Fault Tolerant Analysis For Holonic Manufacturing Systems Based On Collaborative Petri Nets

Fu-Shiung Hsieh
Overseas Chinese Institute of Technology
100 Chiao Kwang Rd. Taichung, 407, Taiwan, R.O.C.

Abstract

Uncertainties are significant characteristics of today's manufacturing systems. Holonic manufacturing systems are new paradigms to handle uncertainties and changes in manufacturing environments. Among many sources of uncertainties, failure prone machines are one of the most important ones. This paper focuses on handling machine failures in holonic manufacturing systems. Machine failure will reduce the number of available resources. Feasibility analysis need to be conducted to check whether the works in process can be completed. To facilitate feasibility analysis, we characterize feasible conditions for systems with failure prone machines. This paper combines the flexibility and robustness of multi-agent theory with the modeling and analytical power of Petri net to adaptively synthesize Petri net agents to control holonic manufacturing systems. The main results include: (1) a collaborative Petri net (CPN) agent model for holonic manufacturing systems, (2) a feasible condition to test whether a certain type of machine failures are allowed based on collaborative Petri net agents and (3) fault tolerant analysis of the proposed method.

Keywords: Holonic manufacturing system, Petri net, fault tolerant

1. INTRODUCTION

Due to the capabilities to deal with changes and uncertainties in today's manufacturing environment, holonic manufacturing systems (HMS)([1]-[6]) have emerged as a paradigm for developing such adaptive manufacturing systems. A Holonic Manufacturing System can be modeled as a cooperative multi-agent system ([7]-[8]) with precedence constraints of production processes and constraints of finite resources. In existing literatures, several design issues of holonic manufacturing systems have been addressed. For example, [1]-[3] addressed modeling of HMS, and [4]-[5] addressed deadlock avoidance control of HMS and [6] addressed scheduling of HMS.

There are three types of basic elements in HMS: resource agent, product agent, and order agent according to [1] and [2]. A resource agent consists of a production resource and relevant components that control the resource. A product agent contains the production process and product knowledge to ensure the correct fabrication of products with sufficient quality. An order agent represents a manufacturing order. The three types of agents interact with each other through the contract net protocol and the algorithms proposed in [9]. The features of HMS pose several challenging design issues. For example, in order to effectively utilize existing resources to achieve the production goal, a coalition process is required to collaboratively accomplish the assigned tasks. Due to resource sharing among distinct processes, resource contention is inevitable. An effective coordination mechanism is required to resolve conflicts in usage of resources. In [5] and [9], we proposed a distributed collaborative network formation process based on contract net protocol ([10]) and multi-agent framework to organize a set of resources to accomplish the assigned task in manufacturing systems. However, the impacts of uncertainties on HMS have not been analyzed.

The capability of HMS to handle uncertainties in manufacturing environments such as failure prone machines and changing processes requires further study. Among many sources of uncertainties, failure prone machines are one of the most important ones. This paper focuses on analyze the impacts of machine failures in holonic manufacturing systems. In HMS, a machine is modeled as a resource agent. To facilitate analysis, a resource agent unavailability model is proposed to capture the effects of machine failure. We classify failure of machines into three categories according to its structure: (1) failure of one or more agents at the same operation and (2) failure of multiple agents at distinct operations. Based on resource agent unavailability model, a fault tolerant analysis of HMS is conducted.

In our previous paper [11], we have proposed a framework for control of HMS based on Petri net agents. This paper will focus on analysis of the fault tolerant property of our approach. Depending on the orders to be processed, the required product holon agents and the resource holon agents will be collaboratively and adaptively form a number of collaborative Petri net (CPNs) to fulfill the order requirements. Depending on the resource configuration, the effects of machine failure vary. Machine failure will reduce the number of available resources. Feasibility analysis needs to be conducted to check whether the works in process system can be completed.

To facilitate feasibility analysis, we characterize feasible conditions for systems with failure prone machines. For a resource agent participating in a place of some non-critical token flow path of a CPN, its failure will not destroy the liveness property. The original resource configuration can still be applied. Otherwise, the liveness property of the CPN may not preserve.

This paper combines the flexibility and robustness of multi-agent system architecture with the modeling and analytical power of Petri net ([12]-[16]) to adaptively synthesize Petri net agents to control holonic manufacturing systems. The main results include: (1) a nominal collaborative Petri net (CPN) agent model for holonic manufacturing systems, (2) a resource agent unavailability model to model machine failure , (3) several feasible condition to test whether a certain type of machine failures are allowed based on Petri net agents and (4) an algorithm to handle machine failure in holonic manufacturing systems. The remainder of this paper is organized as follows. Section 2 reviews the agent coalition processes and proposes a nominal collaborative Petri net model resulting from the agent coalition processes. Section 3 presents liveness condition for nominal CPNs. Section 4 establishes fault tolerant property for CPNs. Section 5 concludes this paper.

2. NOMINALCOLLABORATIVE PETRI NET AGENTS

In our previous paper [11], we have proposed a framework to control HMS based on Petri net agents and contract net protocol [10]. In contract net protocol, there are two roles agents can play: manager or bidder. Four steps are involved for establishment of a contract between a manager and one or more bidders:

Step1: Request for tender: In this stage, the manager announces a task to all potential bidders as shown in Figure 1(a). The announcement contains the detailed description of the task.

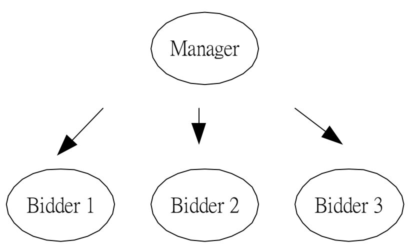

Figure 1(a) Step 1: Request for tenders

Step2:Submission of proposals: On receiving the tender announcement, bidders capable of performing the task draw up proposals and submit to the manager as shown in Figure 1(b).

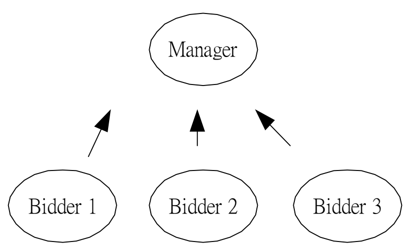

Figure 1(b) Step 2: Submission of proposals

Step3:Awarding of contract: On receiving and evaluating the submitted proposal, the manager awards the contract to the best bidder as shown in Figure 1(c).

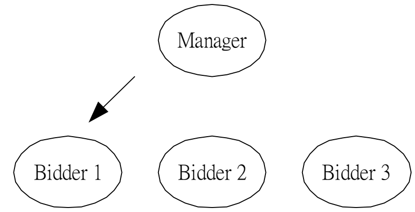

Figure 1(c) Step 3: Awarding of contract

Step4:Establishment of contract: If the awarded bidder commit itself to carry out the required task, it will send a message to the manager and become a contractor as shown in Figure 1(d). Otherwise, the awarded bidder might refuse to accept the contract by notifying the manager. The manager will reevaluate the bids and award the contract(s) to another bidder(s).

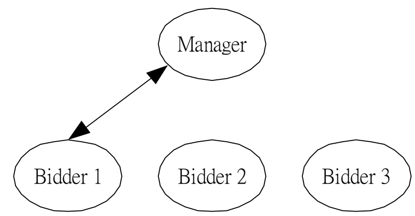

Figure 1(d) Step 4: Establishment of contract

We proposed a distributed collaborative network formation process for HMS based on contract net protocol ([11]). The order holon agents always act as managers to initiate tender processes with the product holon agents as the bidders. The product holon agents act as the managers in turn to issue new request for tenders to resource holon agents for the set of subtasks stemming from the product requirements. The algorithms end up with one or more collaborative networks. A collaborative network is defined as follows.

Definition 2.1: A collaborative network is a directed graph $C = (A_c, E_c)$ for a certain task assigned, where A_c is the set of nodes representing agents in the collaborative network and E_c is the set of directed edges that connect all the agents in A_c with commitment relationship.

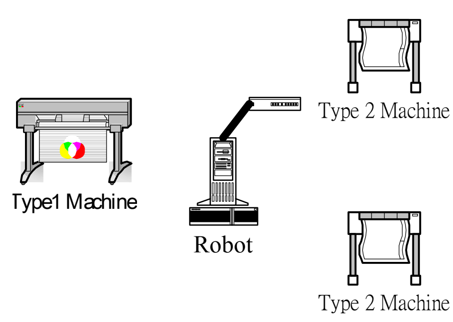

Figure 2

Example 1:Consider a manufacturing system with one Type 1 machines and two Type 2 machines. The system layout is shown in Figure 2. There is a production process that follows the production route described in Table 1, where OP-i is the i-th operation of the production process. Figure 3 illustrates a collaborative network, where r_i represents type-i machine, p_j represents the product agent corresponding to OP-j and o_i represents an order agent. A solid directed edge from resource agent r_i to product agent p_j means that r_i is required by p_j and r_i has established contract with p_j. A dashed directed eadge from resource agent r_i to product agent p_j means that r_i is required by p_j but r_i has not established contract with p_j. A black product agent p_j means that p_j has acquired the required resource agent and the resource agent is dedicated to the operation of the product agent. A collaborative network is therefore an effective method to represent commitment among agents.

operation	OP-1	OP-2	OP-3	OP-4
resource	Type1 machine	Type 2 machine	Type 1 machine	Type 2 machine

Table 1

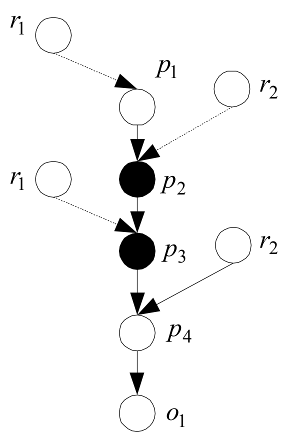

Figure 3

However, collaborative network is lack of the capability to compute the release of resources. To achieve this goal, we convert collaborative networks to Petri nets. Petri Net is a powerful tool for modeling, control and analysis of manufacturing systems. A Petri Net (PN) N is a five-tuple $N = (P, T, I, O, m_0)$, where P is a finite set of places with cardinality $|P|$, T is a finite set of transitions, $I \subset P \times T$ is a set of transition input arcs, $O \subset T \times P$ is a set of transition output arcs, and $m_0 : P \rightarrow Z^{|P|}$ is the initial marking of the PN with Z as the set of nonnegative integers. The marking of N is a vector $m \in Z^{|P|}$ that indicates the number of tokens in each place and is a state of the system. The readers may refer to [12] for further definitions such as enabled transitions, transition firing rules and the set of reachable markings of the PN N from an initial marking m_0, denoted as $R(m_0)$. Please refer to [11] for the details to construct the CPN of a given collaborative network. The CPN constructed using the above procedure is a deterministic , acyclic Petri net defined as follows.

Definition 3.1: A CPN associated with a collaborative commitment graph $j \in \boldsymbol{J}$ is defined as a six tuple $N_j = (P_j, T_j, I_j, O_j, m_{j0}, u_j)$, abbreviated as $N_j(m_{j0}, u_j)$, where u_j is a commitment policy defined based on commitment actions of a given CPN as follows.
In a CPN, a token denotes an agent and transitions represent execution or termination of a contract.

Definition 3.2: A commitment action a is a vector in $Z^{|P_j|}$ that determines how many times that each transition in T_j may be fired concurrently. We will use $a(t)$ to denote the number of transition firing allowed under a. A commitment policy u_j is a mapping that generates a commitment action for the CPN N_j based on its current marking m_j. That is, $u_j : R(m_{j0}) \rightarrow Z^{|P_j|}$.

A commitment action a is a vector that determines the number of idle resource tokens to be allocated to different operations. As distinct collaborative networks may compete for the same types of resources, the marking m_j under a feasible commitment action a must satisfy the following constraints:

$\sum_{j \in \boldsymbol{J}} \sum_{p \in \boldsymbol{P}_{jr}} m_j(p) \le c_r$, where c_r denotes the total number of type- r resources and P_{jr} denotes the set of resource allocation places involved with type- r resources.

The nominal Petri Net model corresponding to Figure 3 is shown in Figure 4.

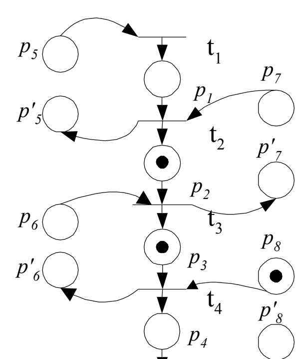

Figure 4

3. LIVENESS CONDITIONS FOR NOMINAL CPN

In Petri net theory, liveness is an important property as it guarantees each transition (corresponding to operation) in it can be fired infinitely many times. To maintain the liveness of a CPN, a minimal set of resources is required. Lack of one or more resources makes one or more transitions unable to fire. The minimal set of resources to maintain the liveness of a CPN N_j can be characterized as a vector $\underline{R}_j \in Z^{|\boldsymbol{R}|}$. In [11], we have developed an algorithm to compute $\underline{R}_j$. Theorem 3.1 establishes a condition to check the liveness of N_j based on $\underline{R}_j$.

Theorem 3.1: Given a CPN N_j with marking $m_j \in \mathcal{M}(m_{j0})$, there exists a commitment policy u_j such that $N_j(m_j, u_j)$ is live if and only if there exists a sequence of commitment actions that bring m_j to a marking m'_j under which $R_j(m_j) \geq \underline{R}_j$, where $R_j(m_j)$ denotes the set of resources in idle state.

Proof:

Sufficiency: Let t_j^f be the last transition of N_j. Let s_j^* denote a minimal firing sequence to fire t_j^f. Assume a marking m'_j is reached from m_j after firing a sequence of transitions under a sequence of commitment actions c_1, c_2,..., c_n with $R_j(m'_j) \geq \underline{R}_j$. As $R_j(m'_j) \geq \underline{R}_j$, it implies s_j^* can be fired under m'_j. Let u'_j denote the commitment policy that fires s_j^* repeatedly. Let u_j be the commitment policy that applies the sequence of commitment actions $c_1, c_2,..., c_n$ to m_j first and then applies u'_j under m'_j. It follows that $N_j(m_j, u_j)$ is live.

Necessity: Given the fact that $N_j(m_j, u_j)$ is live, we show that there exists a commitment policy u''_j that completes all the existing jobs and brings m_j to a marking m'_j under which all resources are returned to idle states. Let u''_j be the job clearing commitment policy obtained from u_j by allowing only the minimum number of transitions required to be fired to complete the existing jobs for the in-process parts. We claim that u''_j will clear all the existing jobs under m_j and reach a marking m'_j under which all resources are returned to idle state and all the existing jobs under m_j are completed and cleared. This must be true because if the existing jobs under m_j cannot be cleared by u''_j, the existing jobs cannot be cleared by u_j either and $N_j(m_j, u_j)$ cannot be kept live, which is a contraction. We prove that $R_j(m'_j) \geq \underline{R}_j$ by contradiction. Suppose $R_j(m'_j) < \underline{R}_j$. There must exist some $j \in \boldsymbol{J}$ and $r \in \boldsymbol{R}$ with $\underline{R}_j(r) > R_j(m', r)$. As all resources are returned to idle state under m'_j, this implies N_j cannot be kept live under m'_j since there exists at least one transition whose resource requirement $\underline{R}_j(r)$ cannot be met under $R_j(m', r)$ and cannot be fired any more. This contradicts to the fact that N_j is live. Q.E.D.

Theorem 3.1 states a necessary and sufficient liveness condition for a single collaborative network. The result can be extended for systems with multiple collaborative networks. Consider a system with a set $\mathcal{N} = \{ N_j, j \in \boldsymbol{J} \}$ of CPNs. Let us denote the system with the set $\mathcal{N}$ of CPNs as $N = \cup_{j \in \boldsymbol{J}} N_j$, where $N \equiv (P, T, I, O, m, u)$, abbreviated as $N(m, u)$ or N and u is jointly defined by $u_j, j \in \boldsymbol{J}$. The following Theorem establishes a liveness condition based on $N = \cup_{j \in \boldsymbol{J}} N_j$.

Theorem 3.2: Given a set of CPNs $N = \cup_{j \in \boldsymbol{J}} N_j$ with marking m, there exists a commitment policy u such that $N(m, u)$ is live if and only if there exists a sequence of commitment actions that bring m to a marking m' under which $R(m') \geq \underline{R}$, where $R(m') \in Z^{|\boldsymbol{R}|}$ denotes the set of resources in idle state under m' and $\underline{R}$ is a minimal set of resources required to complete the tasks of N with $\underline{R} \equiv \sum_{j \in \boldsymbol{J}} \underline{R}_j$.

Proof:

Sufficiency: Let s_j^* denote a minimal firing sequence to fire t_j^f. Assume a marking m' is reached from m after firing a sequence of transitions under a sequence of commitment actions c_1, c_2,..., c_n with $R(m') \geq \underline{R}$. As $R(m') \geq \underline{R} \geq \underline{R}_j$, it implies s_j^* can be fired under m'_j. Let u'_j denote the commitment policy that fires s_j^* repeatedly and u' be the commitment policy that applies u'_1, u'_2,..., $u'_{|\boldsymbol{J}|}$ sequentially. Let u_j be the commitment policy that applies the sequence of commitment actions c_1, c_2,..., c_n to m' first and then applies u' under m'. Hence $N_j(m_j, u_j)$ is live for each $j \in \boldsymbol{J}$. It follows that $N(m, u) = \cup_{j \in \boldsymbol{J}} N_j(m_j, u_j)$ is live.

Necessity: Given the fact that $N = \cup_{j \in \boldsymbol{J}} N_j$ is live, we now show that there exists a commitment policy u'' that completes all the existing jobs and brings m to a marking m' under which all resources are returned to idle state. Let u''_j be the job clearing commitment policy obtained from u by allowing only the minimum number of transitions to be fired to complete the existing jobs while keeping the same commitment policy as u for the in-process parts. We claim that u''_j will clear all the existing jobs of N_j under m_j and reach a marking m'_j under which all resources are returned to idle state and all the existing jobs under m_j are completed and cleared. This must be true because if the existing jobs under m_j cannot be cleared by u''_j, the existing jobs cannot be cleared by u_j either and $N_j(m_j, u_j)$ cannot be kept live, which is a contraction. As the same reasoning holds for each $j \in \boldsymbol{J}$, all the jobs can be completed and cleared after $u'_1, u'_2, ..., u'_{|\boldsymbol{J}|}$ have been applied and the system will reach a marking m'. We prove $R(m') \geq \underline{R}$ by contradiction. Suppose $R(m') < \underline{R}$. There must exist some $j \in \boldsymbol{J}$ and $r \in \boldsymbol{R}$ with $\underline{R}_j(r) > R(m', r)$. As all resources are returned to idle state under m', this implies N_j cannot be kept live under m' as there exists at least one transition whose resource requirement $\underline{R}_j(r)$ cannot be met under $R(m', r)$ and cannot be fired any more. This contradicts to the fact that N is live. Q.E.D.

Theorem 3.1 implies that as long as the set of resources that can be released from the current system state dominates the MRR, the liveness of the system can be maintained. By exploiting the structure of the sequential production processes, release of resources can be evaluated based on the acyclic marked graph MG_j associated with type- j production process. A procedure to obtain MG_j based on decomposition of a given CPPN G_c has been proposed. Please refer to [11] for details. Let P_{jr} denote the set of places to which a resource in use by type- j production process may be released and return to idle state.

Definition 3.1: A token flow path π is a directed path consisting of alternating places and transitions. A token flow path starting with a place p_1 without input transitions and ending with a place p_2 without output transitions is denoted as $\pi(p_1, p_2)$. We use $p \in \pi$ to denote that p is a place of π.

Definition 3.2:The total number of tokens in a token flow path π in an acyclic type- j marked graph MG_j under submarking m_j is denoted as $\pi(m_j) = \sum_{p \in \pi} m_j(p)$.

Definition 3.3: Let $\Gamma_{jr}(p_{ro})$ denote the set of token flow paths for type- r resources ending with a place $p_{ro} \in P_{jr}$.

As each acyclic Marked Graph MG_j, $j \in \boldsymbol{J}$, is a deterministic Petri Net, the number of type- r resources that may stay at or be released to the idle state place of type- r resources under control action a and marking m is $\sum_{j \in \boldsymbol{J}} \sum_{p_{ro} \in P_{jr}} \min_{\pi \in \Gamma_{jr}(p_{ro})} \pi(m_j)$. Combining the above result with Theorem 3.1, the following Corollary holds.

Corollary 3.1: There exists a control policy u such that $G_c(m, u)$ is live if $\sum_{j \in \boldsymbol{J}} \sum_{p_{ro} \in P_{jr}} \min_{\pi \in \Gamma_{jr}(p_{ro})} \pi(m_j) \geq m^*(p_r(0))$, where $m \in \mathcal{M}(m_0)$.

4. FAULT TOLERANT PROPERTY OF CPN

In a CPN, each token corresponds to either an agent. If a resource agent has established contract with other agents but fails to execute the contract, the original commitment may not be feasible. An agent that fails to execute the established contract is modeled as a removal of token in CPN. We let δm denote the change due to removing tokens from the CPN. Depending on the system configuration, we classify δm into three categories according to its structure:

(I) removing tokens from a specific state place p of the CPN:

$\boldsymbol{\delta M}_1(m) \equiv \bigcup_{p \in P} \boldsymbol{\delta M}_p(m)$, where

$\boldsymbol{\delta M}_p(m) = \{ \delta m | 0 \leq \delta m(p) \leq m(p)$ and $\delta m(p') = 0$ for all $p' \in P - \{p\} \}$. This type corresponds to failure of one or more agents at the same operation of some CPN.

(II) removing tokens from a subset of places of one CPN:

$\boldsymbol{\delta M}_2(m) \equiv \bigcup_{j \in \boldsymbol{J}} \bigcup_{F_j \subseteq P_j} \boldsymbol{\delta M}_2(m, F_j)$,

where $\boldsymbol{\delta M}_2(m, F_j) \equiv \{ \delta m | \delta m = \sum_{p \in F_j} \delta m_p$, where $\delta m_p \in \boldsymbol{\delta M}_p(m)$ and $F_j \subseteq P_j \}$. This type corresponds to failure of multiple agents at more than one operations of a given CPN.

In this Section, we characterize the tolerable failure of agents based on the condition of Corollary 3.1. To convey the idea, consider the inequality stated in Corollary 3.1:

$$\sum_{j \in \boldsymbol{J}} \sum_{p_{ro} \in P_{jr}} \min_{\pi \in \Gamma_{jr}(p_{ro})} \pi(m_j) \geq m^*(p_r(0)) \quad (1)$$

Inequality (1) implies that $\min_{\pi \in \Gamma_{jr}(p_{ro})} \pi(m_j)$ remains intact as long as the change $\delta m_j(p)$ of the decrease in the number of tokens of a place p does not reduce the sum of tokens in any of the token flow path $\pi \in \Gamma_{jr}(p_{ro})$ such that $\pi(m'_j) \leq \min_{\pi \in \Gamma_{jr}(p_{ro})} \pi(m_j)$, where m'_j denote the perturbed marking after $\delta m_j(p)$ units of tokens have been removed from m_j. Based on this observation, the following definition is required to convey the above concept. The set of all token flow paths is devided into two categories: critical paths and non-critical paths. Removing one token from a place in a non-critical path has no effect on the liveness property of a CPPN. Removing one token from a place in a critical path will reduce the number of tokens that will be released and may destroy the liveness property of the CPPN.

Definition 4.1: Under control action a and marking m, a token flow path in $\Gamma_{jr}(p_{ro})$ with the total number of tokens along the path equal to $\min_{\pi\in\Gamma_{jr}(p_{ro})} \pi(m_j)$ is called a critical path. The set of critical paths in $\Gamma_{jr}(p_{ro})$ is denoted as $\Gamma^c_{jr}(p_{ro})$ ={ π | $\pi(m_j)$ = $\min_{\pi\in\Gamma_{jr}(p_{ro})} \pi(m_j)$ and $\pi\in\Gamma_{jr}(p_{ro})$ }.

Definition 4.2: $\Gamma^n_{jr}(p) = (\Gamma_{jr} - \Gamma^c_{jr}(p)) \cap \Gamma_{jr}(p)$ represents the set of non-critical paths for Type-r resources in MG_j under control action a and marking m, where $p\in P_{jr}$.

Based on the above definition, the main result is stated as follows.

Theorem 4.1: Given a set of CPNs $N = \bigcup_{j\in \boldsymbol{J}} N_j$ with marking m, for any change $\delta m\in \boldsymbol{\delta M}_p(m)$ with $\delta m_j(p) = \delta_p$ for some $j\in \boldsymbol{J}$, the number of type-r resources that will be released to p_{ro} is decreased by

$\delta\gamma(p_{ro}, m_j) = \delta_p * \alpha(j, p_{ro}, p, m_j) - \beta(j, p_{ro}, p, m_j)$, where

$\alpha(j, p_{ro}, p, m_j)$ =0 and $\beta(j, p_{ro}, p, m_j)$ =0 if $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$,

$\alpha(j, p_{ro}, p, m_j)$ =1 and $\beta(j, p_{ro}, p, m_j)$ =0 if $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$,

$\alpha(j, p_{ro}, p, m_j)$ =0 and $\beta(j, p_{ro}, p, m_j)$ =0 if $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j) - \delta_p \geq 0$

$\alpha(j, p_{ro}, p, m_j)$ =1 and

$\beta(j, p_{ro}, p, m_j) = \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j)$

if $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j) - \delta_p < 0$,

$\alpha(j, p_{ro}, p, m_j)$ =1 and $\beta(j, p_{ro}, p, m_j)$ =0 if $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$.

Proof:

As $\delta m\in \boldsymbol{\delta M}_p(m)$, there must exist a $j\in \boldsymbol{J}$ such that $\delta m_j(p) > 0$ and $\delta m_j(p') = 0$ for all $p'\in P_j$ -{ p }, and $\delta m_{j'}(p) = 0$ for all $p\in P_{j'}$, $j'\in \boldsymbol{J} - \{j\}$. $\min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j - \delta m_j) = \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j)$ and $\min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j - \delta m_j) = \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j)$.

$$\gamma(p_{ro}, m_j) - \delta\gamma(p_{ro}, m_j) \equiv \min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \delta_p, \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \delta_p, \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \} \quad \text{......(2)}$$

If $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, (2) is reduced to

$$\min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \}.$$

In this case, the number of type-r resources that may be released to p_{ro} remain unchanged no matter what δ_p is. For this case, $\alpha(j, p_{ro}, p, m_j)$ =0 and $\beta(j, p_{ro}, p, m_j)$ =0.

If $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, (2) is reduced to

$$\min\{[\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p, \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \}$$

In this case, the number of type-r resources that may be released to p_{ro} will be decreased by δ_p due to the reduction in the number of tokens in place p by δ_p. For this case, $\alpha(j, p_{ro}, p, m_j)$ =1 and $\beta(j, p_{ro}, p, m_j)$ =0.

If $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) = \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, (2) is reduced to

$$\min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} [\pi(m_j) - \delta_p], \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} [\pi(m_j) - \delta_p], \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \}$$

$$= \min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j), \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} [\pi(m_j) - \delta_p], \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \}$$

$$= \min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j), [\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p, \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \}$$

$$= \min\{ \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j), [\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p \}$$

In this case, the number of type-r resources that may be released to p_{ro} will not be decreased due to the reduction in the number of tokens in place p by δ_p if the following holds

$$\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j) - \delta_p \geq 0$$

For this case, $\alpha(j, p_{ro}, p, m_j)$ =0 and $\beta(j, p_{ro}, p, m_j)$ =0.

If $\min\limits_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min\limits_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j) - \delta_p < 0$, the number of type-$r$ resources that may be released to p_{ro} will be decreased by $\delta_p - [\min\limits_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min\limits_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j)]$.

For this case, $\alpha(j, p_{ro}, p, m_j)$ =1 and $\beta(j, p_{ro}, p, m_j) = \min\limits_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) - \min\limits_{\pi\in\Gamma^c_{jr}(p_{ro})} \pi(m_j)$.

If $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, (2) is reduced to

$$\min\{[\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p,\ \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j),\ [\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p,\ \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j)\}.$$

As $\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, and $\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p) \neq \Phi$, we have

$$[\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p \le \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j) \le \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j).$$

By definition,

$$\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j) < \min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j).$$

So

$$[\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p < [\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p.$$

Combining the above results, we have

$$\min\{[\min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p,\ \min_{\pi\in\Gamma^c_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j),\ [\min_{\pi\in\Gamma^n_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p,\ \min_{\pi\in\Gamma^n_{jr}(p_{ro})-\Gamma_{jr}(p)} \pi(m_j)\}$$

$$= \min_{\pi\in\Gamma^c_{jr}(p_{ro})\cap\Gamma_{jr}(p)} \pi(m_j)] - \delta_p$$

In this case, the number of type-r resources that may be released to p_{ro} will be decreased by δ_p due to the reduction in the number of tokens in place p by δ_p. For this case, $\alpha(j, p_{ro}, p, m_j)$ =1 and $\beta(j, p_{ro}, p, m_j)$ =0.

Q.E.D.

Based on Theorem 4.1, we characterize the set of tolerable failure of agents that leaves the system live as follows.

Definition 4.3:

Let $\boldsymbol{\delta\overline{M}}_p(m) \equiv \{\delta m | \delta m \in \boldsymbol{\delta M}_p(m)$ *and* δm *satisfies* (2)$\}$.

$$\sum_{j\in\boldsymbol{J}} \sum_{p_{ro}\in P_{jr}} \min_{\pi\in\Gamma_{jr}(p_{ro})} \pi(m_j) - \delta_p * \alpha(j, p_{ro}, p, m_j) + \beta(j, p_{ro}, p, m) \ge m^*(p_r(0)),\ \forall r\in\boldsymbol{R} \qquad (3)$$

It follows directly from Theorem 4.1 and the above definition that the following theorem offers a sufficient liveness condition for tolerable token loss of a single place.

Theorem 4.2: Given a set of CPNs $N = \bigcup_{j\in\boldsymbol{J}} N_j$ that can be kept live marking m, for any change $\delta m \in \boldsymbol{\delta\overline{M}}_p(m)$, there exists a commitment policy u' under which $N = \bigcup_{j\in\boldsymbol{J}} N_j$ is live.

Example 1(Continued): For Example1, suppose the agent corresponding to machine 2 fails at operation state p_2. Then the CPN in Figure 4 evolves to the one in Figure 5. For this example,

$\Gamma_{jr}(p'_5) = \{p_5t_1p_1t_2p'_5,\ p_7t_2p'_5\}$.

$\Gamma_{jr}(p'_7) = \{p_5t_1p_1t_2p_2t_3p'_7,\ p_7t_2p_2t_3p'_7,\ p_6t_3p'_7\}$.

$\Gamma_{jr}(p'_6) = \{p_5t_1p_1t_2p_2t_3p_3t_4p'_6,$
$p_7t_2p_2t_3p_3t_4p'_6,\ p_6t_3p_3t_4p'_6,\ p_8t_4p'_6\}$.

$\Gamma_{jr}(p'_8) = \{p_5t_1p_1t_2p_2t_3p_3t_4p_4t_5p'_8,$
$p_7t_2p_2t_3p_3t_4p_4t_5p'_8,\ p_6t_3p_3t_4p_4t_5p'_8,\ p_8t_4p_4t_5p'_8\}$

The set of critical paths under the given marking in Figure 4 is as follows.

$\Gamma^c_{jr}(p'_5) = \{p_5t_1p_1t_2p'_5,\ p_7t_2p'_5\}$

$\Gamma^c_{jr}(p'_6) = \{p_6t_3p_3t_4p'_6,\ p_8t_4p'_6\}$

$\Gamma^c_{jr}(p'_7) = \{p_6t_3p'_7\}$

$\Gamma^c_{jr}(p'_8) = \{p_6t_3p_3t_4p_4t_5p'_8,\ p_8t_4p_4t_5p'_8\}$.

Note that as p_2 does not belong to any critical path under Figure 4, removing one token from p_2 has no influence on the liveness of the CPN.

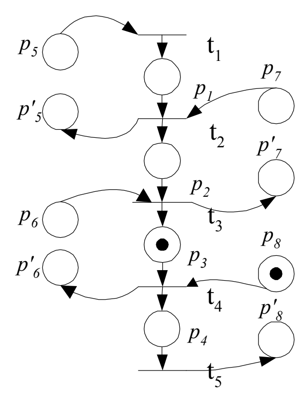

Figure 5

Definition 4.4:Let $\boldsymbol{\delta\overline{M}}_2(m) \equiv \{\delta m | \delta m \in \boldsymbol{\delta M}_2(m)$ and δm satisfies (4) $\}$.

$$\sum_{j\in\boldsymbol{J}} \sum_{p_{ro}\in P_{jr}} \min_{\pi\in\Gamma_{jr}(p_{ro})} \pi(m_j - \delta m_j) \ge m^*(p_r(0)) \quad \forall r\in\boldsymbol{R} \ \ldots(4)$$

Theorem 4.3: Given a set of CPNs $N = \bigcup_{j\in\boldsymbol{J}} N_j$ that can be kept live marking m, for any change $\delta m \in \boldsymbol{\delta\overline{M}}_2(m)$, there exists a commitment policy u' under which $N = \bigcup_{j \in \boldsymbol{J}} N_j$ is live.

5. CONCLUSION

Machine failure is an important type of uncertainties in holonic manufacturing systems. This paper investigates the effects of machine failure on the operation of holonic manufacturing systems. A holonic manufacturing system can be modeled as a set of collaborative Petri nets (CPN), where each CPN corresponds to the set of resources required to complete the operations of a certain production order. Each token in a CPN corresponds to a resource agent. If a resource agent that has established contract with other agent but fails to execute the contract, the original commitment may not be feasible. An agent that fails to execute the established contract is modeled as a removal δm of token in CPN. Depending on the system configuration, we classify δm into three categories: (1) failure of one or more agents at the same operation and (2) failure of multiple agents at distinct operations. Based on this model, we characterize the tolerable machine failure conditions for holonic manufacturing systems.

References

[1] J. Wyns, *Reference Architecture for Holonic Manufacturing,* PhD Dissertation, PMA Division, Katholieke Universiteit, Leuven, Belgium, 1999.

[2] H.V. Brussel, J. Wyns, P. Valckenaers, L. Bongaerts and P. Peeters. Reference architecture for holonic manufacturing systems. 37:255-274, 1998.

[3] Christensen J., Holonic Manufacturing systems – initial architecture and standards directions, in: Proceedings of the *First European Conference on Holonic Manufacturing Systems*, Hannover, December 1994.

[4] Liu S., William A. Gruver, Dilip Kotak, Shahriar Bardi, (2000) Holonic Mamufacturing System For Distributed Control of Automated Guided Vehicles, *Proceedings of the IEEE International Conference on System, Man and Cybernetics*, pp. 1727-1732, Oct. 2000.

[5] Fu-Shiung. Hsieh, " Deadlock Free Task Distribution and Resource Allocation for Holonic Manufacturing Systems based on Multi-Agent Framework", IEEE International Conference on System, Man and Cybernetics, Tucson, Arizona, pp. 2463-2468, Oct. 7-10, 2001.

[6] Gou L, Luh, P. B. Kyoya Y. , Holonic manufacturing scheduling: architecture, cooperation mechanism, and implementation, *Computers in Industry* 37 pp. 213-231, 1998.

[7] Jacques Ferber, "Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence," Addison Wesley, 1999.

[8] N.J. Nilsson, "Artificial Intelligence: A New Synthesis," Morgan Kaufmann Publishers, Inc. San Francisco, California, 1998.

[9] Fu-Shiung. Hsieh, "Modeling and control of holonic manufacturing system based on extended contract net protocol", Proceedings of 2002 American Control Conference, pp. 5037- 5042, Anchorage, Alaska, May 8-10, 2002.

[10] R.G. Smith, "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver", *IEEE Trans. On Computers* Vol. 29, pp. 1104-1113, 1980.

[11]. F. S. Hsieh, "Multi-Agent Control of Holonic Manufacturing Systems Based On Petri Net," *To Appear on Proceedings of the 15-th IFAC World Congress*, 2002.

[12] Tadao Murata, "Petri Nets: Properties, Analysis and Applications," *Proceedings of the IEEE*, vol. 77, No. 4, April 1989.

[13] Y. Narahari and N. Viswanadham,"A Petri net approach to the modeling and analysis of flexible manufacturing systems," *Annals Oper. Res.*, vol. 3, pp. 449-472, 1985.

[14] M. C. Zhou and F. DiCesare, "Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems with shared resources," *IEEE Trans. Robot., Automat.*, vol. 7, no. 4, pp. 515-527, 1991.

[15] M. C. Zhou, F. DiCesare and A.A. Desrochers, "A hybrid methodology for synthesis of Petri net models for manufacturing systems," *IEEE Trans. Robot., Automat.*, vol. 8, no. 3, pp. 350-361, 1992.

[16]. M. D. Jeng and F. DiCesare, "A review of synthesis techniques for Petri nets with applications to automated manufacturing systems," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 1, pp. 301-312, 1993.