use a circle to represent a product holon. To describe the inputs or outputs of a product holon, we use a set of circles called interface nodes. The output interface node of a product holon may connect to the input interface node of another product holon. The dependency between product holons is represented by a digraph \( D(N, E) \).

Definition 2.1: The dependency between product holons is described by a digraph \( D(N, E) \), where \( N = H \cup I \) is a set of nodes, \( E \) is a set of arcs, a node in \( H \) denotes a holon and a node in \( I \) denotes an interface between holons. An arc connecting node \( h \in H \) to node \( i \in I \) means that holon \( h \) will provide its outputs to other holons via interface node \( i \). An arc connecting node \( i \in I \) to node \( h \in H \) means that holon \( h \) may accept the outputs from other holons via interface node \( i \).

Fig. 2 Dependency digraph \( D(N, E) \)

Fig. 2 shows a dependency digraph \( D(N, E) \), where \( H = \{h_1, h_2, \ldots, h_7\} \) and \( I = \{i_1, i_2, i_3\} \). In Fig. 2, product holons \( h_1 \) and \( h_2 \) are two alternative ways to provide intermediate type-i_1 parts to \( h_3 \) and \( h_6 \) while product holons \( h_3 \) and \( h_4 \) are two alternative ways to provide intermediate type-i_3 parts to \( h_5 \) and \( h_6 \). Product holons \( h_5 \) and \( h_6 \) are two alternative ways to provide intermediate type-i_3 parts to \( h_7 \) to finish the processing.

In a given digraph \( D(N, E) \), the number of outgoing arcs of a holon node \( h \) is called the out-degree of \( h \). A product holon node \( h \) with zero out-degree is called a final product holon. In Fig. 2, \( h_7 \) is a final product holon. We use \( \text{Out}(h) \) and \( \text{In}(h) \) to denote the set of output interface nodes and the set of input interface nodes of product holon \( h \), respectively.

Fig. 3 (a) Minimal Collaborative Networks: \( C_1 \) and (b) \( C_2 \)

Fig. 3 (c) Non-minimal Collaborative Networks \( C_3 \)

Definition 2.2: A collaborative network for product holon \( h \) is described by a digraph \( C(N_C, E_C) \subseteq D(N, E) \), where \( N_C \) is a finite set of nodes and \( E_C \) is a finite set of arcs. \( N_C = H_C \cup I_C \), \( H_C \subseteq H \), \( I_C \subseteq I \), \( h \in H_C \), and for each \( h' \in H_C \), there exists \( h'' \in H_C \) connecting to \( i \) for each \( i \in \text{In}(h) \). If there is exactly one incoming arc and one outgoing arc for each interface node in \( I_C \), \( C(N_C, E_C) \) is called a minimal collaborative network for product holon \( h \).

There are eight minimal collaborative networks \( C_1 \sim C_8 \) to perform the task ending with product holon \( h_7 \) in Fig. 2, where \( H_{C_1} = \{h_1, h_3, h_6, h_7\} \), \( H_{C_2} = \{h_1, h_3, h_5, h_7\} \), \( H_{C_3} = \{h_1, h_4, h_5, h_7\} \), \( H_{C_4} = \{h_2, h_3, h_5, h_7\} \), \( H_{C_5} = \{h_1, h_4, h_6, h_7\} \), \( H_{C_6} = \{h_2, h_3, h_6, h_7\} \), \( H_{C_7} = \{h_2, h_4, h_6, h_7\} \). Fig. 3 (a) and (b) show two minimal collaborative
networks $C_1$ and $C_2$ to perform the task ending with final product holon $h_7$, where $H_{C_1} = \{ h_1, h_3, h_6, h_7 \}$ and $H_{C_2} = \{ h_1, h_3, h_4, h_7 \}$. Fig. 3 (c) shows a non-minimal collaborative network in which there are two holons $h_1$ and $h_2$ connecting to $i_1$.

In general, the number of minimal collaborative networks that can perform a given task grows exponentially with the size of $D(N, E)$. The existence of multiple minimal collaborative networks for a task poses an optimization issue. An important problem is to minimize the overall cost while forming a minimal collaborative network to perform a task with a timing constraint. Let $\omega$ denote the timing constraints. Let $w_c$ denote the cost of $C(N_c, E_c)$ in the holonic processes composition problem can be stated informally as follows:

**Holonic Processes Composition (HPC) Problem:**

$$\min w_c \quad \text{s.t.} \quad C(N_c, E_c) \subseteq D(N, E)$$

A challenge in the development of a solution methodology to HPC problem is due to the distributed computing environment of holonic systems. An effective solution method must be compliant with the holonic architecture. The solution approach proposed in this paper is based on interactions of holons in cooperative, distributed architecture that exhibit the characteristics of HMS. To describe the details of our solution method for the HPC Problem, a mathematical formulation is required. To capture the interactions of holons through the underlying workflows and analyzing the relevant timing requirements, we propose timed Petri net models.

Although the digraph model $C(N_c, E_c)$ clearly represents the dependency between holons, it cannot be used to analyze the dynamics of the collaborative network based on $C(N_c, E_c)$ due to the lack of detailed process models. To formally formulate the optimization problem, we propose a collaborative Petri net model for $C(N_c, E_c)$.

3. Timed Collaborative Petri nets

The collaborative Petri net (CPN) model for $C(N_c, E_c)$ is obtained by combining the corresponding workflow Petri net and with the resource activity Petri net models. A brief introduction to Petri nets is first given. A timed Petri net (PN) $G$ is a five-tuple $G = (P, T, F, m_0, \mu)$, where $P$ is a finite set of places with cardinality $|P|$, $T$ is a finite set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation, $m_0 : P \rightarrow \mathbb{Z}^{\geq 0}$ is the initial marking of the PN with $Z$ as the set of nonnegative integers and $\mu : T \rightarrow Z$ is a mapping that specifies the firing time for each transition. The marking of $G$ is a vector $m \in \mathbb{Z}^{\geq 0}$ that indicates the number of tokens in each place and is a state of the system. In Petri Net, $t^*$ denotes the set of input places of transition $t$ and $t^*$ denotes the set of output places of transition $t$. A transition $t$ is enabled and can be fired under a marking $m$ if and only if $m(p) \geq F(p, t)$.

$$\forall p \in t^* \quad \text{Firing a transition once removes one token from each of its input places and adds one token to each of its output places. A marking } m \text{ is reachable from } m \text{ iff there exists a firing sequence bringing } m \text{ to } m'. \text{ The reachability set } R(m_0) \text{ denotes all the markings reachable from } m_0. \text{ The readers may refer to (Murata, 1989) for a tutorial on Petri nets.}

In modeling the internal workflow of a product holon, we use a place to denote a production state whereas a transition to represent an operation. The workflow Petri net of product holon $h$ is defined as follows.

$$W_h \quad \mu \quad \text{as the set of out-}$$

**Fig. 4 Workflow Petri nets $W_{h_1} - W_{h_3}$ for product holons $h_1 - h_3$**

**Definition 3.1:** The workflow Petri net of product holon $h$ is an acyclic Petri net $W_h = (P_h \cup P^{in}_h \cup P^{out}_h, T_h, F_h, m_0, \mu_h)$, where each place in $P_h$ has exactly one input transition and exactly one output transition, each place in $P^{in}_h$ has exactly one output transition but no input transition and each place in $P^{out}_h$ has exactly one input transition but no output transition and $m_0(p) = 0 \forall p \in P_h \cup P^{in}_h \cup P^{out}_h$.

**Definition 3.2:** The null operations of task $\tau$ are denoted by a set of transitions $T_\tau$. Without loss of generality, we assume there is only one transition in $T_\tau$. That is, $T_\tau = \{ t_\tau \}$.

A product holon $h$ with $t_\tau \in T_h$ is a final product holon.

Fig. 4 illustrates the workflow Petri nets of the seven product
When a type-activity of type $\Phi$, to merge multiple Petri nets through common places, the resource is not used, it stays at idle state in resources, where $\Omega$. A resource may take part in a set of activities, Remark that there is a one-to-one mapping between a place in $P_{h_{out}}$ and an interface node in $In(h)$. There is a one-to-one mapping between a place in $P_{h_{out}}$ and an interface node in $Out(h)$.

Fig. 5 Workflow Petri net for $W_D$

To analyze the system based on Petri net models, we define the operators $\|\|$ to merge multiple Petri nets through common places, transitions, or arcs.

Definition 3.3: Given two Petri nets $G_1 = (P_1, T_1, F_1, m_{10}, \mu_1)$ and $G_2 = (P_2, T_2, F_2, m_{20}, \mu_2)$, $G_1 \| G_2 = (P, T, F, m_0, \mu)$, where $P = P_1 \cup P_2, T = T_1 \cup T_2$,

$$F(p, t) = \begin{cases} F_1(p, t) & \text{if } p \in P_1 \text{ and } t \in T_1 \\ F_2(p, t) & \text{if } p \in P_2 \text{ and } t \in T_2 \end{cases}$$

$$F(t, p) = \begin{cases} F_1(t, p) & \text{if } p \in P_1 \text{ and } t \in T_1 \\ F_2(t, p) & \text{if } p \in P_2 \text{ and } t \in T_2 \end{cases}$$

$$m_0(p) = \begin{cases} m_{10}(p) & \text{if } p \in P_1 \\ m_{20}(p) & \text{if } p \in P_2 \end{cases}$$

$$\mu(t) = \begin{cases} \mu_1(t) & \text{if } t \in T_1 \\ \mu_2(t) & \text{otherwise} \end{cases}$$

We construct a workflow Petri net model $W_D = \|_{h \in H} W_h$ for a given dependency digraph $D(N, E)$. Fig. 5 shows the model $W_D = \|_{h \in H} W_h$ for the dependency digraph $D(N, E)$ in Fig. 2, where $H = \{h_1, h_2, h_3, h_4, h_5, h_6, h_7\}$ and the corresponding workflow Petri nets $W_{h_1} - W_{h_7}$ are shown in Fig. 4. To model the workflow associated with a collaborative network $C$, we define the collaborative workflow as follows.

Definition 3.4: The collaborative workflow associated with a collaborative network $C(N_C, E_C)$ is $W_C = \|_{h \in H_C} W_h$.

According to Definition 2.2, it follows that $W_C$ is connected. Fig. 6 shows the collaborative workflow associated with $C_1$ in Fig. 3 (a). A collaborative workflow does not take into account the interactions with resources. Resource Petri net models are proposed to capture the dynamics of resources. Let $R$ denote the set of resource types in the system. A type-$r$ resource may take part in a set of activities $\Omega_{hr}$ involved in $W_h$. The $k$-th activity of type-$r$ resources, where $k \in \Omega_{hr}$, is described by an activity Petri net $A^k_C = (P^k_C, T^k_C, F^k_C, m^k_{i0}(p)), \mu^k_C$, where $m^k_{i0}(p)$ is the number of resources allocated. Fig. 7(a), 7(b), 7(c) and 7(d) show the resource activity models associated with $h_1, h_3, h_6$ and $h_7$, respectively. Remark that $T^k_C \cap T^k_C = \Phi$ for $k \neq k'$. When a type-$r$ resource is not used, it stays at idle state place $p_r$. To represent the idle states for all types of resources, we define a set of idle state places as follows.

Definition 3.5: $P_{e} = \{p_r | r \in R\}$ is the set of idle state places of all resources. Each place in $P_{e}$ is the idle state of a certain type of resource. To model the interactions among resources and a collaborative workflow we construct a collaborative Petri net (CPN) $G_C(m_0) = (P, T, F, m_0, \mu) = \|_{h \in H_C} W_h \|_{r \in R, k \in \Omega_{hr}} A^k_C$ by merging $A^k_C$. 

Fig. 6 Collaborative Workflow $W_C, H_C = \{h_1, h_3, h_6, h_7\}$
be a minimal cost feasible execution sequence with workflow $W_h \forall h \in H_C$.

A CPN is an instance of holonic process to execute a given task. Fig. 8 shows the CPN corresponding to $C_1$ in Fig. 3(a). Each CPN $G_C$ is a subnet of $G = \bigcap_{h \in H} W_h \bigcap_{r \in R : k \in \Omega_w} A_r^k$. The problem to find a minimal cost feasible execution sequence to execute a task $\tau$ can be formulated based on $G$. We need the following definition.

**Property 3.1:** Let $\omega(M) = \sum_{p \in \mu} m_0(p)$ denote the number of times transition $\tau$ appears in firing sequence $s$. Firing sequence $s$ is a feasible execution sequence for executing task $\tau$ if $\omega(M) = 0$ and $m_0(p) > 0 \forall p \in P$. A subnet $G_C$ is defined by $G_{C_1} = \{h_1, h_2, h_3, h_4\}$.

**Definition 3.7:** Let $\nu(t)$ denote the number of times transition $\tau$ appears in firing sequence $s$. Firing sequence $s$ is a feasible execution sequence for executing task $\tau$ if $\nu(t) \geq 1$. Property 3.1: Let $s$ be a minimal cost feasible execution sequence. Then each transition in $s$ must be distinct.

**Definition 3.8:** A feasible execution sequence $s$ is said to satisfy time constraint $\omega(t)$ if it completes a task $\tau$ by time instant $\omega(t)$.

**Definition 3.10:** A subnet $G' \subseteq G$ is a feasible subnet if there exists a feasible execution sequence $s$ for executing task $\tau$. A feasible subnet $G' \subseteq G$ is said to satisfy a timing constraint $\omega(t)$ if it exists a feasible execution sequence $s$ that completes the task $\tau$ by time instant $\omega(t)$.

**Definition 3.11:** The cost of $G_\omega$ is defined by $w(G_\omega) = \sum_{\tau \in \Omega_w} \mu(\tau)$.

Based on Definition 3.10 and Definition 3.11, the HPC problem can be formulated as follows.

**Optimization Problem:**

$$\min_{G_C \subseteq G} w(G_\omega)$$

The aforementioned cost minimization problem aims to find an execution sequence that accomplishes a task with minimal costs. As the operations are represented by transitions in a Petri net model, a prerequisite for an execution sequence to accomplish a task is that the liveness of the transitions involved must be maintained. Otherwise, the task

---

**Fig. 7(a)** Activity Petri net $A^k_1$ involved in $h_1$

**Fig. 7(b)** Activity Petri net $A^k_1$ involved in $h_3$

**Fig. 7(c)** Activity Petri net $A^k_1$ involved in $h_6$

**Fig. 7(d)** Activity Petri net $A^k_1$ involved in $h_7$

**Fig. 8** Collaborative Petri net $G_{C_1} = \{h_1, h_2, h_3, h_4\}$

Definition 3.6: $M_0 = \{m_0(p) = 0 \forall p \in P \setminus P_0$ and $m_0(p) > 0 \forall p \in P\}$ denotes the set of initial markings of $G$.
cannot be terminated. The cost optimization problem is closely related to liveness of Petri nets.

A solution to the minimal cost firing sequence problem must satisfy the minimal resource requirement (MRR). A MRR of \( G_C(m_0) \) is an initial marking of \( G_C(m_0) \) with the minimal tokens for the existence of a control policy to keep \( G_C(m_0) \) live. The set of MRR of \( G_C(m_0) \) is denoted by \( M_0' \).

We have the following property.

Property 3.2: \( G_C(m_0) = \prod_{h \in H_C} W_h(\{ e \in R, k \leq \Omega_{ho} \} A^h_k) \) is feasible if and only if \( m_0(p_r) \geq m^*(p_r) \forall r \in R \), where \( m^* \in M_0' \) (Property 5.2 in (Hsieh, 2008)).

The computational complexity to compute \( m^* \) is NP-Complete for workflows with tree structure. Theorem 3.1 provides an upper bound of \( m^* \).

Theorem 3.1: Given \( G_C(m_0) = \prod_{h \in H_C} W_h(\{ e \in R, k \leq \Omega_{ho} \} A^h_k) \), \( m^*(p_r) \leq \Omega_r \forall r \in R \) (Theorem 5.2 in (Hsieh, 2008)), where \( \Omega_r \) denotes the set of type- \( r \) activity circuits involved in \( W_C = \prod_{h \in H_C} W_h \).

It follows from Theorem 3.1 that the marking in Fig. 8 satisfies the timing constraints.

4. Condition for Satisfying Timing Constraints

Individual product holons must satisfy certain timing constraints so that the overall collaborative workflow can meet the timing requirements. The timing constraints for a product holon are determined by the timing constraints imposed on the downstream product holons. For the system in Fig. 1, suppose the task must be finished by time instant \( \omega_t \) in \( h_7 \). To achieve this objective, product holon \( h_7 \) imposes further timing constraints \( \omega_t \) and \( \omega_5 \) on \( h_5 \) and \( h_6 \), respectively. These timing constraints mean \( h_5 \) must finish the task by \( \omega_t \) and \( h_6 \) must finish the task by \( \omega_5 \), respectively. To find the collaborative workflow that satisfy the timing constraints, we must compute the timing constraints \( \omega_t \) for each product holon \( h_n \). In this section, we propose a method to calculate the timing constraints.

Whether a collaborative workflow Petri net can meet the timing requirements is determined by the transition firing time and the structure of the net. To facilitate the calculation of the timing constraint for each transition in \( W_h \), we construct a reversed Petri net model \( W'_h \).

Definition 4.1: \( W'_h \) is a reversed Petri net obtained by reversing the direction of each directed arc in the Petri net workflow model \( W_h \).

Figure 9 shows the reversed Petri net models \( W'_{h_1}, W'_{h_2}, W'_{h_6}, W'_{h_1} \) corresponding to the workflow models \( W_{h_1}, W_{h_2}, W_{h_6}, W_{h_1} \) in Figure 4.

Suppose the timing constraint imposed on \( h \) is \( \omega_7 \). The timing constraint for each transition in \( W_h \) can be calculated by firing each transition in \( W'_h \) once, starting with the final transition \( t \). Each token generated in \( W'_h \) is assigned a token value \( \omega(p) \), where \( p \) denotes the place in which the token resides. To compute the timing constraint for each transition in \( W'_h \), we fire each transition in \( W'_h \) once. Let \( t \) be a transition in \( W'_h \) with *\( t \) = \{ \( p \) \} and \( t^* = \{ p_1, p_2, \ldots, p_k \} \). Once transition \( t \) is fired, the token value \( \omega(p) \) is updated as follows: \( \omega(p) = \omega(p) - \mu(t) \forall k \in \{1, 2, \ldots, K\} \).

For the example in Figure 9, suppose the timing constraint imposed on \( h_7 \) is \( \omega_7 \). Transition \( t_{31} \) is fired first. Firing \( t_{31} \) brings a token to \( i_3 \). The timing constraint for \( i_3 \) is \( \omega(i_3) = \omega_7 - \mu(t_{31}) \). As \( i_3 \) is the output of \( W_{h_6} \), the timing constraint for \( h_5 \) is \( \omega_5 = \omega(i_3) \). To calculate the timing con-
transit for each transition in \( W_h \), we put a token at place \( i_3 \) in \( W_h \) and assign a token value \( \omega(i_3) \) to it. Transition \( t_{26} \) is then fired and a token is brought to place \( p_{20} \) with token value \( \omega(p_{20}) = \omega_5 - \mu(t_{26}) \). Next, transition \( t_{22} \) is then fired. One token is brought to place \( p_{18} \) with token value \( \omega(p_{18}) = \omega(p_{20}) - \mu(t_{22}) \) and the other token is brought to place \( p_{19} \) with token value \( \omega(p_{19}) = \omega(p_{20}) - \mu(t_{22}) \). Next, transition \( t_{22} \) is then fired and a token is brought to place \( i_1 \) with token value \( \omega(i_1) = \omega(p_{19}) - \mu(t_{22}) \). Finally, transition \( t_{22} \) is then fired and a token is brought to place \( i_2 \) with token value \( \omega(i_2) = \omega(p_{19}) - \mu(t_{22}) \). This completes the calculation of token value for each place in \( W_h \). Based on the token value for each place, the timing constraint for each transition in \( W_h \) can be found easily. For example, the timing constraint to start firing transition \( t_{23} \) is \( \omega(t_{23}) = \omega(i_1) \). The timing constraint to start firing transition \( t_{24} \) is \( \omega(t_{24}) = \omega(i_2) \).

By following a similar procedure, the timing constraint for each \( W_h \) can be calculated. Fig. 9 shows the calculation of token values associated with the timing constraints for \( W_h, W_h, W_h, W_h \) based on \( W_h, W_h, W_h, W_h \).

Let \( W_h = (p_h \cup P_h^m \cup P_h^s, T_h, F_h, m_h, h_h) \). Let \( \omega_t \) denote the timing constraint for transition \( t \in T_h \). The following property states the condition

Property 4.1: If \( \omega_t \geq 0 \) for each \( t \in T_h \), product holon \( h \) satisfies the timing constraints.

By removing the workflow Petri nets not satisfying the timing constraints from a collaborative workflow \( W_C \), we obtain a collaborative workflow \( \tilde{W}_C \) that satisfies timing constraints.

Example: For Fig. 4, suppose \( h_1 \) imposes a timing constraint \( \omega_1 \) on \( h_2 \). Based on \( \omega_1 \), \( h_3 \) imposes timing constraints on \( h_1, h_2, h_3, \) and \( h_4 \). Suppose that \( \omega_t \geq 0 \) for each \( t \in T_h \) for \( h \in \{h_2, h_3, h_4\} \). Suppose there exists \( t \in T_h \) with \( \omega_t < 0 \). In this case, \( h_1 \) does not satisfy the timing constraints and is excluded from the collaborative workflow.

Fig. 10(a) shows the resulting collaborative workflow \( \tilde{W}_1 = W_h \| W_h \| W_h \| W_h \| W_h \).

Example: For Fig. 4, suppose \( h_1 \) imposes a timing constraint \( \omega_1 \) on \( h_2 \). Based on \( \omega_1 \), \( h_6 \) imposes timing constraints on \( h_1, h_2, h_3, \) and \( h_4 \). Suppose that \( \omega_t \geq 0 \) for each \( t \in T_h \) for \( h \in \{h_1, h_2, h_3, h_4\} \). Fig. 10(b) shows the resulting collaborative workflow \( \tilde{W}_2 = W_h \| W_h \| W_h \| W_h \| W_h \).

5. Condition for the existence of a solution

Although timing constraints have been taken into consideration in constructing the collaborative workflows in the previous section, there may not always exist a solution to the aforementioned optimization problem if there are sufficient resources available. For the existence of a solution, there must be sufficient resources available. In (Hsieh, 2008), we have shown that the existence of a feasible execution sequence can be characterized based on the existence of a feasible execution sequence for individual product holons. Existence of a feasible execution sequence for an individual product holon \( h \) can be verified by the minimal resource requirements (MRR) of the corresponding Petri net.
model $\hat{G}_h$, which is obtained by adding an input transition for each input interface place of $G_h$ and adding an output transition for each output interface place of $G_h$ and adding arcs to connect the associated resource idle places, where $G_h = \mathcal{W}_h \parallel \bigoplus_{i=1}^{n_h} A_i^h$.

Example: In Fig. 11, $|\Omega_{h_r}| = 2$, $|\Omega_{h_r}| = 3$, $|\Omega_{h_r}| = 2$, $|\Omega_{h_r}| = 1$.

Therefore, $m^*_h(r_1) \leq 2$, $m^*_h(r_2) \leq 3$, $m^*_h(r_3) \leq 2$, $m^*_h(r_4) \leq 1$.

Example: In Fig. 11, $|\Omega_{h_r}| = 2$, $|\Omega_{h_r}| = 1$, $|\Omega_{h_r}| = 1$.

Therefore, $m^*_h(r_3) \leq 1$, $m^*_h(r_5) \leq 1$, $m^*_h(r_6) \leq 1$, and $m^*_h(r_5) \leq 2$.

It follows from Theorem 5.1 that the following holds.

Property 5.2: $\hat{G}_h$ with initial marking $m_{i0}(p_r)$ is feasible if $m_{i0}(p_r) \geq |\Omega_{h_r}| \forall r \in R$.

A solution for the optimization problem must consist of a set of feasible product holons. Therefore, any product holon that is not feasible must be excluded. For example, suppose $\hat{G}_h$ is feasible for all $h$ with the exception of $h_2$. In this case, we exclude $h_2$ from $\hat{W}_i$ and obtain $\hat{W}_i'$ as shown in Fig. 12(a). Similarly, we exclude $h_2$ from $\hat{W}_2$ and obtain $\hat{W}_2'$ as shown in Fig. 12(b). In the remainder of this paper, we use $\hat{W}_C'$ to denote the Petri net obtained by removing infeasible product holons from a collaborative workflow $\hat{W}_C$ that satisfies timing constraints.

For the existence of a solution to the optimization problem, there must exist a collaborative workflow $\hat{W}_C' \subseteq \hat{W}_C$ such that there exists at least one upstream holon connecting to each interface place in $\hat{W}_C'$. Fig. 12(a) illustrates a net $\hat{W}_C'$. For which there does not exist any solution to the optimization problem due to the lack of upstream holons connecting to place $i_1$. Inspiring by this example, the structural condition for the existence of an optimal solution to the optimization problem can be stated as follows.

Property 5.3: Let $\hat{W}_C'$ denote the Petri net obtained by removing infeasible product holons from a collaborative workflow $\hat{W}_C'$ that satisfies timing constraints $\omega$. There exists a solution to the optimization problem only if there exists at least one upstream feasible holon connecting to each interface place in $\hat{W}_C'$.

Property 5.3 imposes a constraint on the structure of $\hat{W}_C'$. A net $\hat{W}_C$ that satisfies Property 5.3 implies that there exists a solution to the optimization problem.

To find the optimal solution, we define the cost of a product holon as follows.

Definition 5.1: The cost of product holon $h \in H$ is defined by $\omega_h = \sum_{t \in T_h} \mu(t)$.

To characterize an optimal collaborative network, we need the following definition.

Definition 5.2: Let $C_h(N_h, E_h) \subseteq D(N, E)$ be a minimal collaborative network for product holon $h$, where $N_h = H_h \cup I_h$.
\( \subseteq H \cup I \). The cost of \( C_h \) is defined by 
\[ w(C_h) = \sum_{h' \in H_h} w_{h'} \cdot C_h \]
which is an optimal collaborative network for product holon \( h \) if 
\[ w(C_h) = \min_{C_h \in C(h)} \]
where \( C(h) \) denotes the set of all minimal collaborative networks for product holon \( h \).

Definition 5.3: \( U_i(i) = \{ h' | h' \text{ connects to } i, \text{ where } i \in \text{ln}(h) \} \)
denotes the set of holons directly connecting to the interface node \( i \) of holon \( h \) in \( D(N, E) \).

Property 5.3: Suppose \( C_h \) is a minimal cost collaborative network for product holon \( h \). Then the following equation holds:
\[ w(C_h) = w_h + \sum_{i \in \text{ln}(h), h' \in U_i(i)} \min w(C_{h'}) \]
If product holon \( h \) is a source product holon, \( \text{ln}(h) = \Phi \) and the aforementioned equation reduces to 
\[ w(C_h) = w_h \].

Based on the previous definition, the optimal solution can be found by selecting the minimal cost feasible upstream product holons. For each \( i \in \text{ln}(h) \), product holon \( h \) selects \( h' \in U_i(i) \) with minimal \( w(C_{h'}) \). If there exists at least one \( h' \in U_i(i) \) for each \( i \in \text{ln}(h) \), there exists an optimal solution.

---

**Fig. 12(a) \( \tilde{W}_1 \) for \( \tilde{W}_1 \)**

**Fig. 12(b) \( \tilde{W}_2 \) for \( \tilde{W}_2 \)**

Example 1: The process to select the minimal cost feasible upstream product holons starts from the product holon in the upstream. For \( \tilde{W}_2 \) in Fig. 12(b), assume 
\[ w(C_{h_6}) < w(C_{h_6}) \]. As \( \text{ln}(h_6) = \{ i_1, i_2 \} \), \( h_6 \) will select the minimal cost product holons that connect to \( i_1 \) and \( i_2 \). As there is only one feasible product holon \( h_1 \) connecting to \( i_1 \), \( h_1 \) is selected by \( h_6 \). As there are two product holons \( h_2 \) and \( h_4 \) connecting to \( i_2 \), \( h_6 \) will select the one with minimal cost. As \( w(C_{h_i}) < w(C_{h_i}) \), \( h_3 \) is selected by \( h_6 \). As \( \text{ln}(h_3) = \{ i_3 \} \), \( h_7 \) will select the minimal cost product holons that connect to \( i_3 \). As \( h_5 \) is the only holon connecting to \( i_3 \), \( h_7 \) will select \( h_5 \). The resulting optimal solution is shown in Fig. 13.

6. **Multi-layer contract net to find a minimal cost solution**

A multi-layer contract net protocol is applied to form a collaborative network to execute a task. The multi-layer contract net protocol consists of four types of messages: CFP(Call For Proposal), SOP(Submission Of Proposal), AOC(Awarding Of Contract) and EOC(Establishment of Contract). CFP consists of several fields, including part type requirements, timing constraints and cost requirements. SOP consists of the fields of part type provided, workflow model and cost offered. An AOC message is used to notify the minimal cost bidders based on evaluation of the submitted proposals. An EOC message is used by the awarded bidder to confirm the establishment of contracts with the manager.

A holon that acts as a manager issues a CFP message to the potential bidders. On receiving a CFP message, the bidder may issue a CFP message in turn to the potential bidders. A potential bidder may submit a proposal to the corresponding manager by sending a SOP message. Fig. 13 shows the flow chart to handle CFP. After evaluating the proposals submitted by the bidders, a manager sends an AOC message to the best bidders. On receiving an AOC message, the best bidders respond to the manager by sending an EOC message to establish the contracts.

By removing the workflow Petri nets not satisfying the timing constraints from a collaborative workflow \( W_C \), we obtain a collaborative workflow \( \tilde{W}_C \) that satisfies timing constraints.

Example: Fig. 14 shows the UML sequence diagram that details the interactions between product holons for the example in Fig. 1. The value of \( w(C_{h}) \) are listed in Table 1. Suppose Holon \( h_7 \) issues CFP1 to potential bidders \( h_5 \) and \( h_6 \). Holon \( h_5 \) issues CFP2 in turn to \( h_1, h_2, h_3 \) and \( h_4 \). Similarly, Holon \( h_6 \) issues CFP3 to \( h_1, h_2, h_3 \) and \( h_4 \). On receiving CFP2, \( h_1, h_2, h_3 \) and \( h_4 \) check Property 4.1 and Property 5.2 to determine whether to submit their proposals. As \( h_j \) does not satisfy Property 4.1 for \( h_5 \), it will not submit any proposal to \( h_5 \). As \( h_2 \) does not satisfy Property 5.2, it will not submit any proposal to \( h_5 \). As a result, \( h_5 \) only receives two proposals SOP1 and SOP2 from \( h_3 \) and \( h_4 \), respectively. As the proposals do not satisfy Property 5.3, \( h_5 \) will not submit any proposal to \( h_7 \). On receiving CFP3, \( h_1, h_2, h_3 \) and \( h_4 \) check Property 4.1 and Property 5.2 to determine whether to submit their proposals. As \( h_j \) satisfies the timing constraints whereas \( h_2 \) does not satisfy Property 5.2, only \( h_1, h_3 \) and \( h_4 \) submit their proposals by sending SOP3 ~ SOP5 to \( h_7 \). Based on the proposals, holons \( h_6 \) confirm that SOP3 ~ SOP5 sat-
isfy Property 5.2 and Property 5.3. Therefore, \( h_6 \) submits a proposal SOP6 to \( h_7 \). Holon \( h_7 \) then issues AOC1 message to \( h_6 \). On receiving AOC1, \( h_6 \) must determine the best bidders. Assume that \( w(C_{h_6}) < w(C_{h_j}) \), the minimal cost bidder is \( h_j \). Therefore, \( h_6 \) issues AOC2 to \( h_j \) and AOC2 to \( h_3 \). Finally, all holons that have received AOC respond by sending EOC messages back to the corresponding managers.

### Table 1 Values of \( w(C_{h_i}) \)

<table>
<thead>
<tr>
<th>( h )</th>
<th>( w(C_{h_i}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_1 )</td>
<td>5</td>
</tr>
<tr>
<td>( h_2 )</td>
<td>10</td>
</tr>
<tr>
<td>( h_3 )</td>
<td>20</td>
</tr>
<tr>
<td>( h_4 )</td>
<td>30</td>
</tr>
<tr>
<td>( h_5 )</td>
<td>20</td>
</tr>
<tr>
<td>( h_6 )</td>
<td>10</td>
</tr>
<tr>
<td>( h_7 )</td>
<td>15</td>
</tr>
</tbody>
</table>

### Fig. 14 Interaction diagram for finding optimal solution \( C_{h_i} \)

7. **Conclusion**

To meet the changing customers’ demands with minimal costs while delivering the products timely, manufacturers rely on a flexible, effective architecture to dynamically collaborate with the partners to fulfill the orders’ requirements, optimize the costs and reduce the risk. Holonic manufacturing systems provide a reconfigurable, flexible and decentralized manufacturing environment to accommodate changes and meet customers’ requirements dynamically based on the notion of holons. This paper presents a systematic methodology to synthesize minimal cost production processes that satisfy timing constraints based on holonic system architecture. A holonic process is dynamically formed by a set of product holons and resource holons to execute a task. We consider the problem to synthesize holonic processes with minimal costs while meeting the timing constraints in holonic systems. We formulate this problem based on a hybrid model in which contract net protocol is adopted as the negotiation protocol and Petri net is used to specify and analyze the timing and resource constraints. To specify the costs of operations, we augment the Petri net with a cost function. We formulate an optimization problem to minimize the cost while meeting the timing constraints based on the Petri net models. Our method for solving the optimization is broken down into two parts: a method to find the collaborative workflows.
that satisfy timing constraints and a method to find the minimal cost solution. All holons that cannot meet the timing requirements are excluded. We also establish a simple condition to test whether there exists an optimal solution based on the structure of the collaborative workflow. We illustrate how to apply a multi-layer contract net protocol to find the optimal solution by using an example. As our methodology is generic and developed based on holistic system architecture, it can be tailored for specific HMS applications.

References


