International Journal of Computer Theory and Engineering, Vol. 2, No. 2 April, 2010
1793-8201

Analysis of Parallel Manufacturing Processes with Resource Sharing

Farooq Ahmad, Hejiao Huang and Xiao-long Wang

***Abstract*—Multiple resource sharing has a complex nature in parallel manufacturing processes due to the competition of different operations for scarce resources. This paper presents the Petri net model for parallel manufacturing processes with shared resources called parallel process net with resources (PPNR). The structural characteristics of PPNR are discussed and siphon based characterizations of live and reversible PPNR are also presented. Simple structure based conditions and properties for conservative PPNR are presented for checking its liveness and reversibility. Finally, proposed method of net modeling is demonstrated by a practical example.**

***Index Terms*—Petri net, resource sharing, parallel processing flows, liveness.**

I. Introduction

Handling resource sharing becomes an important aspect during flexible manufacturing system (FMS) design and allocation of resources is always a challenging task in the modeling. Further, the resource-sharing among different operations to be performed does not imply its simultaneous usage because each operation utilizes a resource type exclusively and releases it after completion. Furthermore, a resource type held by one operation can not be preempted by another operation.

The parallel flows of multiple products in a FMS can be identified as different jobs to be completed. For instance, parallel or concurrent processing of raw products of different types on limited number of resources such as machines, robots, buffers etc. is a common situation in FMS. While sharing the limited number of resources, these jobs have a particular operation routing that determines the order in which resources must be assigned to the product. Therefore, allocation of limited number of resources to the different operations to achieve the efficient output of FMS is not a trivial task and requires the modeling of operation flows in FMS.

Petri nets (PNs) have been extensively used for modeling, analysis and design of FMS [1]-[6] because of the ability of PNs for formal verification to detect the important behavioral properties of modeled systems. The PN model of a manufacturing system (MS) is either constructed by a top-down or bottom-up approach [7]. In a top-down approach [8]-[10], firstly the high-level description of system is presented and then model is stepwise refined by adding the subnets until a complete net model is achieved. For FMSs with shared resources, the top-down approach has a drawback that the subnets are strongly coupled and it is hard to find the small size of final expanded model. Where as, in bottom-up approach [11]-[13], the net modules of specified subsystems are constructed. Then, they are combined by sharing common places, transitions or subnets. However, the verification of desired set of properties for such type of integrated model is not straightforward. Consequently, the disadvantage of bottom-up approach is the great difficulty in the verification of desired behavior of the large-scale integrated model.

To cope with these shortcomings of modeling approaches, this paper presents a new class of PNs called *parallel process net with resources* (PPNR) for FMSs having parallel processes. There are two stages which comprise the modeling procedure of PPNR. Firstly, the parallel process net (PPN) is constructed to specify the process flows of each part type in a MS without considering the resources. The step of PPN portraits the parallel processing of parts and depicts the order of different operations. Further, this step assists for the proper assignment of resources required to process each part type and provide the resource-allocation policy according to the given production plan. Thereafter, the marked resource places denoting the availability of resource types are added to the PPN. In this way, PPNR can model more complex resource-sharing and interacting parallel processes in FMSs.

The main power of the PN as mathematical tool is its support for analysis to study important characteristics for synthesis after modeling of the physical MS has been performed. Structural analysis illuminates the important structural characteristics of the PN model and useful for its synthesis. One of the main advantages of the structural approach for characterization of the PN model is that it is independent of the reachable states of a system, which is computationally impractical for large-scale MSs. Moreover, structural approach for the analysis of FMSs is mainly based on minimal siphons [12], [14]-[19]. Although the number of siphons grows quickly and in the worst case grows exponentially fast with respect to the PN size [20], [21], even though it is practical because there is no need to generate the reachability graph which suffers from state-explosion problem [22].

This paper presents a number of characterizations of live and reversible PPNR based on siphons and minimal marked siphons. Further, several structural characteristics of PPN and then PPNR are described. These characteristics help to identify the requirements of the structure of PPNR and its behavior. Furthermore, main interest of the simple structural based conditions and properties, presented in this paper, is in

their use for the synthesis of live and reversible PPNR.

The paper is organized as follows. Some related terminologies are introduced in Section 2. In Section 3, the formal definition and the characterization of PPN is presented. Section 4 introduces the PPNR and presents the characterization of live and reversible PPNR. The demonstration of the proposed modeling procedure is presented in Section 5 and concluding remarks are presented in Section 6.

II. Definitions And Concepts

In this section, some basic definitions and notations of ordinary (for the sake of simplicity) PN are described. The related terminology and notations are mostly taken from [23], [24].

Definition 1: (Petri net) A Petri net *PN*, is a five tuple, $PN=(P,T,I,O,M_0)$. Where, $P=\{p_1,p_2,\ldots,p_{|P|}\}$ is a finite set of places, $|P|>0$; $T=\{t_1,t_2,\ldots,t_{|T|}\}$ is a finite set of transitions, $|T|>0$; $I:T\to P$ is the input function, which is a mapping from transitions to the set of places and it indicates the input places of transitions; $O:T\to P$ is the output function, which is a mapping from transitions to the set of places and it indicates the output places of transitions, $P\cap T=\varnothing$ and $P\cup T\neq\varnothing$.

Let $I(t_j)$ represent the set of *input places* of transition $t_j\in T$ and $p_i\in P$ is an input place of a transition t_j if $p_i\in I(t_j)$; $O(t_j)$ represents the set of *output places*, then p_i is an output place of t_j if $p_i\in O(t_j)$.

The input and output functions can be extended to map the set of places P into the set of transitions T such as $I:P\to T$ and $O:P\to T$. Then, set $I(p_i)$ represents the set of *input transitions* of place $p_i\in P$ and set $O(p_i)$ represents the set of *output transitions* of place $p_i\in P$.

The incoming arc from p_i to t_j is represented by $(p_i,I(t_j))$ and outgoing arc from t_j to p_i be $(p_i,O(t_j))$. Similarly, $(t_j,I(p_i))$ represents the incoming arc from t_j to p_i as $t_j\in I(p_i)$ and arc $(t_j,O(p_i))$ represents outgoing arc from p_i to t_j as $t_j\in O(p_i)$, when the set of places maps into the set of transitions; $\forall t_j\in T,\forall p_i\in P$.

The structure of a PN is defined by the set of places, set of transitions, input function and output function. A PN structure without M_0 is denoted by $N=(P,T,I,O)$. A PN structure *N* is said to be strongly connected if and only if every node $x_i\in P\cup T$ is reachable from every other node $x_j\in P\cup T$ by a directed path. A PN structure *N* is said to be *self-loop-free* or *pure* if and only if $\forall t_j\in T$, $I(t_j)\cap O(t_j)=\varnothing$, i.e. no place can be both an input and an output of the same transition. A PN structure *N* is said to be *state-machine* if and only if, $\forall t_j\in T$, $|I(t_j)|=|O(t_j)|=1$ and said to be *marked graph* if and only if $\forall p_i\in P$, $|I(p_i)|=|O(p_i)|=1$.

A marking is a function $M:P\to\mathbb{N}$ (non-negative integers) and initial marking is denoted by M_0. A PN with given initial marking is denoted by (N,M_0). The set of all reachable markings from M_0 is denoted by $R(M_0)$ which is a definite set of markings of PN such that, if $M_k\in R(M_0)$ and $M_k\xrightarrow{t_j}M_k'$ for some $t_j\in T$, then $M_k'\in R(M_0)$.

Definition 2: (*Firing rule*) The firing rule identifies the transition enabling and the change of marking. Let $M(p_i)$ be the number of tokens in place p_i, then for $\forall t_j\in T$; t_j is enabled under marking M if and only if $\forall p_i\in I(t_j):M(p_i)\geq 1$. The change of marking M to M' by firing the enabled transition t_j is denoted by $M\xrightarrow{t_j}M'$ and defined for each place $p_i\in P$ by $M'(p_i)=\begin{cases}M(p_i)-1 \text{ for every } p_i\in I(t_j)\\ M(p_i)+1 \text{ for every } p_i\in O(t_j)\\ M(p_i) \text{ otherwise.}\end{cases}$

Definition 3: (*P-invariant, T-invariant and conservativeness*) For a PN (N,M_0), a P-invariant is a $|P|$-vector $y\geq 0$ such that $Ay=0$, where A is the $|T|\times|P|$ incidence matrix. Similarly, a T-invariant is a $|T|$-vector $x\geq 0$ such that $A^Tx=0$. A PN is said to be conservative if and only if there exists a P-invariant $y>0$.

Definition 4: (*Siphons, traps and minimal siphons*) A set of places $S\subseteq P$ is called a *siphon* if $I(S)\subseteq O(S)$ and it is called a *trap* if $O(S)\subseteq I(S)$. A siphon S is called *minimal* if there does not exits S' such that $S'\subset S$.

Definition 5: (*Liveness*) A transition $t_j\in T$ is said to be *live* if $\forall M_k\in R(M_0)$ there is marking M_k' reachable from M_k such that M_k' enables t_j and PN (N,M_0) is *live* if $\forall t_j\in T$: t_j is live.

Definition 6: (*Reversibility*) A marking M_k of PN (N,M_0) is called *reversible* if and only if for every marking M_k' reachable from M_k there exist a firing sequence reproducing M_k and (N,M_0) is called *reversible* if and only if M_0 is reversible.

III. Parallel Process Net

The process flow of each part type (raw or in-process material) is represented by the token flow in a PN model. The places are used to model the different operations (e.g., machining, holding, assembling and transformation etc.) to be executed over the part types. The resource types (e.g., machines, buffers, robots, etc.) are also modeled by the initially marked places referring to the availability of resources.

The transitions grant to advance a part type into the finished

product as they represent the start and termination of manufacturing processes in the PN model. There are output (input) arcs from resource places to those transitions that move a process to (from) the operation state by using these resources. The sequence of transition firing represents the production path in the PN model of a production plan and there may be several paths to achieve a final product in a same plan.

The *parallel process net* (PPN) is constructed as a first step for the specification of parallel processing flows of each part type in a manufacturing system without considering the resources. The production plan in a manufacturing system has specific input point(s) for raw material and output point(s) for finished product(s). In a PPN, such input points are combined by the single transition with single marked input place as shown in Fig. 1(a). Such single transition, denoted by t_0, initiates the parallel execution of jobs in a modeled FMS. The single input place of transition t_0 is called *initial place*, denoted by p_0, and every cycle in parallel process net include it. In this way the PPN is *strongly connected* and *self loop free*. The output points for finished parallel jobs, which are modeled by places, are synchronized by single transition, called *synchronizing transition* and denoted by t_s, as depicted in Fig. 1(a). A token in a place p_f in Fig. 1(a), called *final place*, represents the completion of processing. The transition which finishes the single production iteration is denoted by t_f where p_f stands for its input place and p_0 as its output place. For formal definition of PPN, the set of places excluding the initial place p_0 and final place p_f is denoted by P_{OP} and the set of transitions representing the start and termination of different operations in PPN is denoted by T_{OP}.

Definition 7: The PPN is a PN $PN = (P, T, I, O, M_0)$ such that

i. $P = P_{OP} \cup \{p_0, p_f\}$

ii. $T = T_{OP} \cup \{t_0, t_s, t_f\}$

iii. $I(p_0) = \{t_f\}$ and $O(p_0) = \{t_0\}$

iv. $I(t_0) = \{p_0\}$ and $|O(t_0)|$ is the number of raw parts entering into the system

v. $I(p_f) = \{t_s\} \vee T_f \subseteq T_{OP}$ and $O(p_f) = \{t_f\}$

vi. $I(t_f) = \{p_f\}$ and $O(t_f) = \{p_0\}$

vii. $M_0(p_0) = 1 \Rightarrow M_0(p_i) = 0 \ \forall p_i \in P \setminus \{p_0\}$

viii. $\forall M_k \in R(M_0)\,;\ M_k(p_f) = 1 \Rightarrow M_k(p_i) = 0$ $\forall p_i \in P \setminus \{p_f\}$

Property 1: Every cycle in PPN contains the initial place p_0, final place p_f and transition t_f.

Proof: The proof for the statement of Property 1 is trivial, as by removing the initial place p_0 final place p_f and transition t_f and its associated arcs, PPN is acyclic.

Property 2: Each isolated parallel processing flow in PPN is executable.

Proof: Every processing flow is a sequence of operations represented by the transitions for their beginning and termination, which can be fired if their input places representing the first operation in order are marked. Since $O(t_0)$ are the set of places representing the specific input points for the processing of parts entering into the system, which become marked by firing initial transition t_0. From (iv) and (vii) of Definition 7, t_0 is executable.

Property 3: In PPN, an individual parallel processing flow is a T-invariant.

Proof: An independent sequence of operations is for completing the processing of a part entering into the system, which is interpreted as production path. Each cycle in PPN contains such type of production path, which is executable in isolation due to Property 2. From Definition 7(vii), $M_0(p_0) = 1$ and each cycle contains the initial place p_0, sequence of transitions in each cycle is a T-invariant.

Property 4: Every cycle in PPN is a *strongly connected state-machine*.

Proof: Form Property 3, every cycle contains the individual parallel processing of a part represented by the sequence of operations. Further every cycle contains common initial place p_0, initial transition t_0, synchronizing transition t_s, final place p_f and final transition t_f. According to the Property 1, by removing them, there are independent parallel processing flows represented by the sequence of operations, where every transition representing either beginning of an operation or end of operation has only one input and output place representing a specific operation. Hence every cycle is the strongly connected state-machine with the transitions representing the firing sequence that completes the processing of a part entering into the system.

Property 5: Every cycle in PPN is a siphon with initial place p_0, final place p_f and transition t_f.

Proof: Trivial, corresponding set of places in a cycle is a siphon. From Property 1, by removing initial place p_0, final place p_f and transition t_f there is no cycle in PPN. Hence every cycle is a siphon in PPN containing p_0, final place p_f and transition t_f.

Property 6: A strongly connected PPN is reversible.

Proof: Every cycle in PPN represents the production routing of parts entering into the system. Further, from Property 3, every cycle is a T-invariant representing the firing sequence that completes the processing of each part. Therefore, M_0 is reachable from any intermediate marking $M_k \in R(M_0)$ in PPN.

Property 7: A strongly connected PPN is live.

Proof: Property 5 follows that every cycle in PPN is a siphon with common place p_0, final place p_f and transition t_f. From Properties 2, 3 and 4, every cycle in PPN contains a production path for a part entering into the system and no other cycle exists in it, which follows that every cycle is not only a strongly connected state-*machine* but also a *marked graph*. This implies that every siphon in PPN is also a trap. From Definition 7(vii), every cycle is marked, which implies that every siphon has a marked trap. Hence PPN is live due to

[23].

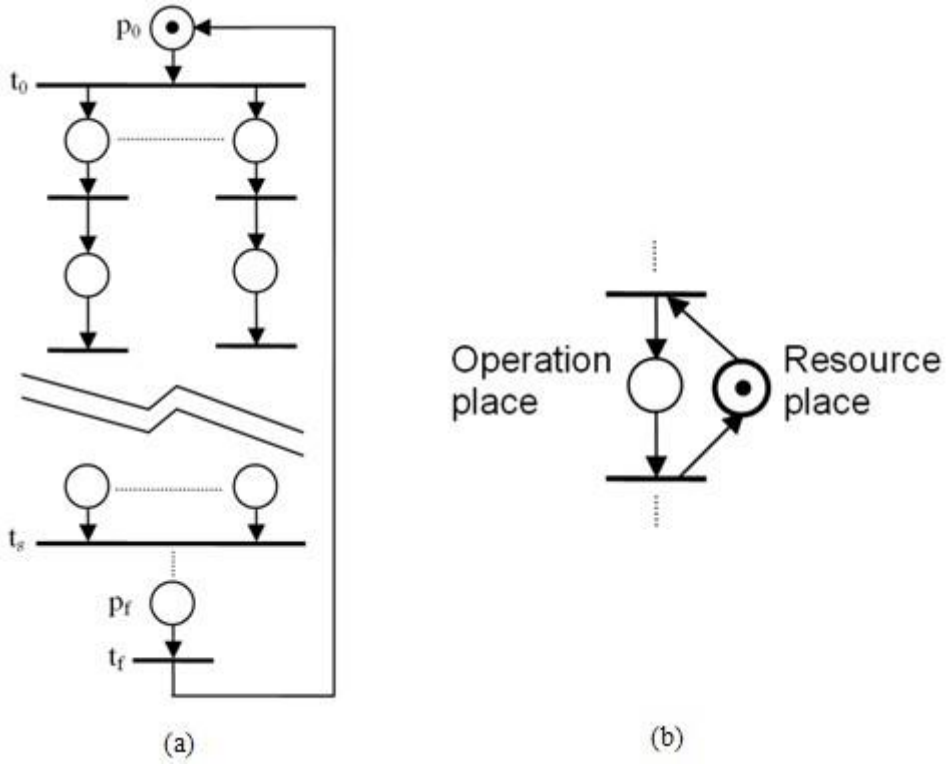

Fig. 1 (a) The sketch of a parallel process net and (b) assignment of a resource to the process

It is extremely desirable that every transition in PPN models an actual event of process execution or termination and there is not a redundant transition in it. Further, every operation can proceed towards its completion without any interruption and execution of any additional operation. In order to make sure the existence of these requirements, the characterization of PPN illuminates that there is no dead transition in it and it is *reversible*. Since attention is focused on efficient resource-allocation to the parallel manufacturing processes, the complete characterization of the class of PPN is beyond the scope of this paper.

IV. Parallel Process Net With Resources

The step of PPN assists for the proper assignment of resources required to process each part type. Thereafter, the marked resource-places denoting the availability of resource types are added to the PPN such that input and output transitions of each operation-place act as its output and input transition respectively, as shown in Fig. 1(b).

The PN model of parallel manufacturing processes with marked resources-places will be called, from now on, *parallel process net with resources* (PPNR).

For the purpose of defining PPNR formally, the set of places P, excluding the initial place p_0 and final place p_f, is divided into the set of operation-places P_{OP} and the set of resource-places P_R.

Definition 8: The PPNR is a PN $PN = (P, T, I, O, M_0)$ such that

i. $P = P_{OP} \cup P_R \cup \{p_0, p_f\}$ and $P_{OP} \cap P_R = \varnothing$

ii. $T = T_{OP} \cup \{t_0, t_s, t_f\}$

iii. $I(p_0) = \{t_f\}$ and $O(p_0) = \{t_0\}$

iv. $I(t_0) = \{p_0\}$ and $O(t_0) \cap P_R = \varnothing$

v. $I(p_f) = \{t_s\} \vee T_f \subseteq T_{OP}$ and $O(p_f) = \{t_f\}$

vi. $I(t_f) = \{p_f\}$ and $O(t_f) = \{p_0\}$

vii. $M_0(p_0) = 1 \wedge M_0(p_i) = 1 \; \forall p_i \in P_R$, while
$M_0(p_f) = 0 \wedge M_0(p_i) = 0 \; \forall p_i \in P_{OP}$

viii. $\forall M_k \in R(M_0)$;
$M_k(p_0) + M_k(p_f) = 1 \Rightarrow M_k(p_i) = 0$
$\forall p_i \in P_{OP}$

ix. $PN = (P \setminus P_R, T, I, O, M_0)$ with $M_0(p_0) = 1$ and $M_0(p_i) = 0 \;\; \forall p_i \in P \setminus P_R \cup \{p_0\}$ is a strongly connected and self-loop free PPN.

In the PPNR introduced by Definition 8, a token in place p_0 represents the instance of waiting to initiate the parallel processing of multiple products while the initiation of a parallel processing is modeled by the firing of transition t_0. Similarly, token in a place p_f represents the completion of processing instances, while the event of a process completion is modeled by the firing of transition t_f. Transition t_f allows re-circulation of token from place p_f to place p_0 in order to model the iterative execution of PPNR to produce the required number of final products.

Property 8: For each $p_i \in P_R$ there exits a minimal P-invariant with only marked place $p_i \in P_R$.

Proof: Since each resource-place $p_i \in P_R$ is added to the PPN in such a way that input and output transitions of each operation-place $p_j \in P_{OP}$ act as its output and input transition respectively, as shown in Fig. 1(b). From Property 4, every individual parallel process is a *state-machine*. Further, $I(t_0) \cap P_R = O(t_0) \cap P_R = I(t_f) \cap P_R = O(t_f) \cap P_R = \varnothing$.

Therefore, allocation of each resource place to each operation place is represented by directed cycle with only marked place $p_i \in P_R$ representing the availability of a resource. Hence allocation of resources in PPN imposes the existence of minimal P-invariant for each $p_i \in P_R$.

Property 8 directly follows that allocation of resources to the operations in PPNR is in conservative way, which implies that resources can neither be created nor destroyed. Moreover, the existence of P-invariant y_i for each resource place $p_i \in P_R$ follows that $P_R \cap \|y_i\| = \{p_i\}$, $P_{OP} \cap \|y_i\| \neq \varnothing$ and $\{p_0\} \cap \|y_i\| = \varnothing$. Property 8 depicts that resources can be iteratively used and released and appropriate for the actual requirement of the manufacturing system.

Property 9: Every minimal P-invariant y_i is a minimal siphon with $M_0(p_i) = 1,\ \forall p_i \in P_R$.

Property 9 follows that for each resource place p_i, there exists a minimal siphon with only marked place p_i.

Theorem 10: Let (N, M_0) be a PPNR with "non-shared" resources, then it is live.

Proof: Since there is not a dead transition in PPN, it is live. Property 9 implies that every resource added to PPN makes an initially marked minimal siphon with the operation place. Because every operation is performed on an independent resource, therefore every operation place along with a resource place in PPNR also makes a trap, which is marked due to Definition 8(vii). Minimal siphons thus obtained by independent resource places contain a marked trap which directly follows the liveness of PPNR due to [23].

Theorem 10 follows that places representing the shared resources in PPNR have the potential of deadlock because each operation utilizes a resource exclusively and releases it after the completion. For the processing flow of a part, the transition between two operations represents the end of first operation and also the beginning of second operation in order. The utilization of a resource is represented by transferring of a token from resource place to operation place through a transition representing the beginning of an operation on that resource. A transition at any marking of PPNR is said to be enabled if its preceding operation place as well as its input resource place are marked by a token.

Theorem 11: An PPNR is free from deadlocks, if and only if $\nexists\ M_k \in R(M_0)$ such that $M_k(p_i) = 0$, for any of $p_i \in I(t_j)$, for each $t_j \in T_{OP}$ representing the beginning of operations at M_k.

Theorem 11 is related to the non-existence of a marking where the execution of operations is blocked due to the unavailability of resources. The input places of the transitions representing the beginning of operations are unmarked often representing the unavailability of resources. The unavailability of required resource is often due to the reason of holding of that resource by another operation. This situation of circular-waiting leads to the existence of a siphon whose resource places are unmarked.

Theorem 12: An PPNR is live if and only if there does not exists a siphon S, such that $\exists M_k \in R(M_0)$, $M_k(p_i) = 0\ \forall p_i \in P_R \cap S$.

Proof: Assume that PPNR is live and that there is siphon S and reachable marking M_k, for which $M_k(p_i) = 0\ \forall p_i \in P_R \cap S$. Properties 5 and 9 lead to the fact that every siphon contains an initial place p_0, a final place p_f or at least one resource place. From Theorem 11, $M_k(p_i) = 0\ \forall p_i \in P_R \cap S$ implies that $M_k(p_i) = 0$ for any of $p_i \in I(t_j)$, for each $t_j \in T_{OP}$ representing the beginning of operations at M_k. Therefore, siphon is deadly marked at M_k and PPNR is not live, by contradiction, there does not exits a siphon S which holds the condition given in the statement of the theorem.

Conversely, for every siphon in PPNR, $\forall M_k \in R(M_0)$: $M_k(p_i) \neq 0\ \forall p_i \in P_R \cap S$. This implies that resource places in siphons remain marked for each reachable marking in PPNR, every siphon would not eventually become empty, which directly follows that every siphon has a marked trap. Hence PPNR is live.

Theorem 13: An PPNR is reversible if and only if it is live.

The statement of Theorem 13 is straightforward, for the characterization point of view for PPNR, liveness is similar to the reversibility.

The main requirement for PPNR is that every parallel processing flow in the system is able to complete, without any deadlock. The statements of the Theorems 10 to 13 depict the conditions under which this requirement is accomplished.

V. Application Example

This section illustrates the PN model of parallel manufacturing processes with shared resources in the form of PPNR.

The manufacturing system given in this example produces the final product from three primitive parts by using four machining centers, two assembly stations A_1 and A_2, two robots R_1 and R_2 and a buffer B. Each machining centre MC_i contains a machine M_i, $i = 1, 2, 3, 4$. When either A_1 or A_2 is ready to execute the assembly task, it requests both robots R_1 and R_2 and acquires them if they are available. When A_1 (A_2) completes the assembly task, it releases both the robots.

It is assumed that input parts are always available to be fixture and that the finished product to be removed. Further, once the system is executed, it can not be interrupted and

system can not begin a new iteration before termination.

The production plan is given as follows:

1) Part 1 is machined by M_1 and part 2 is machined by M_2. Each part automatically fixtures to the pallet and loaded into a machine.
2) After processing, parts 1 and 2 enter the assembly station A_1 for producing part S.
3) Part 3 is machined first by M_3 and then by M_4. In M_3, part 3 automatically fixtures to the pallet and loaded into a machine. After processing, the robot R_1 unloads the intermediate part from M_3 into the buffer B and M_3 is released.
4) From the buffer B, intermediate part is automatically loaded into M_4 and processed. When M_4 finishes processing a part, the robot R_1 unloads a product, a part T, and M_4 is released.
5) The assembly station A_2 assembles parts S and T to produce the final product.

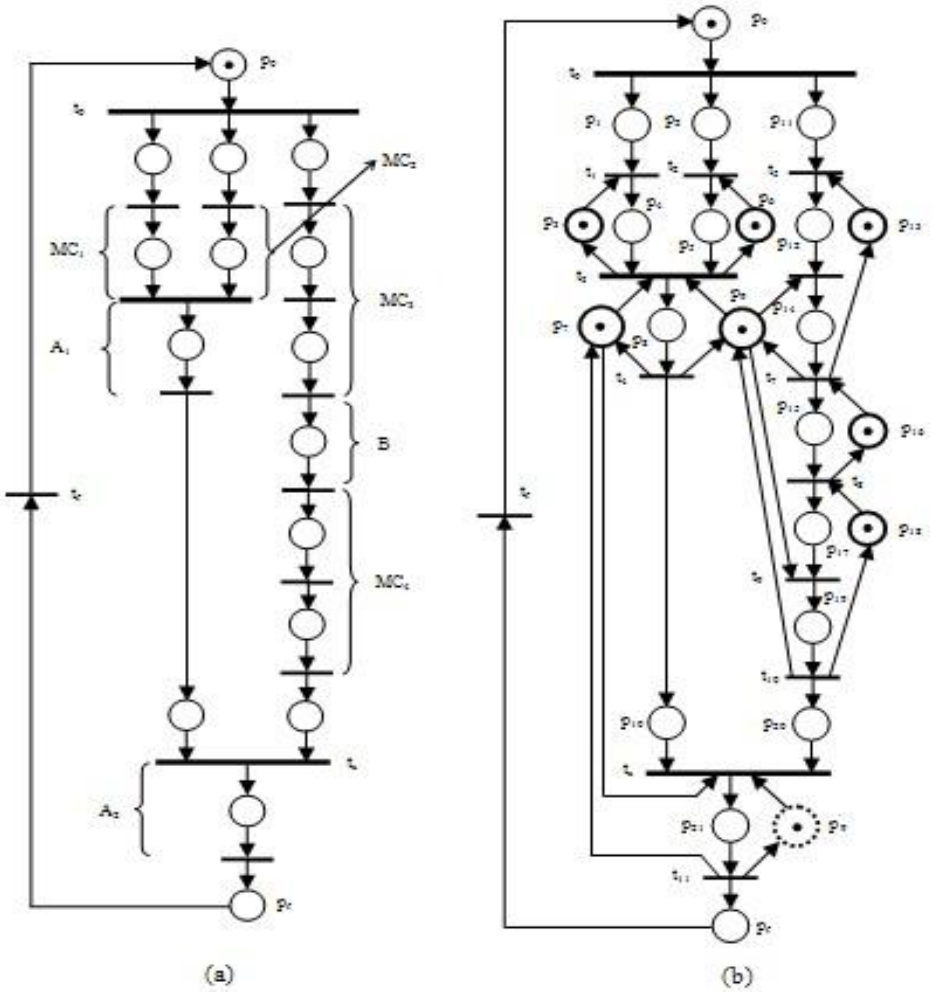

Fig. 2 (a) PPN and (b) PPNR of given manufacturing system

Modeling PPNR for given manufacturing system:

According to the explanation given in Section 3, firstly the PPN of production plan given above is constructed to indicate the process flow of each part type. Fig. 2(a) clearly depicts the parallel processing flows in manufacturing system as well as resource requirement for operations to be performed.

The PPNR is constructed in second step by adding the resource places denoting the availability of resources and resources shared by several processes. Fig. 2(b) shows the resultant PPNR and its transitions denote starting or finishing of the operations. The places of PPNR in Fig. 2(b) represent the operations performed in the manufacturing system and resource types which are explained in Table 1. The shared robots R_1 and R_2 are represented by places drawn in Fig. 2(b) by larger circles while the robot R_1 is drawn twice and represented by the places p_9 and p_9' for clear presentation. In fact, both the places p_9 and p_9' are representation of single place.

From Property 9, every resource place has a marked minimal siphon. Further, For PPNR given in Fig. 2(b), every siphon contains a marked trap and would never become empty. Theorem 12 directly follows that given PPNR is live, which implies its reversibility.

VI. Conclusion

A new class of nets, PPNR, is presented to model the parallel processing of multiple parts on shared resources. The proposed PPNR has the capability to model the real-life manufacturing systems having complex resource sharing. It has been presented that the more complex production plans of manufacturing systems implicated in resource-sharing can be modeled by PPNR.

This paper further presents a number of characterizations of live and reversible PPNR based on siphons and minimal marked siphons. Simple structure based conditions and properties for conservative PPNR are presented for checking the liveness and reversibility. These characteristics help to identify the requirements of the structure of PPNR and its behavior. Furthermore, simple structural based conditions and properties, presented in this paper, are found practical for the synthesis of live and reversible PPNR.

Table 1: Interpretation of places of Fig. 2(b)

Operation places	Interpretation	Resource places	Interpretation
p_0	Initial place of PPNR	p_3	Machine M_1 available
p_1	Part 1 is available	p_6	Machine M_2 available
p_2	Part 2 is available	p_7	Robot R_1 available
p_4	Part 1 machined by M_1	p_9	Robot R_2 available
p_5	Part 2 machined by M_2	p_{13}	Machine M_3 available
p_8	Parts 1&2 assembled on A_1	p_{16}	Buffer B available
p_{10}	Part S is available	p_{18}	Machine M_4 available
p_{11}	Part 3 is available		
p_{12}	Part 3 machined by M_3		
p_{14}	Intermediate part unloaded by R_1		
p_{15}	Intermediate part in buffer B		
p_{17}	Intermediate part machined by M_4		
p_{19}	Intermediate part unloaded by R_1		
p_{20}	Part T is available		
p_f	Final place of PPNR		

Acknowledgment

First author (F. A.) is grateful to the Higher Education Commission of Pakistan and The Punjab Education Department, Pakistan for providing him a scholarship for pursuing doctoral studies at Harbin Institute of Technology Shenzhen Graduate School, Shenzhen (China).

References

[1] M. Kamath, and N. Viswanadham, "Application of Petri net based models in the modeling and analysis of flexible manufacturing systems," Proceeding of IEEE International Conference on Robotics and Automation, vol. 3, 1986, pp. 312-317.

[2] J. Ezpeleta, , and J. Martinez, "Synthesis of live models for a class of FMS," Proceeding of IEEE International Conference on Robotics and Automation, vol. 3, 1993, pp. 557-563.

[3] M. C. Zhou, F. DiCesare, and A. A. Desrochers, "A hybrid methodology for synthesis of Petri net models for manufacturing systems," IEEE Trans. on Robotics and Automation, 8(3), 1992, pp. 350-361.

[4] M. C. Zhou, K. McDermott and P. A. Patel, "Petri net synthesis and analysis of a flexible manufacturing system cell," IEEE Trans. on Systems, Man, and Cybernetics, 23(2), 1993, pp. 523-531.

[5] T. H. Sun, C. W. Cheng and L. C. Fu, "A Petri net based approach to modeling and scheduling for an FMS and a case Study," IEEE Trans. on Industrial Electronics, 41(6), 1994, pp. 593-601.

[6] M. C. Zhou, and K. Venkatesh, Modeling, Simulation and Control of FMS, a Petri net Approach. 1999, Word Scientific Publishing Co.

[7] H. Huang, Enhancing the property preserving Petri net process algebra for component-based system design (with application to designing multi-agent systems and manufacturing systems). 2004, PhD Thesis, City University of Hong Kong.

[8] K. H. Lee, J. Favrel and P. Baptiste, "Generalized Petri net reduction method," IEEE Trans. on Systems, Man, and Cybernetics, 17(2), 1987, pp. 297-303.

[9] M. C. Zhou, and F. DiCesare, "Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems with shared resources," IEEE Trans. on Robotics and Automation, vol. 7, 1991, pp. 515-527.

[10] N. Hamerrlian, "Refinement of open protocols for modeling and analysis of complex interactions in multi-agent systems," Lecture Notes in Artificial Intelligence, vol. 2691, 2003, pp. 423-434.

[11] Y. Souissi, "On liveness preservation by composition of nets via a set of places," LNCS, Springer-Verlag, vol. 524, 1990, pp. 277-295.

[12] J. Ezpeleta, M. J. Colom and J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," IEEE Trans. on Robotics and Automation, vol. 11, 1995, pp. 173-184.

[13] H. Huang, L. Jiao and T. Y. Cheung, "Property-preserving composition of augmented marked graphs that share common resources," Proceeding of IEEE International Conference on Robotics and Automation, vol. 1, 2003, pp. 1446-1451.

[14] F. Chu, and X. L. Xie, "Deadlock analysis of Petri nets using siphons and mathematical programming," IEEE Trans. on Robotics and Automation, 13(6), 1997, pp. 793-804.

[15] Z. Li, and M. C. Zhou, "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems" IEEE Trans. on Systems, Man, and Cybernetics-Part A: Systems and Humans, 34(1), 2004, pp. 38-51.

[16] Z. Li, and N. Wei, "Deadlock control of flexible manufacturing systems via invariant-controlled elementary siphons of Petri nets," Int. J Adv. Manuf. Technol, Springer-Verlag, vol. 33, 2007, pp. 24-35.

[17] M. Jeng, X. Xie, and M. Y. Peng, "Process nets with resources for manufacturing modeling and their analysis," IEEE Trans. on Robotics and Automation, 18(6), 2002, pp. 875-889.

[18] Z. Li, and M. C. Zhou, "On siphon computation for deadlock control in a class of Petri nets," EEE Trans. on Systems, Man, and Cybernetics-Part A: Systems and Humans, 38(3), 2008, pp. 667-679.

[19] Z. Li, M. C. Zhou and M. Jeng, "A maximally permissive deadlock prevention policy for FMS based on Petri net siphon control and the theory of regions," IEEE Trans. on Automation Science and Engineering, 5(1), 2008, pp. 182-188.

[20] E. R. Boer, and T. Murata, "Generating basis siphons and traps of Petri nets using the sign incidence matrix," IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications, 41(4), 1994, pp. 266-271.

[21] R. Cordone, L. Ferrarini and L. Piroddi, "Some results on the computation of minimal siphons in Petri nets," Proceedings of the 42nd IEEE Conference on Decision and Control, 2003, pp. 3754-3759.

[22] A. Valmari, "The state explosion problem," Lectures on Petri Nets I: Basic Models, LNCS vol. 1491, 1998, pp. 429-528.

[23] T. Murata, "Petri nets: properties, analysis and application," In Proceedings of IEEE, 77(4), 1989, pp. 541-580.

[24] J. L. Peterson, Petri Net Theory and the Modeling of Systems, 1981, Prentice-Hall: Englewood Cliffs, NJ.

Farooq Ahmad received the M. Sc. degree from University of The Punjab and M. Phil. degree from GC University Lahore, Pakistan in Statistics in 1994 and 2001 respectively, and now he is PhD scholar in the Computer Science Department of Harbin Institute of Technology Shenzhen Graduate School, China. His research interests include the formal methods for system design, specifications and verification of the distributed and manufacturing systems using Petri nets.

He is also a Lecturer in Statistics in the Punjab Education Department, Pakistan.

Hejiao Huang received the B. S. and M. S. degrees in Applied Mathematics form Shaan Xi Normal University in 1997 and 1999 respectively. Further, she received PhD degree in Computer Science from City University of Hong Kong in 2004. Her research interests include Petri net theory and applications, graph theory and application, optical and wireless mesh networks, machine learning and formal methods for system design.

Dr. Huang is currently an Associate Professor in the Department of Computer Science of Harbin Institute of Technology Shenzhen Graduate School, China.

Xiao-long Wang received the B.E. degree in Computer Science from Harbin Institute of Technology, China, and the M.E. degree in Computer Architecture from Tianjin University, China, in 1982, and 1984, respectively, and the Ph.D. in computer science and engineering from Harbin Institute of Technology, China, in 1989, where he became an Associate Professor in 1990. He was a senior research fellow at the Polytechnic University from 1998 to 2000, a Professor of computer science at Harbin Institute of Technology, China. His research interests include artificial intelligence, machine learning, computational linguistics and Chinese information processing.