Computers and Mathematics with Applications 64 (2012) 1–10

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

Module-based architecture for a periodic job-shop scheduling problem

Farooq Ahmad *, Sher Afzal Khan

Faculty of Information Technology, University of Central Punjab Lahore 54000, Pakistan

ARTICLE INFO

Article history:
Received 31 October 2011
Received in revised form 5 February 2012
Accepted 12 February 2012

Keywords:
Petri net
Periodic scheduling
Asynchronous synthesis
Net-modules

ABSTRACT

This paper addresses the Petri net (PN) based design and modeling approach for a periodic job-shop scheduling problem. Asynchronous synthesis for net-modules of the jobs is suggested in this paper for optimal allocation of shared resources to different operations. To make sure the completion of all the jobs in a single iteration of a production cycle and the correct calculation of a makespan, the synchronization problem among jobs is tackled by introducing the special synchronizing transition in the model. A timed-place PN is adopted for the purpose of finding the feasible schedule in terms of the firing sequence of the transitions of the PN model by using the heuristic search method. Further, the characterization of the PN model is performed and it is shown that the PN model for a periodic job-shop scheduling problem is equivalent to a class of PN known as parallel process net with resources (PPNRs). The modeling approach is demonstrated with a practical example and a makespan is calculated for the example.

1. Introduction

Scheduling is a fundamental problem in the control and an important parameter of the performance evaluation of any type of a system, e.g., distributed computer system, database system or manufacturing system [1]. Scheduling is the problem of optimal allocation of the system resources to its set of processes and selection among alternative plans [2]. Moreover, the problem of optimal allocation of resources leads to the optimal modeling of the scheduling problem.

Job-shop scheduling has received this large amount of attention due to the reason that it falls into the class of NP-complete problems because the computation time required to obtain the global optimal schedule grows exponentially with the problem size, and it is among the most difficult to formulate and solve [3]. Job-shop scheduling can be thought of as a fixed operation route on shared resources for each job while the route is not necessarily the same for a predetermined collection of jobs. Therefore, this problem again leads toward efficient allocation of limited resources over a specified time to perform a collection of jobs and can be thought of as the problem of job-shop system modeling. Moreover, handling of shared-resources [4] becomes an important aspect during the design phase and allocation of resources is always a challenging task in the modeling of a scheduling system [5]. Therefore, parallel or concurrent processing of different part types on a limited number of resources such as machines, robots, buffers etc. is a common situation in job-shop scheduling and non optimal allocation of resources can lead to a deadlock situation [6–8].

This paper focuses on the design and modeling strategy of an iterative version of the job-shop scheduling problem which is known as the periodic job-shop scheduling problem where the batch of size n of each job is processed iteratively. The modeling perspective of the periodic job-shop scheduling problem is important, from a practical point of view, though its research results are less enough than the non-periodic one.

Petri net (PN) has been recognized as a flexible modeling technique because it is well suited for modeling the multifarious constraints in the scheduling problems [9–14]. Due to its graphical representation, the PN model can be developed and

* Corresponding author. Tel.: +92 3324655020.
E-mail addresses: dr.farooq@ucp.edu.pk, farooq190@gmail.com (F. Ahmad), sherafzal@ucp.edu.pk (S.A. Khan).

0898-1221/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2012.02.031

enhanced from the logical sequencing of the system. Further, PN modeling incorporates concurrency [15], non-determinism, timing information and resource-sharing and capture structural interactions to model deadlocks, conflicts, buffer-sizes and precedence relations [16,17]. In addition, PNs have the salient feature to perform qualitative and quantitative analysis of the modeled system due to their underlying mathematical foundation [18].

Nevertheless, the literature on the PN based modeling and design for the periodic job-shop scheduling problem is not extensive. A PN model for production scheduling is either constructed by a top-down or bottom-up approach [19,20]. In a top-down approach [21–23], first the high-level description of a system is presented and then the model is stepwise refined until a complete net model is achieved, whereas in a bottom-up approach [19,24–26] the net modules of specified subsystems are constructed and finally they are combined by merging common places, transitions or subnets.

For the PN model of the non-periodic scheduling problem, the initial marking is distinct from the final marking and the optimal path from initial to final marking leads toward its solution while for the periodic one the initial marking is same as the final marking [27]. In the conventional PN based modeling approaches for periodic scheduling [27–30], each job has its own cycle and the processing of ith job on a resource can be started before the $(i-1)$th job's processing at all the resources has been completed. The resources in such PN models are not bound to wait till the completion of the last job processing, which lead to the following problems.

1. The model may direct to miss the completion of the cycle of the job and keep on cycling before it falls down to its final state.
2. The model may lead to the final state without completing all the jobs in the whole production cycle. In this way, the makespan of a single iteration of the job-shop instance may not be correct.

In order to alleviate these issues about periodic scheduling, this paper introduces the asynchronous synthesis of net-modules of the jobs which are constructed by the following steps. First, the process-flow model of each job consisting of the chain of operations is constructed. Thereafter, net-modules are developed consisting of marked resource places and the transitions of the process-flow model. These net-modules are merged to obtain the PN model of non-periodic scheduling by using the concept of asynchronous synthesis. Further, to handle the synchronization problem [31] among various jobs to be completed for the purpose of completing the single iteration of the production cycle, a special place and two transitions are added to the model for non-periodic schedule. This step leads toward the final model for the job-shop periodic scheduling in the form of parallel process net with resources (PPNRs) [4,32]. In this way, the PN model for the iterative version of the job-shop scheduling problem guarantees the correct makespan.

A timed-place PN [9,28,33] is adopted in this paper for the purpose of finding the feasible schedule in terms of the firing sequence of the transitions of the PN model by using the heuristic search method [9,34]. In the timed-place PN model, time is only associated with places and all transitions are immediate. A timed-place net is obtained by assigning the time delays to the operation places by keeping the resource places, initial and final places for each individual job immediate. The special place p_0 is also immediate while tokens in the immediate places are readily available.

The paper is organized as follows. Section 2 describes the basic terminology of PN and the concepts to be used in the rest of the paper. The job-shop system's specifications and the assumptions are discussed in Section 3. Section 4 explains the PN based modeling procedure by constructing the net-modules of the jobs for periodic job-shop scheduling. Characterization of the proposed PN model is performed in Section 5. A real life application example to demonstrate the design and modeling strategy introduced in this paper is presented in Section 6 and some conclusive remarks are given in Section 7.

2. Definitions and concepts

In this section, some basic definitions and notations of PN are described. Further, some important concepts needed for the rest of the paper are also discussed in this section.

Definition 1 (*[35,36], Petri Net*)**.** A *Petri net* (N, M_0) is a net $N = (P, T, F, W)$ with an initial marking M_0, where, P is a finite set of *places* where $|P| = m$; T is a finite set of *transitions* where $|T| = n$, such that $P \cap T = \varnothing$ and $P \cup T \neq \varnothing$; $F \subseteq (P \times T) \cup (T \times P)$ is the *flow relation*; W is a *weight function* such that $W : F \to N^+$, $W(x, y) \in N^+$ if $(x, y) \in F$ and zero otherwise, M_0 is a function $M_0 : P \to N$ such that $M_0(p)$ represents the number of *tokens* in place $p \in P$.

A PN with its entire arc weights as 1's is known as an ordinary PN [36] and its net structure is written as $N = (P, T, F)$.

Definition 2 (*Pre-Set and Post-Set*)**.** For $x \in P \cup T$, $\text{pre}(x) = \{y \mid (y, x) \in F\}$ and $\text{post}(x) = \{y \mid (x, y) \in F\}$ are called the *pre-set* (*input set*) and *post-set* (*output set*) of x, respectively. For a set $X \subseteq P \cup T$, $\text{pre}(X) = \cup x \in X \text{pre}(x)$ and $\text{post}(X) = \cup x \in X \text{post}(x)$.

Definition 3 (*Path and Closed Path*)**.** For a Petri net $(N, M_0) = (P, T, F, M_0)$, a *path* $\sigma = x_1 x_2 \ldots x_n$ is a sequence of places and transitions such that $(x_i, x_{i+1}) \in F$, $1 = i = n$; $\sigma = x_1 x_2 \ldots x_n$ is said to be *elementary* if, $\forall x_i, x_j \in \sigma, i \neq j \Rightarrow x_i \neq x_j$. A directed path is said to be closed if $(x_{i+1}, x_i) \in F$. Further, closed is also called a directed cycle.

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 $\forall t_j \in T$, $\text{pre}(t_j) \cap \text{post}(t_j) = \varnothing$

i.e. no place can be both an input and an output of the same transition. There is a *structural conflict* in N if and only if $\exists p_i \in P : |\text{post}(p_i)| \geq 2$, or for the set of transitions $T' \subseteq T$ there is a common input place $p_i \in P$.

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 $M_k[t_j\rangle M'_k$ for some $t_j \in T$, then $M'_k \in R(M_0)$.

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 $\forall p_i \in \text{pre}(t_j) : M(p_i) \geq 1$.

Definition 4 (*[36] Place-Invariant and Minimal Place-Invariant*)**.** For a PN (N, M_0), a place-invariant is an m-vector $y \geq 0$ such that $B^T \cdot y = 0$, where B is an $m \times n$ incidence matrix. It is represented by the non-zero entries of vector $y \geq 0$, which is called the support of place-invariant. Place-invariant is said to be *minimal* if no proper subset of the support is also a support.

Definition 5 (*[37,38] Place Fusion*)**.** Consider two Petri nets $(N_1, M_{10}) = (P_1, T_1, F_1, M_{10})$ and $(N_2, M_{20}) = (P_2, T_2, F_2, M_{20})$, where $P_1 \cap P_2 = R \neq \varnothing$. Let $(N, M_0) = (P, T, F, M_0)$ be composed from (N_1, M_{10}) and (N_2, M_{20}) by operation *place fusion*. Then the elements in (N, M_0) are defined as follows: $P = P_1 \cup P_2$, $T = T_1 \cup T_2$, $F = F_1 \cup F_2$, and

$$M_0(p) = \begin{cases} M_{10}(p) & p \in P_1 \\ M_{20}(p) & p \in P_2 \\ \max\{M_{10}(p), M_{20}(p)\} & p \in R. \end{cases}$$

Definition 6 (*[32] Parallel Process Net with Resources*)**.** The parallel process net with resources (PPNRs) is a PN which is equivalent to $PN = (P, T, F, M_0)$ 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. $\text{pre}(p_0) = \{t_f\}$ and $\text{post}(p_0) = \{t_0\}$
iv. $\text{pre}(p_f) = \{t_s\} \vee \{t_j\}$, for any $t_j \subseteq T_{OP}$ and $\text{post}(p_f) = \{t_f\}$
v. $\text{pre}(t_f) = \{p_f\}$ and $\text{post}(t_f) = \{p_0\}$
vi. $|\text{post}(t_0)|$ is the number of raw parts entering into the system
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. $PN = (P \setminus P_R, T, F, M_0)$ 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, self-loop free, live and reversible parallel process net.

Definition 7 (*[33] Place-Timed Petri Net*)**.** A *place-timed Petri Net* (N, M_0) is a Petri net with a time parameter. That is $N = (P, T, F, \tau)$, where P, T, F, and M_0 are the same as given in Definition 1, τ is a time function $\tau : P \to N$ (set of positive integers) associated to places such that for $p \in P$, $\tau(p)$ represents the time duration for a token to be available in p.

3. Job-shop system specifications and assumptions

Job-shop scheduling problem.

The job-shop scheduling problem is described as follows.

A finite set of n jobs: $J = \{J_1, J_2, \ldots, J_n\}$.

Job J_i consists of a set of operations $\{o_{i1}, o_{i2}, \ldots, o_{ik}\}$ for $1 \leq i \leq n$.

A finite set of m resources: $R = \{R_1, R_2, \ldots, R_m\}$.

An *operation* represents the process being performed on a resource. Therefore, resource requirements are considered in an *operation*. Each *job* in a system is composed of a chain of operations and the generic sequence of processes remains the same.

For the purpose of the specification of a system for the job-shop scheduling problem, the job-shop scheduling system (JSS) is defined as:

$$\text{JSS} = (R, O, \Psi, T)$$

where,

R: a set of resources $R = \{R_1, R_2, \ldots, R_m\}$;
O: a set of all operations for different jobs to be completed in JSS by the set of resources;
$\Psi : R \to 2^{|O|}$, a resource allocation function of the set of resources into the subsets of operations;
$T : R \times O \to N^+$, a mapping from a particular resource to be used for a particular operation into the number of time units taken by the operation.

Each job is determined by its own sequence of operations, i.e., job J_i for $1 \leq i \leq n$ consists one of the $2^{|O|}$ subsets of operations. In addition, an operation can alternatively be performed on a different set of resources and the time of the

operation performed on the set of resources may not be the same. Therefore, this argument directly follows the flexibility of JSS which implies $O_i \cap O_j \neq \varnothing$ for some $i \neq j$, where $O_i, O_j \subseteq O$. Moreover, $\Psi(R_i) = O_i$ explains the set of operations O_i to be performed on the resource $R_i,\ i = 1, 2, \ldots, m$. The prescribed time for a certain operation on a resource can be written as $T(R_i, o_{ij}) = \tau_j,\ i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots\ldots, |Oi|$.

The following assumptions are to be considered for JSS.

Each resource can handle at most one operation at a time.

Each operation depends only on the preceding operation and the next operation can start after the completion of the preceding one.

Each operation is non-preemptive.

Each operation needs to be processed during an uninterrupted period of a given length on one or some given machines: $\{\tau_{i_1}, \tau_{i_2}, \ldots, \tau i_k\}$.

The resources in JSS may be workers, who are responsible for operating the machines, robots or other equipments.

The purpose of JSS is to find an optimal schedule, that is, a reasonable allocation of the operations to time intervals and resources such that the makespan is minimized.

4. Petri net modeling for periodic scheduling

In this section, the specifications of JSS discussed above are transformed into the Petri net model by mapping the operations and resources to places while the start and end of the operations to transitions.

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

Definition 8 (*Asynchronous Synthesis Net*)**.** Let $(N_1, M_1), (N_2, M_2) \ldots$ and (N_n, M_n) be n net modules with $N_i = (P_i \cup R, T_i, F_i, W_i)$, $1 \leq i \leq n$, satisfying: $P_i \cap R = \varnothing$ for $1 \leq i \leq n$ and $P_i \cap P_j = \varnothing$, $T_i \cap T_j = \varnothing$, $F_i \cap F_j = \varnothing \forall i, j \in N^+\ i \neq j$. (N, M) with $N = (P, T, F, W)$ is said to be an asynchronous synthesis net resulting from the merge of n nets if and only if

1. $P = \bigcup_{i=1}^{n} P_i$
2. $T = \bigcup_{i=1}^{n} T_i$
3. $F = \bigcup_{i=1}^{n} F_i$
4. $W(p, t) = W_i(p, t)$ if $(p, t) \in F_i$ and $W(t, p) = W_i(t, p)$ if $(p, t) \in F_i$ for $i = 1, 2, \ldots, n$
5. $M(p) = M_i(p)$ if $p \in P_i$ for $i = 1, 2, \ldots, n$.

The resources in the conventional PN models for periodic scheduling are not bound to wait till the completion of the last job processing, which lead to the following problems. First, the model may direct to miss the completion of the cycle of the job and keep on cycling before it falls down to its final state. In addition, the model may lead to the final state without completing all the jobs in the whole production cycle. Both the problems convey the lake of synchronization in completing all the jobs. To tackle these issues about periodic scheduling the following steps for modeling are introduced.

- The process-flow model of a job J_i consisting of the chain of operations is constructed by the set of places $\{p_{i0}, p_{i1}, \ldots, p_{ik}\}$ called operation places. Each transition in the PN operation flow model represents either the start or the end of the operation. The places p_{i0} and p_{ik} are called the initial and final places for job J_i and there is no common operation place within the operation flow models of the jobs. The process flow model of job J_i in the job shop scheduling problem is modeled as a state-machine [36] with the unique path $p_{i0}t_{i1}p_{i1}t_{i2} \ldots p_{ij}t_{ij} \ldots p_{ik}t_{ik}p_{ik}1 \leq i \leq n$ with an initial marking $M_{i0}(p_{i0}) = 1$.
- Thereafter, sub-nets consisting of marked resource places and the transitions of the process flow model, denoting the availability of resources, are composed into the process flow model of jobs; demonstration of this step is shown in Fig. 1a. The composition is performed by using the transition fusion operator [31] in the synchronic synthesis manner [39]. In addition, these sub-nets act as an *external agent* to perform different operations on required resources. Now, the module for an individual job J_i in the job shop scheduling problem is defined as $N_i = (P_i \cup R, T_i, F_i, W_i)$. A resource place $r_i \in R$ in the composed module has input and output transitions such that $(r_i, t_{ij}), (t_{ij+1}, r_i) \in F_i$ where t_{ij} represents the start of the operation(s) and t_{ij+1} represents the end of the operation(s) to be performed on resource r_i. Initial marking of the net-module for J_i is $M_{i0}(p_{i0}) = 1, M_{i0}(r_i) = 1\ \forall r_i \in R$ and zero otherwise.
- Third for the non-periodic scheduling, the obtained net-modules for jobs are merged through asynchronous synthesis with the implementation of a place merging operator described in Definition 5 on the shared resource places. This is known as module-based architecture in a bottom-up fashion.
- Finally, to get the PN model of the periodic job-shop scheduling, a transition t_0, with $\text{pre}(t_0) = \{p_0\}$ and $\text{post}(t_0) = \{p_{i0}\},\ i = 1, \ldots, n$, is added which simply initiates the parallel execution of the jobs. Further to synchronize the completion of all the jobs in a single production cycle and to generate the iterative scheduling, the synchronizing transition t_s is added such that $(p_{ik}, t_s) \in F$ where $\text{pre}(t_s) = \{p_{ik}\};\ i = 1, 2, \ldots, n$ and $\text{post}(t_s) = \{p_0\}$.

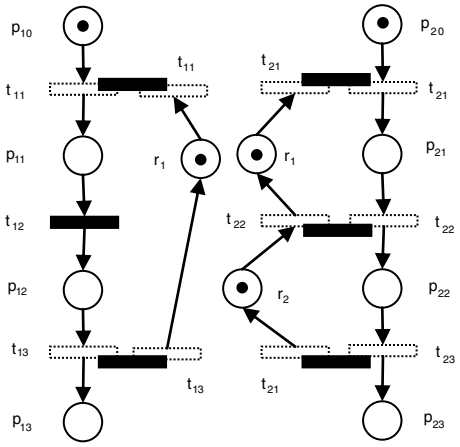

Fig. 1a. Demonstration of the fusion of sub-nets to the process flow model in synchronous synthesis manner.

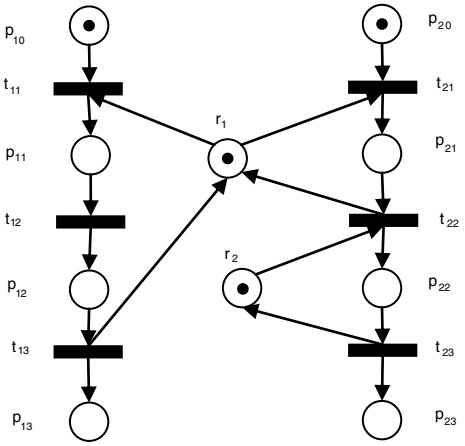

Fig. 1b. Asynchronous synthesis net by merging the net-modules of the jobs.

Table 1
Resource allocation to the operations of two jobs.

	Resources	
	Operation 1	Operation 2
Job 1	r_1	r_1
Job 2	r_1	r_2

To demonstrate the PN based modeling procedure for the periodic job-shop scheduling problem, an example is presented.

Example. Two jobs sharing two resources, each job contains two operations. Both the operations of job 1 are processed on resource r_1 and operations 1 and 2 of job 2 are processed on resources r_1 and r_2, respectively (see Table 1).

The resources are allocated to the process flow model of each job to form its net-module as shown in Fig. 1a. Next, Fig. 1b portraits the non-periodic version of the job-shop scheduling problem in the form of asynchronous synthesis net. Finally, the periodic version of the job-shop scheduling problem described above is shown in Fig. 1c.

Timed-place net for scheduling.

For the purpose of scheduling, time is only associated with places and all transitions are immediate. A timed-place net is obtained by assigning the time delays to the operation places by keeping the resource places immediate as a token in a resource place is readily available. Zero time is assigned to a special place p_0 as it represents the initialization and termination of a schedule. Immediate transitions follow that a transition representing the end of an operation is the same transition representing the beginning of the next operation. Because time delays are associated with places, therefore a token at an operation place becomes available after a delay corresponding to the duration time of the operation. Moreover, the sequence

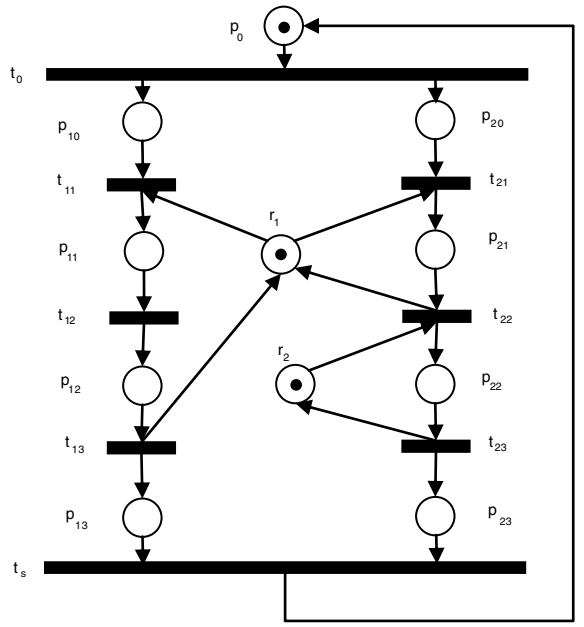

Fig. 1c. The PN model of the iterative version of the job-shop scheduling problem in the example.

of transitions firing represents the production path in a PN model of production plan and there may be several paths to achieve a final product in a same plan.

5. Characterization of the Petri net model for periodic scheduling

Property 1. *Every net-module for an individual job is executable.*

Proof. Each net-module without resource places is a process-flow model consisting of a chain of operations in the form of a state-machine. Such a type of state-machine is executable in sequence because for each job J_i; $M_{i0}(p_{i0}) = 1$. Since marked resource places are added to the transitions in such a way that they become output and/or input places of transitions. Further, resources are used exclusively and follow the "hold while waiting" property. Hence every individual net-module is sequentially executable. □

Property 2. *Final marking is reachable for every net-module of an individual job.*

Proof. Property 2 is a consequence of Property 1. □

Property 3. *For each $r \in R$, in the net-module of an individual job, there exists a minimal place-invariant with the marked place $r \in R$.*

Proof. Each resource place $r \in R$ is added to the process-flow model for job J_i by synchronous synthesis in such a way that it becomes the input place for the transition t_{ij} representing the start of operation(s) and the output place for the transition t_{ik} representing the end of operation(s) to be performed on $r \in R$ such that $(r, t_{ij}), (t_{ik}, r) \in F_i$. In this way, the directed closed path from $r \in R$ back to $r \in R$ is obtained. Therefore, allocation of each resource place to the process flow model is represented by the directed cycle with only marked place $r \in R$. Hence allocation of resources imposes the existence of minimal place-invariant for each $r \in R$. □

Property 4. *The process flow model for each job in the PN model of the periodic job-shop scheduling problem makes a cycle containing the initial place p_0, transition t_0 and a synchronizing transition t_s.*

Proof. The proof for the statement of Property 4 is trivial, as by removing the initial place p_0, synchronizing transition t_s and their associated arcs, non-iterative version of the model is left. Then by removing the resource places and their associated arcs there are acyclic process-flow models for the jobs. □

Property 5. *The PN model of the periodic job-shop scheduling problem is equivalent to a class of PN, PPNRs.*

Table 2
The periodic job-shop scheduling problem with resource requirements.

Jobs	Operations			
	O1	O2	O3	O4
J1	M1:8	Buffer slot	M3:4	–
J2	M2:5	M3:3	M1:7	Buffer slot
J3	M2:6	M1:5	Buffer slot	M3:2

Table 3
Interpretation of operation and resource places of the model in Fig. 2b.

Places for J1	Interpretation	Resource place	Interpretation
P10	J1 available	M1	M1 available
P11	M1 processing J1	M2	M2 available
P12	J1 part in buffer slot	M3	M3 available
P13	M3 processing J1	BS1	J1's buffer slot available
P14	J1 completed	BS2	J2's buffer slot available
Places for J2	Interpretation	BS3	J3's buffer slot available
P20	J2 available		
P21	M2 processing J2		
P22	M3 processing J2		
P23	M1 processing J2		
P24	J2 part in buffer slot		
P25	J2 completed		
Places for J3	Interpretation		
P30	J2 available		
P31	M2 processing J3		
P32	M1 processing J3		
P33	J2 part in buffer slot		
P34	M3 processing J3		
P35	J3 completed		

Proof. By applying the place merging operator to the modules of individual jobs in an asynchronous synthesis manner, the PN model of a non-periodic version of the JSS problem is obtained. Then by adding the special place p_0, a transition t_0 and t_s to the asynchronous synthesis net, the model for the periodic version of the JSS problem is obtained. □

Now suppose a transition t_f and a place p_f is added such that $\text{pre}(p_f) = \{t_s\}$, $\text{post}(p_f) = \{t_f\}\text{pre}(t_f) = \{p_f\}$ and $\text{post}(t_f) = \{p_0\}$. This step satisfies all the conditions of the Definition 6 of PPNRs. Further, by applying the fusion of serial transitions operator [36] to t_s, p_f, t_f, the transition t_f is merged into the transition t_s. This directly leads to the model for the periodic job-shop scheduling problem which preserves all the properties of PPNRs.

6. Application example

This section explains how to apply the asynchronous synthesis on different modules of the jobs to model the Job-shop periodic scheduling problem in order to find the optimal schedule. Suppose the system consists of 3 machines of different kinds to process 3 jobs, J1, J2 and J3. The job J1 has two operations while jobs J2 and J3 have 3 operations which are processed on different machines in the sequence order. Further, each job requires a buffer slot after completing the operation on machine M1. The details of the operation distribution and the processing time units of each operation on different machines are shown in Table 2.

The module for each job is constructed as follows, where place $p_{i,j}$ represents the operation $O_{i,j}$, i.e., jth operation of ith job. The interpretation of the places of the model in Fig. 2b is given in Table 3.

- Net module for J1: first operation of job J1 is performed on machine M1 following the buffer slot BS1 and second operation is performed on machine M3. To construct the module for job J1, marked resource places M1, BS1 and M3 are added to the process-flow model $p_{1,0}, t_{1,1}, p_{1,1}, t_{1,2}, p_{1,2}, t_{1,3}, p_{1,3}, t_{1,4}, p_{1,5}$ (i.e., state machine for J1) such that $(M_1, t_{1,1})$, $(t_{1,2}, M_1)$, $(BS_1, t_{1,2})$, $(t_{1,3}, BS_1)$, $(M_3, t_{1,4}) \in F_1$.
- Net module for J2: all three operations of job J2 are processed on respective machines M2, M3 and M1 following the buffer slot. Therefore marked places M2, M3, M1 and BS2 are added to the path $p_{2,0}, t_{2,1}, p_{2,1}, t_{2,2}, p_{2,2}, t_{2,3}, p_{2,3}, t_{2,4}, p_{2,4}, t_{2,5}, p_{2,5}$ representing the process-flow model for job J2 to construct the module for job J2 such that $(M_2, t_{2,1})$, $(t_{2,2}, M_2)$, $(M_3, t_{2,2})$, $(t_{2,3}, M_3)$, $(M_1, t_{2,3})$, $(t_{2,4}, M_1)$, $(BS_2, t_{2,4})$, $(t_{2,5}, BS_2) \in F_2$.
- Net module for J3: the process-flow model for job J3 is represented by the state machine $p_{3,0}, t_{3,1}, p_{3,1}, t_{3,2}, p_{3,2}, t_{3,3}, p_{3,3}, t_{3,4}, p_{3,4}, t_{3,5}, p_{3,5}$. The operations of job J3 go through the machine M2, M1 following the buffer slot BS3 and M3 in sequential manner. To construct the module of job J3, marked places M2, M1, BS3 and M3 are added to the path such that $(M_2, t_{3,1})$, $(t_{3,2}, M_2)$, $(M_1, t_{3,2})$, $(t_{3,3}, M_1)$, $(BS_3, t_{3,3})$, $(t_{3,4}, BS_3)$, $(M_3, t_{3,4})$, $(t_{3,5}, M_3) \in F_3$.

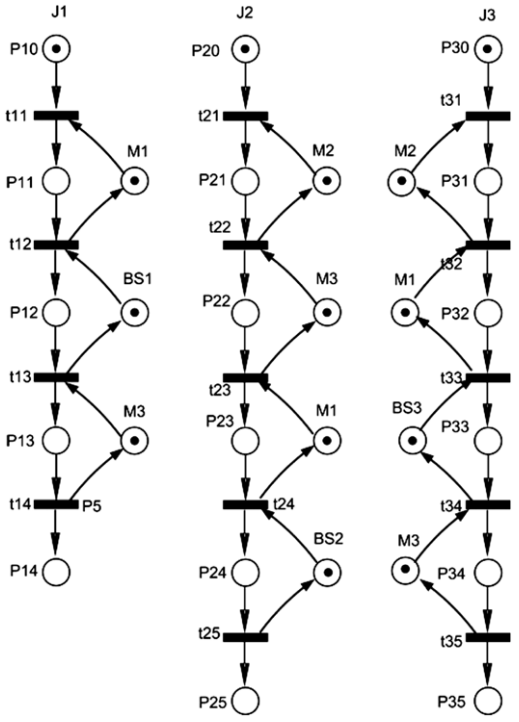

Fig. 2a. Net-modules for three jobs.

Now the net-modules for all the jobs are merged through asynchronous synthesis described in Definition 8 by merging the common resource places in the modules to obtain the model for non-periodic schedule. Finally, the PN model for the periodic job-shop schedule is obtained by adding the place p_0, transition t_0 and synchronizing transition t_s to the model for the non-periodic schedule. The modules of the three jobs are shown in Fig. 2a and the final model for the job-shop periodic scheduling is shown in Fig. 2b.

Consider that all the jobs having the same batch size are ready for processing at time zero. The optimal production schedule is found from start to end of one cycle because the production path is cyclically repeated as many times as the batch size of the jobs.

Next, reachability graph [36] is generated for the PN model in Fig. 2b to know whether our modeled system is deadlock free and to seek a production schedule to minimize the time required to complete the jobs, i.e., the makespan. The firing sequences, i.e., possible paths from initial node back to it in the reachability graph represent the individual operation sequences because initial and final markings are same for periodic scheduling. The optimal schedule can be searched among different operation sequences in the reachability graph by using the heuristic search [9,34]. The optimal firing sequence representing the operation sequence of the schedule is $\sigma = t_0t_{11}t_{31}t_{12}t_{32}t_{33}t_{21}t_{13}t_{14}t_{34}t_{35}t_{22}t_{23}t_{24}t_{25}t_s$ and the schedule for single iteration of the system given in the form of Gantt chart is shown in Fig. 3 with makespan of 25 time units. Moreover, for a single iteration, if the system needs one unit of time as a setup time then batch size of n of all three jobs have a makespan equal to $25n + n$.

7. Conclusions

The Petri net (PN) based design and modeling approach for the iterative version of the job-shop scheduling problem has been addressed, which has more practical importance then the non-iterative one.

For this purpose, net-modules for each have been constructed by composition of sub-nets containing the marked resource places to the process-flow models of the jobs. Then, asynchronous synthesis net from net-modules for the jobs has been constructed for optimal allocation of shared resources to different operations of the jobs. For the purpose of the iteration of the same schedule a special place and two special transitions have been added to the non-periodic version of the PN model. Further, to make sure the completion of all the jobs in a single iteration of the production cycle and the correct calculation of a makespan, the synchronization problem among jobs has been tackled by these special synchronizing transitions in the model. The timed-place PN has been adopted for the purpose of finding the feasible schedule in terms of the firing sequence of the transitions in the reachability graph of the PN model by using the heuristic search method.

Moreover, the characterization of the PN model has been performed and it is shown that the PN model for the periodic job-shop scheduling problem is equivalent to a class of PN know as parallel process net with resources (PPNRs). It has been

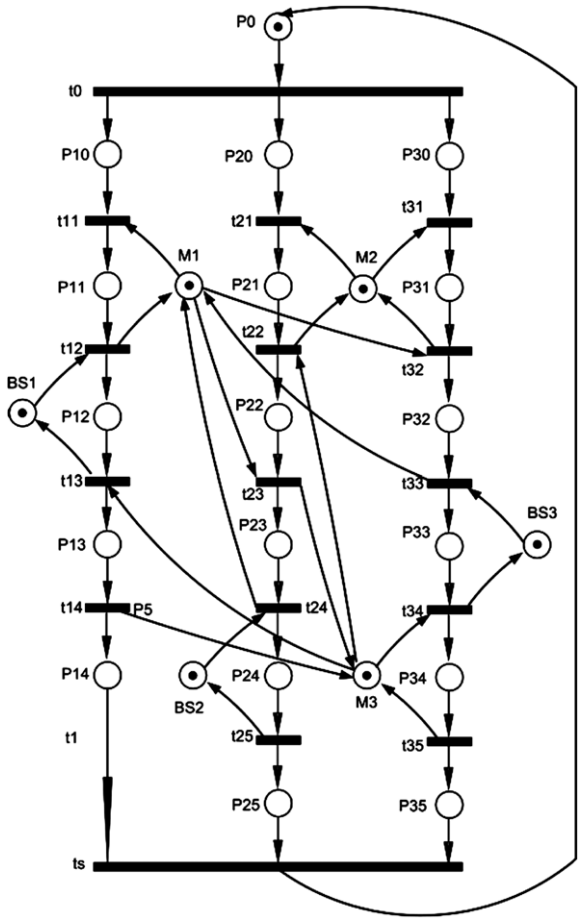

Fig. 2b. The final model for the periodic job-shop scheduling problem.

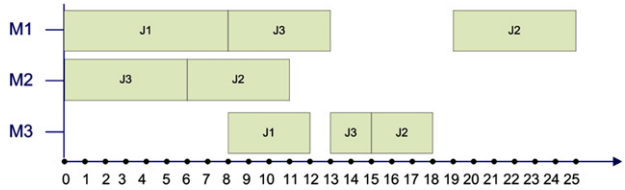

Fig. 3. Solution of the scheduling problem in the form of Gantt chart.

concluded that by using the module-based architecture, one cannot only design a correct system model which iterates properly, but also find the optimal scheduling in a formal way. The modeling approach has also been demonstrated and the makespan has been calculated for a practical example given in the paper. For the purpose of verification, the proposed PN model for the periodic job-shop scheduling problem will be analyzed in the future work for deadlock detection, its recovery, boundedness, reversibility and reachability problems etc.

References

[1] B. Hurz, M.C. Zhou, Modeling and Control of Discrete-Event Dynamic Systems: with Petri Nets and other Tools, Springer-Verlag London Limited, 2007.
[2] A. Rudek, R. Rudek, A note on optimization in deteriorating systems using scheduling problems with the ageing effect and resource allocation models, Computers & Mathematics with Applications 62 (2011) 1870–1878.
[3] G. Tuncel, G.M. Bayhan, Applications of Petri nets in production scheduling: a review, International Journal of Advanced Manufacturing Technology 34 (2007) 762–773.
[4] F. Ahmad, H. Huang, X. Wang, Analysis of the Petri net model of parallel manufacturing processes with shared resources, Information Sciences 181 (2011) 5249–5266.
[5] L. Jiao, H. Huang, T.Y. Cheung, Handling resource sharing problem using property-preserving place fusions of Petri nets, Journal of Circuits, Systems and Computers 17 (2008) 365–387.

[6] Z.W. Li, M.C. Zhou, On siphon computation for deadlock control in a class of Petri nets, IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans 38 (2008) 667–679.
[7] J. Li, H. Huang, F. Ahmad, Deadlock avoidance of a kind of JSP with multi-resources sharing, in: Proceedings of the 11th Joint Conference on Information Sciences, JCIS-2008 Proceedings, Atlantis Press, 2008, doi:10.2991/jcis.2008.95.
[8] Z.W. Li, N. Wei, Deadlock control of flexible manufacturing systems via invariant-controlled elementary siphons of Petri nets, International Journal of Advanced Manufacturing Technology 33 (2007) 24–35.
[9] T.H. Sun, C.W. Cheng, L.C. Fu, Petri net based approach to modeling and scheduling for an FMS and a case study, IEEE Transactions on Industrial Electronics 41 (1994) 593–601.
[10] W.M. Zuberek, W. Kubiak, Timed Petri net in modeling and analysis of simple schedules for manufacturing cells, Computers & Mathematics with Applications 37 (1999) 191–206.
[11] M.C. Zhou, K. Venkatesh, Modeling, simulation, and control of flexible manufacturing systems: a Petri net approach, in: Intelligent Control and Intelligent Automation, World Scientific, Singapore, 1998.
[12] W. Zhang, T. Freiheit, H. Yang, Dynamic scheduling in flexible assembly system based on timed Petri nets model, Robotics and Computer-Integrated Manufacturing 21 (2005) 550–558.
[13] A. Christian, R. Francois, A Petri net model and a general method for on and off-line multi-resource shop floor scheduling with setup times, International Journal of Production Economics 74 (2001) 63–74.
[14] G.M. Bayhan, G. Tuncel, Modelling, behavioral analysis and performance evaluation of an automative assembly plant using Petri nets, International Journal of Industrial Engineering 9 (2002) 238–247.
[15] J.L.G. Guirao, F.L. Pelayo, J.C. Valverde, Modeling the dynamics of concurrent computing systems, Computers & Mathematics with Applications 61 (2011) 1402–1406.
[16] M.C. Zhou, H.S. Chiu, H.H. Xiong, PN Scheduling of FMS using branch and bound method, in: International Conference on Industrial Electronics, Control, and Instrumentation, 21st IEEE IECON, 1995, pp. 211–216.
[17] K. Venkatesh, M.C. Zhou, Object-oriented design of FMS control software based on object modeling techniques diagrams and Petri nets, Journal of Manufacturing Systems 17 (1998) 118–136.
[18] R. Zurawski, M.C. Zhou, Petri nets and industrial applications: a tutorial, IEEE Transactions on Industrial Electronics 41 (1994) 567–582.
[19] H. Huang, Component-based design for job-shop scheduling systems, International Journal of Advanced Manufacturing Technology 45 (2009) 958–967.
[20] J.M. Proth, X. Xie, Petri Nets: A Tool for Design and Management of Manufacturing Systems, John Wiley & Sons, Inc., 1997.
[21] N. Hamerrlian, Refinement of open protocols for modeling and analysis of complex interactions in multi-agent systems, in: Lecture Notes in Artificial Intelligence, vol. 2691, Springer-Verlag, 2003, pp. 423–434.
[22] M.C. Zhou, F. DiCesare, Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems with shared resources, IEEE Transactions on Robotics and Automation 7 (1991) 515–527.
[23] T. Kis, D. Kiritsis, P. Xirouchakis, K.P. Neuendorf, Petri net model for integrated process and job shop production planning, Journal of Intelligent Manufacturing 11 (2000) 191–207.
[24] J. Ezpeleta, M.J. Colom, J. Martinez, A Petri net based deadlock prevention policy for flexible manufacturing systems, IEEE Transactions on Robotics and Automation 11 (1995) 173–184.
[25] H. Huang, L. Jiao, T.Y. Cheung, Property-preserving composition of augmented marked graphs that share common resources, in: Proceeding of IEEE International Conference on Robotics and Automation, vol. 1, 2003, pp. 1446–1451.
[26] I.B. Abdallah, H.A. E1Maraghy, Deadlock detection and avoidance in FMS: a PN approach, International Journal of Advanced Manufacturing Technology 14 (1998) 704–715.
[27] D.Y. Lee, F. DiCesare, Petri nets and heuristic search for periodic scheduling, in: Proc. IEEE Int Systems, Man and Cybernetics Conf, IEEE SMC'92, 1992, pp. 998–1003.
[28] D.Y. Lee, F. DiCesare, FMS scheduling using Petri nets and heuristic search, IEEE Transactions on Robotics and Automation 10 (1994) 123–132.
[29] H. Tohme, M. Nakamura, E. Hachiman, K. Onaga, Evolutionary Petri net approach to periodic job-shop-scheduling, in: Proc. IEEE Int Systems, Man, and Cybernetics Conf. IEEE SMC '99, 1999, pp. 441–446.
[30] G. Bucci, A. Fedeli, L. Sassoli, E. Vicario, Timed state space analysis of real-time preemptive systems, IEEE Transactions on Software Engineering 30 (2004) 97–111.
[31] L. Jiao, T.Y. Cheung, W. Lu, Handling synchronization problem in petri net-based system design by property-preserving transition-reduction, The Computer Journal 48 (2005) 692–701.
[32] F. Ahmad, H. Huang, X.-L. Wang, Petri net modeling and deadlock analysis of parallel manufacturing processes with shared resources, Journal of Systems and Software 83 (2010) 675–688.
[33] M.D. Jeng, C.S. Lin, Petri nets for the formulation of aperiodic scheduling problems in FMSs, in: 6th International Conference on Emerging Technologies and Factory Automation Proceedings, ETFA'97, 1997, pp. 375–380.
[34] Y.L. He, G.N. Wang, Petri nets based deadlock-free scheduling for flexible manufacturing systems, in: 9th International Conference on Control, Automation, Robotics and Vision, ICARCV'06, 2006, pp. 1–5.
[35] W. Reisig, Petri Nets: An Introduction, in: EATCS Monographs on Theoretical Computer Science, vol. 4, Springer-Verlag, Berlin, Heidelberg, 1985.
[36] T. Murata, Petri nets: properties, analysis and application, Proceedings of IEEE 77 (1989) 541–580.
[37] L. Jiao, H. Huang, T.Y. Cheung, Property-preserving composition by place merging, Journal of Circuits, Systems and Computers 14 (2005) 793–812.
[38] L. Jiao, H. Huang, T.Y. Cheung, Handling resource sharing problem using property-preserving place fusions of Petri nets', Journal of Circuits, Systems and Computers 17 (2008) 365–387.
[39] Z.W. Li, M.C. Zhou, Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach, Springer-Verlag London Limited, 2009.