We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Chapter 1

Automated Petri-Net Modelling for Batch Production Scheduling

Dejan Gradišar and Gašper Mušič

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48467

1. Introduction

Production scheduling is a fundamental function in production control. It has an immediate and considerable impact on the efficiency of related manufacturing processes and significantly influences the overall production performance.

The primary characteristic of batch production is that the output of the process appears in quantities of materials or lots called batches. All components are completed at a workstation before they move to the next one. These kinds of production environments appear in chemical, pharmaceutical, food and similar industries.

The control of batch processes poses difficult issues as these processes are neither continuous nor discrete, but have the characteristics of both. ISA society introduced a multi-part S88 standard where the first part [1] defines the models and terminology for batch plants and control systems. S88 provides a framework for the development of technologies that not only support control activities of batch processes but also management activities, such as scheduling. This is illustrated in [14] where a generic framework is defined for interpreting a multi-purpose/product batch plant in terms of S88 constructs for scheduling purposes.

In order to cope with the behaviour of a batch production process an appropriate mathematical model is needed. When the behaviour is described by such a model, formal methods can be used, which usually improve the understanding of systems, allow their analysis and help in implementation. MILP based formulations of batch process features are typically used as shown in [12]. Nevertheless, Petri nets have also been applied in different aspects of modelling, qualitative and quantitative analysis, control, planning and scheduling of batch processes [4]. Independently of the chosen framework, the modelled behaviour is often extremely complex. Within the changing production environment the effectiveness of batch production modelling is, therefore, a prerequisite for the effective design and operation of batch systems.

To determine a model, data from different existing information systems could be used. From production management systems, such as Manufacturing Resource Planning (MRP II) and Enterprise Resource Planning (ERP), data about the needs, product structure and process

structure could be gained [21]. On the other hand, data from the production process could be used to determine the actual resource availability. MRP II and ERP systems are commonly used in discrete manufacturing for upper level production control, such as production planning. Standard production management tools, such as MRP, are also used in batch production environment [20]. As defined with standard S88.01 the required raw materials and their quantities are determined from a dedicated data structure named *Formula*. This way the formula can be linked to standard Bill of Materials (BOM) used within MRP II concept [7].

While in a discrete manufacturing processes BOM are used to determine process materials and their quantities needed for production of finished products, in batch production processes these data are given with Formula. The same is with information that defines a sequence of operations required to produce an item. In discrete manufacturing processes these are defined with Routing, and Manufacturing recipes are used in batch production.

These two groups of data items, together with the given resource units, form the basic elements of the production process. These data can be effectively used to build up a model of the batch production system with timed Petri nets. An algorithm will be introduced, which builds a Petri-net model from the existing data. The model is built directly in a top-down manner, starting from the Formula (BOM) and the Manufacturing recipes (routings) [21].

First a class of Petri nets used is presented in a formal manner with detailed discussion on time representation. Next a method to describe a Formula with Petri net structure is given. Root item, representing the product, is composed of sub-items (sub-processes). Later a method of describing the basic production activities with timed Petri net is presented. The obtained model is applied in optimisation of batch scheduling problem.

2. Timed PN

In the Petri net literature, three basic ways of representing time in Petri nets are used [2]: firing durations (FD), holding durations (HD) and enabling durations (ED). When using FD principle the transition firing has duration [23]. In contrast, when using HD principle, a firing has no duration but a created token is considered unavailable for the time assigned to transition that created the token, which has the same effect. With ED principle, the firing of the transitions has no duration while the time delays are represented by forcing transitions that are enabled to stay so for a specified period of time before they can fire. This is a more general concept since it allows for modelling of task interruption. Some authors use an even more general concept, which assigns delays to individual arcs, either inputs or outputs of a transition [10].

When modelling several performance optimisation problems, e.g. scheduling problems, such a general framework is not needed. It is natural to use HD when modelling most scheduling processes as operations are considered non-preemptive. The timed version of CPNs defined by [9] uses a HD equivalent principle, where the unavailability of the tokens is defined implicitly through the corresponding time stamps. While CPNs allow the assignment of delays both to transition and to output arcs, we further simplify this by allowing time delay inscriptions to transitions only. This is sufficient for the type of examples investigated here, and can be generalised if necessary.

To include a time attribute of the marking tokens, which determines their availability and unavailability, the notation of timed CPN will be adopted. Tokens are accompanied with a timestamp, which is written next to the token number and separated from the number by @. E.g., two tokens with time stamp 10 are denoted 2@10. A collection of tokens with different

time stamps is defined as a multiset, and written as a sum (union) of sets of timestamped tokens. E.g., two tokens with time stamp 10 and three tokens with timestamp 12 are written as $2@10 + 3@12$. The timestamp of a token defines the time from which the token is available.

Time stamps are elements of a time set TS, which is defined as a set of numeric values. In many software implementations the time values are integer, i.e., $TS = \mathbb{N}$, but will be here admitted to take any positive real value including 0, i.e., $TS = \mathbb{R}_0^+$. Timed markings are represented as collections of time stamps and are multisets over $TS : TS_{MS}$. By using HD principle the formal representation of a P/T timed Petri net is defined as follows. $TPN = (\mathcal{N}, M_0)$ is a timed Petri net system, where: $\mathcal{N} = (P, T, Pre, Post, f)$ is a Timed Petri net structure, $P = \{p_1, p_2, ..., p_k\}, k > 0$ is a finite set of places, $T = \{t_1, t_2, ..., t_l\}, l > 0$ is a finite set of transitions. $Pre : (P \times T) \to \mathbb{N}$ is the input arc function. If there exists an arc with weight k connecting p to t, then $Pre(p,t) = k$, otherwise $Pre(p,t) = 0$. $Post : (P \times T) \to \mathbb{N}$ is the output arc function. If there exists an arc with weight k connecting t to p, then $Post(p,t) = k$, otherwise $Post(p,t) = 0$. $f : T \to TS$ is the function that assigns a non-negative deterministic time delay to every $t \in T$. $M : P \to TS_{MS}$ is the timed marking, M_0 is the initial marking of a timed Petri net.

To determine the availability and unavailability of tokens, two functions on the set of markings are defined. The set of markings is denoted by $\mathbb{M}$. Given a marking and model time, $m : P \times \mathbb{M} \times TS \to \mathbb{N}$ defines the number of available tokens, and $n : P \times \mathbb{M} \times TS \to \mathbb{N}$ the number of unavailable tokens for each place of a TPN at a given time τ_k. Note that model time also belongs to time set TS, $\tau_k \in TS$.

Using the above definitions, addition and subtraction of timed markings, and the TPN firing rule can be defined. Given a marked $TPN = (\mathcal{N}, M)$, a transition t is time enabled at time τ_k, denoted $M[t\rangle_{\tau_k}$ iff $m(p, M, \tau_k) \geq Pre(p,t)$, $\forall p \in \bullet t$. An enabled transition can fire, and as a result removes tokens from input places and creates tokens in output places. The newly created tokens are accompanied by timestamps depending on the model time and the delay of transition that created the tokens. If marking M_2 is reached from M_1 by firing t at time τ_k, this is denoted by $M_1[t\rangle_{\tau_k} M_2$. The set of markings of TPN $\mathcal{N}$ reachable from M is denoted by $R(\mathcal{N}, M)$.

3. Modelling procedure

Petri nets are a family of tools that provide a framework, which can be used for various problems that appear during the life-cycle of a production system [18]. In this section we present the modelling of production system using timed Petri nets for the purpose of performance control. When timed Petri nets are used, it is possible to derive performance measures such as makespan, throughput, production rates, and other temporal quantities. The Petri net model is built based on the data stored in production management information systems, i.e., ERP system.

3.1. The class of production systems

With the method presented here several scheduling problems that appear in various production systems can be solved. In a discrete manufacturing different jobs are needed to produce a final product that is composed of several components. Similarly in batch production different activities have to be performed in order to produce a final product. However, here the resultant product is produced with some irreversible change, e.g. products are mixed from quantities of ingredients.

Different management systems (*ERP*) can be applied for different types of production systems to plan the production process activities. We are assuming here a management system that can provide plan for both, discrete and batch process. The system generates work orders that interfere with the demands for the desired products. Different jobs/procedures are needed to produce a desired product. Set of operations needed to produce one item represent a job. In general, more operations have to be performed using different resources in order to complete a specific job. To complete a specific product, more sub-products may be needed. To list these components a BOM is used in discrete manufacturing and formulas in batch manufacturing. These components determine sub-jobs that are needed to manufacture a parent item. In this way the general scheduling problem is defined that can be applied both in discrete or batch production environment and can be given as:

- n jobs are to be processed: $J = \{J_j\}, j = 1, ..., n$,
- r resources are available: $M = \{M_i\}, i = 1, ..., r$,
- each job J_i is composed of n_j operations: $O_j = \{o_{jk}\}, k = 1, ..., n_j$,
- each operation can be processed on (more) different sets of resources $S_{jkl} \in R$; l determines the number of different sets,
- the processing time of each operation o_{jkl}, using resource set S_{jkl}, is defined with T_{jkl},
- precedence constraints are used to define that some operations within one job has to be performed before a set of operations in another job.

Using this definition, the following assumptions have to be considered:

- Resources are always available and never break down.
- Each resource can process a limited number of operations. This limitation is defined by the capacity of resources.
- Operations are non pre-emptive.
- When an operation is performed, it is desirable to free the resources so that they can become available as soon as possible. Intermediate buffers between processes are common solutions. It is common for batch processes that successive operations need to be performed on the same resource as predecessor. In this case the resource is free when the last operation is finished.
- Processing times are deterministic and known in advance.
- Work orders define the quantity of desired products and the starting times. Orders that are synchronised in time are considered jointly.

3.2. Modelling of production activities

Here we present a method of describing the production-system activities with timed Petri nets using the holding-duration representation of time. The places represent resources and jobs/operations, and the transitions represent decisions or rules for resources assignment/release and for starting/ending jobs.

To make a product, a set of operations has to be performed. We can think of an operation as a set of events and activities. Using a timed PN, events are represented by transitions and activity is associated with the presence of a token in a place.

An elementary operation can be described with one place and two transitions, see Figure 1. When all the input conditions are met (raw material and resources are available) the event that starts the operation occurs, t_1. This transition also determines the processing time of an operation. During that time the created token is unavailable in place p_2 and the operation is being executed. After that time the condition for ending the operation is being satisfied and t_2 can be fired. Place p_1 is not a part of the operation, it determines the input condition, e.g. the availability of the input material.

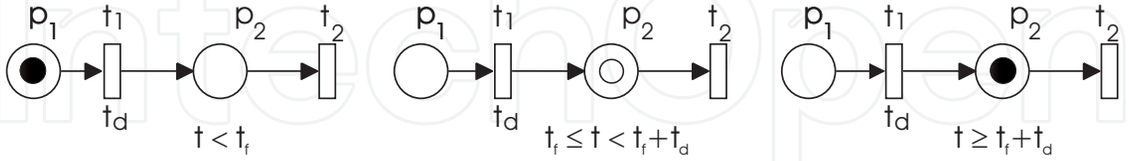

Figure 1. Operation described with timed Petri net.

When parallel activities need to be described the Petri-net structure presented in Figure 2 is used. Transition t_0 sets the input conditions for the parallel execution of two operations. In places p_{01} and p_{02} operations can wait for the available resource(s). The time delays of the transitions t_{11in} and t_{12in} define the duration of each operation. An available token in place p_{11} (p_{12}) indicates that operation is finished. Transition t_1 is used to synchronise both operations.

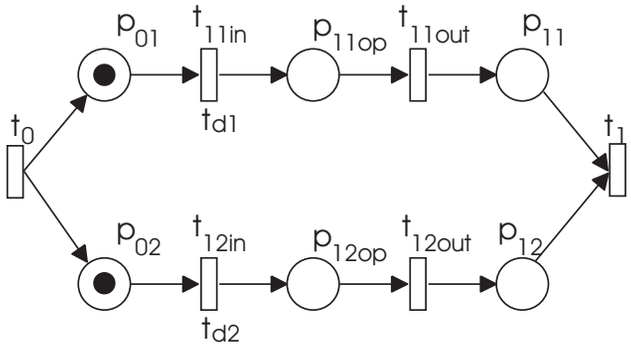

Figure 2. Two parallel operations.

An operation might need resources, usually with a limited capacity, to be executed; this is illustrated in Figure 3. Place p_{R1} is used to model a resource R_1. Its capacity is defined with the initial marking of that place. The resource is available to process the operation if there are enough available tokens in it. When the resource is used at the start of the operation the unavailable token appears in place p_{1op}. After the time defined by transition t_{1in} the token becomes available, t_{1out} is fired, and the resource becomes free to operate on the next job. For this reason zero time needs to be assigned to the transition t_{1out}. An additional place p_1 models the control flow. When the token is present in this place, the next operation can begin.

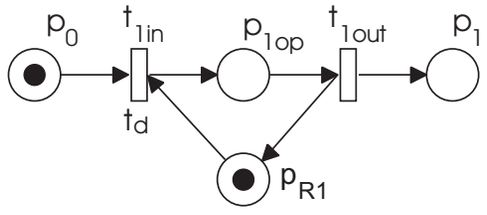

Figure 3. Operation that uses a resource with finite capacity.

A particular operation can often be done on more different (sets of) resources with different availability, and the time duration can be different on each set of resources. An example where

an operation can be executed on two different sets of resources is shown in Figure 4. If the operation chooses resource R_3, its time duration is determined with the transition $f(t_{2in}) = t_{d2}$. Otherwise the set of resources, composed of R_1 and R_2, is being selected and its operation time is defined with $f(t_{1in}) = t_{d1}$.

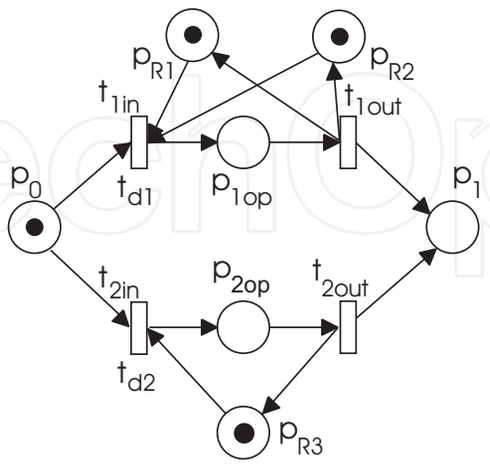

Figure 4. Operation that can be performed on two different sets of resources.

There are common situations where more operations use the same resource, e.g., an automated guided vehicle (AGV) in a manufacturing system or a mixing reactor in a batch system. This can be modelled as shown in Figure 5.

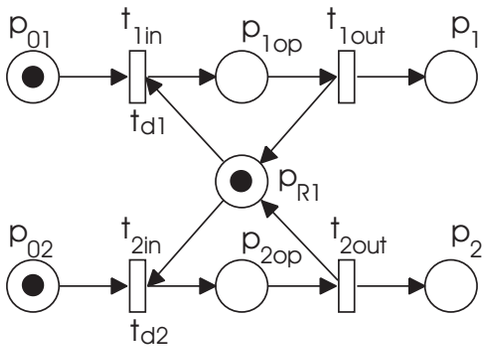

Figure 5. Shared resource.

Precedence constraints are used to define technological limitations and the sequence of operations. An example of two successive operations is shown in Figure 6, depicted as $Op1$ and $Op2$. In this figure an example of technological limitations is also shown. Here, the situation where operation $Op1$ precedes operation $Op3$ is considered. For this purpose an additional place p_{pr1} is inserted between the transition t_{1out} (ending $Op1$) and the transition t_{3in} (starting $Op3$). The weight n of the arc, which connects p_{pr1} to t_{3in}, prescribes how many items need to be produced by the first operation before the second operation can begin.

3.3. Modelling using the data from production-management systems

The most widely used production-management information system in practice are MRP II and ERP. Data stored in those systems can be used to build up a detailed model of the production system with Petri nets. In discrete manufacturing these data are bills of material and routing, while in batch manufacturing master formula and recipe are used to determine the production

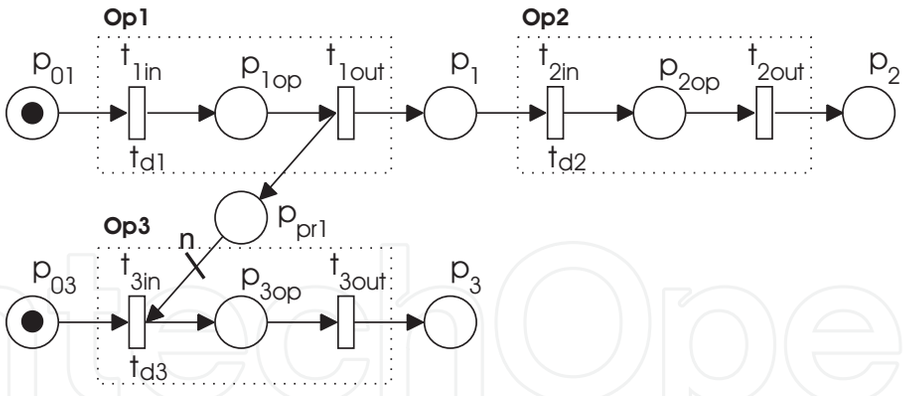

Figure 6. Precedence constraint.

process. Work orders are used to determine which and how many of finished products have to be produced.

3.3.1. Bill of materials (Formula)

The Bill of materials (BOM) is a listing or description of the raw materials and items that make up a product, along with the required quantity of each. In batch manufacturing other terms are used instead, i.e. Formula.

The BOM used in this work is defined as:

$BOM = (R, E, q, pre)$, where:

- $R = \{r_1\}$ is a root item.
- $E = \{e_1, ..., e_i\}$ is a finite set of sub-items,
- $q : E \rightarrow \mathbb{N}$ is the function that defines the quantities for each sub-item e_i. $\mathbf{q}$ represents an $i \times 1$ column vector whose ith entry is $q(e_i)$.
- $pre : (E \times E) \rightarrow \{0, 1\}$ is a precedence-constraints function. It defines the precedence-constraints matrix **pre**, where $\mathbf{pre}(i, j) = 1$ indicates that the i-th item precedes the j-th item. It can also be interpreted as a directed graph.

R is a root item and represents the product that is composed of sub-items described with $e_i \in E$. The number of required sub-items is determined with the vector $\mathbf{q}$. When any sub-item has to be produced before another, the precedence function pre is used to define it. All the sub-items have to be finished before the operation for the subsequent sub-items can begin. A required property of **pre** is that only zero values can be on its diagonal, i.e. a sub-item cannot precede itself. An item is never allowed to become (indirectly) a component of itself. In other words, if the BOM structure is seen as a directed graph, this graph should be cycle-free [21].

If any of the sub-items e_i are composed of any other sub-items, the same BOM definition is used to describe its dependencies. The items at the highest level of this structure represent a finished product, and those at the lower level represent raw materials. The items that represent raw materials do not have a BOM.

Table 1 shows an example of a BOM describing the production of product I, which is composed of three components, i.e., three items of J, one item of K and and two items of

Item	Sub-item	Quantity	Precedence constraints
I	J	3	0 1 0
	K	1	0 0 0
	L	2	0 0 0

Table 1. Example of the BOM structure.

L. From the precedence-constraint matrix it is clear that all of the items J has to be completed before the production of item K can begin.

The mathematical representation of the BOM of item I would be represented as:

$$BOM = (R, E, \mathbf{q}, \mathbf{pre}), \text{ where } R = \{I\},$$

$$E = \{J\ K\ L\}, \quad \mathbf{q} = [3\ 1\ 2] \quad \text{and} \quad \mathbf{pre} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

To start with building a model, let us assume that, for each item from the BOM, only one operation is needed. As stated before, each operation can be represented with one place and two transitions (Figure 1). To be able to prescribe how many of each item is required the transition t_{Rin} and the place p_{Rin} are added in front, and p_{Rout} and t_{Rout} are added behind this operation. The weight of the arcs that connect t_{Rin} with p_{Rin} and p_{Rout} with t_{Rout} are determined by the quantity q_0 of the required items. In this way an item I is represented with a Petri net as defined in Figure 7.

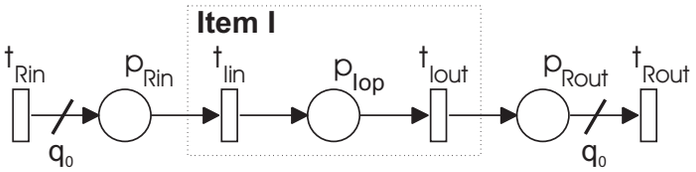

Figure 7. PN structure representing one item in the BOM.

As finished product is defined with a structure of BOMs, the construction of the overall Petri net is an iterative procedure that starts with the root of the BOM and continues until all the items have been considered. If the item requires any more sub-assemblies (i.e., items from a lower level) the operation, the framed area of the PN structure presented in Figure 7, is substituted with lower-level items. If there are more than one sub-items, they are given as parallel activities.

The substitution of an item with sub-items is defined as follows:

- Remove the place p_{Iop} and its input/output arcs.
- Define the PN structure for sub-components, as it is defined with a BOM. Consider the precedence constraints.
- Replace the removed place p_{Iop} by the sub-net defined in the previous step. The input and output transitions are merged with the existing ones.

The result of building the PN model of this example (Table 1) is given in Figure 8, where item I is composed of three subitems: J, K and L.

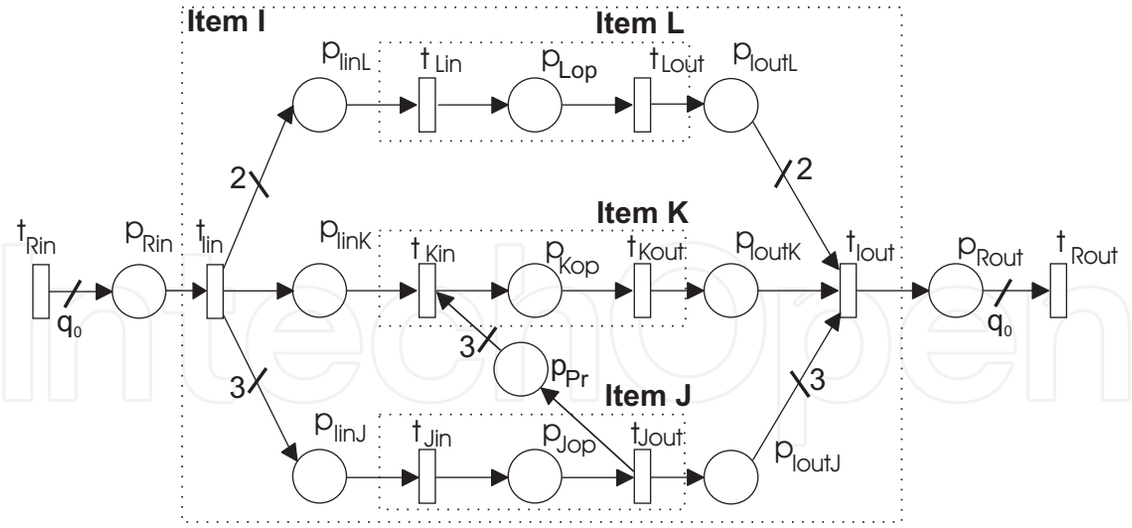

Figure 8. BOM structure defined by PN.

3.3.2. Routings (Recipe)

For each item that can appear in production process, and does not represent a raw material, a routing is defined. It defines a job with sequence of operations, each requiring processing by a particular resource for a certain processing time, which are needed for transforming raw material into the (sub)product. This information are provided by routing tables in discrete manufacturing, and by recipes in batch manufacturing industries. The table contains a header, where the item that is being composed is defined and the lines where all the required operations are described. For each operation one line is used.

As an example, the routing table for item K is given as presented in Table 2. Three operations are needed to produce this item; the first of these operations can be done on two different resources. Similar notation is used for other possible cases, e.g. an operation that needs three resources R_1 and two R_2, or one resource R_1 and three R_3 would be presented by $(3 \times R_1, 2 \times R_2)/(R_1, 3 \times R_3)$.

Operations	Duration	Resources
Op10	10s/9s	R1/R3
Op20	20s	R2
Op30	12s	R1

Table 2. Routing of product K.

The implementation of the routing data in one item of a BOM is defined as follows:

- Remove the place p_{Xop} and its input/output arcs.
- Define a PN structure for the sub-components, as it is defined with routing data. Also precedence constraints are considered here.
- Place p_{Xop} is replaced with the sub-net defined in previous step, where input and output transitions are merged with the existing ones.

Function that defines PN structure for every sub-component yields the corresponding sequence of production operations from the routing table and for each operation a timed Petri net is built as defined in section 3.3.1. All the placed operations are connected as prescribed

by the required technological sequence, and each operation is assigned to the required places representing appropriate resources.

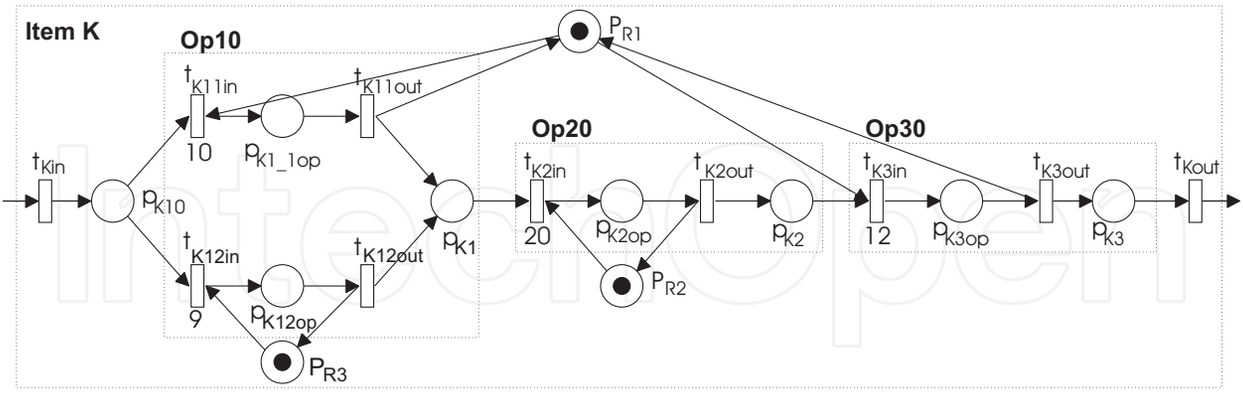

Figure 9. Routing of product *K* modelled with timed Petri net.

The PN structure in Figure 9 is achieved if the sequence of operations described with a routing table (Table 2) is modelled. The resulting PN structure is inserted into the main PN model, on the place where item *K* (p_{Kop}) was.

The routings/recipes are submodels that are inserted (by substitution, as defined previously) into the main model defined with the BOM structure. However, some activities of any sub-item may also be described with a BOM, i.e., in the case they are composed of semi-products. The construction of the overall Petri-net model can be achieved by combining all of the intermediate steps.

3.3.3. Work order

The work order ($WO = [R, q_0, st]$) determines which and how many of the finished products have to be produced. Each product (R) can be represented with a Petri-net model, shown in Figure 7, where one place is added in front and one at the end of the structure to determine the start and end of the work. As usually more products are produced at one time (one product is represented with one batch), the weight of the arc that connects t_{Rin} and p_{Rin} are used to determine the number of required batches (q_0). To be able to consider different starting times for different quantities of one product the general structure shown in Figure 10 is used. q_0 determines the number of products to be finished. Orders for products which should start with the production at the same time are merged and every group of products with the same starting time is modelled with places $p_1, p_2, ...p_n$ and with tokens located in them. The timestamps, which are assigned to these tokens determine the starting time of every group of products. Wights $q_1, q_2, ...q_n$ determine the number of products, where $q_1 + q_2 + ... + q_n$ is equal to q_0. The token in the place p_{end} implies that WO is finished.

3.4. Specifics of batch production processes

In previous sections (3.3.1 – 3.3.3) we present methods to represent formula, recipe and work orders with Petri nets. As given so far these elements can be equally used for discrete and batch process environments. However, as mentioned in 3.1 there are some specifics in batch production processes.

Actually batch production is more complicated and formula and recipe are more connected as are BOM and routings [7]. There are common situations where precedence constraints

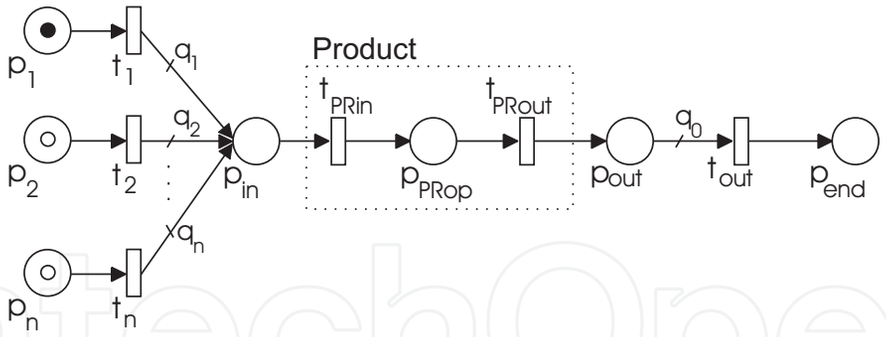

Figure 10. Petri net structure of a work order.

are not used only to define interdependencies between different (finished) items, but also for interdependencies between job operations of various items that are defined with recipes. In this situations the definition of BOM (Formula) has to be extended.

Definition of a sub-items set E is extended in a way that it contain also the information about operations that produce that item: $E = \{(e_1, o_{1k}), ...(e_i, o_{in_j})\}$.

Further also a precedence-constraint function *pre* is extended in order to include information about how many operations are needed for every item. As defined in 3.1, set of operations needed to produce j-th item is given as $O_j = \{o_{jk}\}$, where $k = 1, ..., n_j$. Size of the matrix is therefore defined with $[m \times m]$, where $m = \sum n_j$, $j = 1...n$. Here n is number of items that make up a product. In this way we can define an extended precedence-constraints function *pre* as $pre : ((E \times O) \times (E \times O)) \rightarrow \{0, 1\}$. Element from constraint matrix $pre(p_{ij}, p_{kl}) = 1$ indicates that the j-th operation from i-th item precedes l-th operation from k-th item. Here index p_{ij} in a matrix **pre** presents j-th operation of i-th item and p_{kl} l-th operation of k-th item.

3.5. Procedure for building the PN model

With WO demands for product to be finished are passed. For each WO a Petri net model has to be defined. The modelling procedure can be summarised in the Algorithm 1.

Algorithm 1 Read BOM

```
[R, q, st] = readWO()
For i = 1 to length(R)
    E = readBOM(R(i))
    PN = placePN(R(i), E, q(i), [ ], st(i), x0, y0)
    PN = routing(PN, R(i))
end
```

First, the data about the WO are read. The products that are needed to be produced are given in R; in vector **q** the quantities of the desired products are passed; and vector **st** is used to determine the starting time of each product. For each product the Petri-net structure, shown in Figure 10, is determined and placed on the model. The step when the *routing*() is called is described in more detail with algorithm 2.

First, the routing and the BOM data are read from the database (functions *readRouting()* and *readBOM()*). For each operation that comprises the routing, the algorithm checks whether it is made up of sub-item(s) or this is an operation. In the first case, the function *placePN()* is used to determine the PN structure of the given structure BOM. Precedence constraints are added

Algorithm 2 Read Routing

```
function PN = routing(PN, R)
datRoute = readRouting(R)
[E, q, pre] = readBOM(R)
for i = 1 to length(datRoute.Op)
    if datRoute.Resources ==BOM
      PN1 = placePN(R, E, q, pre, [ ])
      PN = insertPN(PN, PN1)
      for j = 1 to length(E)
        PN1 = routing(PN1, E(j))
      end
    else
      PN = constructPN(PN, datRoute(i))
      PN = insertPN(PN, PN1)
    end
  end
```

if they exist. With the function *insertPN()* the resulting subnet is inserted into the main PN structure. If the operation represents the production operation, the function *constructPN*() is called. With it, basic elements (Figures 1–6) are recognised, joined together and placed in the model, again using the function *insertPN()*. All the data about resources and time durations are acquired from the routing table. The described algorithm has been implemented in Matlab.

The resulting model is stored in a XML-based format employing Petri Net Markup Language (PNML) [6].

3.6. Verification

When the model is built up, it should be verified to see whether it reflects the system operation as defined with data about the production process. Some interesting properties of the model can be checked with the P-invariant analysis. Several P-invariants can be identified in the model. Their number is defined with the sum of all the resources, the number of product routes and the number of all precedences that are present in the model. It can be stated that the weighted sum of tokens that belongs to every P-invariant, which is a consequence of a resource, is equal to the capacity of that resource. The weighted sum of all other invariants is defined with the common number of batches of demanded product.

If possible, the model is later simplified in a way, that eliminated nodes do not influence the model behaviour.

4. Scheduling

Scheduling is one of the most important management functions and is a fundamental problem in the control of any resource-sharing organisation. Scheduling problems are very complex and many have been proven to be NP hard [8].

Literature on deterministic scheduling classifies the manufacturing scheduling problems according to machine environment structure, processing characteristics and constraints, and objectives. Standard machine environment structures lead to standard scheduling problems, e.g., open shop, flow shop and job shop problems, which are commonly studied. All three

problem classes address a problem of sequencing n jobs (tasks) through a set of r machines (resources) where every job has to be processed once on every machine and every such job operation requires a specified processing time. The problems differ in restrictions on the job routings.

The scheduling problems related to batch plants possess a more complicated structure compared to standard scheduling problems. Batch plants are flexible and alternative resources can be used for conveying recipe operations. There may be different operation processing time assignments based on which equipment is used for processing and there may be specific requests on precedences or allowable intermediate storage time. This significantly complicates the problem of operations scheduling in batch processes. A comprehensive review of the state-of-the art of short-term batch scheduling is presented in [12]. Different types of batch scheduling problems are specified and the types of optimization models are reviewed. The presented models result in a formulation of MILP optimization problem and its solution yields an optimal schedule.

Petri nets can be used to effectively model all standard deterministic scheduling problem classes. Furthermore, the modelling power of Petri nets allows for derivation of models also for problems, which do not have standard problem structure but are closer to real process specifics. This is typical e.g. in batch systems where complex interconnections among process equipment are possible and vary considerably with used batch recipes. Even when the models are not as general as the above mentioned MILP problem representations, Petri nets can be used to model main model components of a two stage batch scheduling approach as defined in [12]. In contrast to monolithic approach the two stage approach assumes that the number of batches of each size is known in advance. The scheduling stage therefore concentrates on the allocation of processing resources to the batches while the plant work load is determined in a previous stage.

4.1. Petri net based derivation of optimal or sub-optimal schedules

To derive a feasible schedule, the obtained Petri net model can be simulated by an appropriate simulation algorithm. During the simulation, the occurring conflicts are resolved 'on the fly', e.g. by randomly choosing a transition in conflict that should fire. Instead, heuristic dispatching rules [5], such as Shortest Processing Time (SPT), can be introduced when solving the conflicting situations. The schedule of process operations can be determined by observing the marking evolution of the net. Depending on the given scheduling problem a convenient rule should be chosen. Usually, different rules are needed to improve different predefined production objectives (makespan, throughput, production rates, and other temporal quantities).

A more extensive exploration of the reachability tree is possible by PN-based heuristic search method proposed by [11]. It is based on generating parts of the Petri net reachability tree, where the branches are weighted by the time of the corresponding operations. The chosen transition firing sequence corresponds to a schedule, and by evaluating a number of sequences a (sub)optimal schedule can be determined. The method is further investigated in [22], where a modified heuristic function is proposed and tested on a number of benchmark tests. The problems of the approach are in the complexity of the reachability tree, which can generally not be completely explored. The search has to be limited to predefined maximum tree size in order to complete in a reasonable time. In addition to that, the heuristic function used within the search has to be chosen such that the search is directed more into the depth of the tree,

which makes the obtained solution very sensitive to decisions taken at initial levels of the tree and in many cases the quality of the obtained solutions is rather low.

Recent reports in scheduling literature show an increased interest in the use of meta-heuristics, such as genetic algorithms (GA), simulated annealing (SA), and tabu search (TS). Meta-heuristics have also been combined with Petri net modelling framework to solve complex scheduling problems [19]. With such an approach, the modelling power of Petri nets can be employed, and relatively good solutions of scheduling problems can be found with a reasonable computational effort, although the convergence to the optimum can not be guaranteed. Compared to reachability tree based search methods, meta-heuristics require less memory.

The problem is that these methods require a sort of neghbouring solution generation strategy. This is easily accomplished for well structured problems, e.g. standard scheduling problems, but may be problematic for general models. In contrast, reachability tree methods as well as priority rule based methods can be used with any type of Petri net model. This motivates the investigation of combined methods, such as the combination of dispatching rules and local search [13].

Dispatching rules, however, do not always enable to reach the optimum even if combined with optimization methods. Using the rule based conflict resolution strategy the solution space is explored in a time driven manner where a transition is fired whenever at least one transition is enabled. In contrast, the reachability tree based methods enable to explore the solution space in an event driven manner. It is possible that a chosen firing sequence imposes one or more intervals of idle time between transitions, i.e. some transitions are enabled but do not fire due to waiting for enablement of another transition in accordance to the chosen sequence. The difference is important in cases when the optimal solution can be missed unless some idle time is included in the schedule as shown in [15]. In other words, the optimal solution generally belongs to the class of semi-active schedules [16]. The schedules generated by an event-driven reachability tree search are semi-active schedules.

4.2. Algorithmically generated Petri net models and schedules

In contrast to academic investigation of static scheduling problems the real manufacturing environment is far more dynamic. Planned work orders are often changing, priority orders are inserted, planned resources may become unavailable. A fast derivation of models that adequately represent the current situation in the process is therefore of extreme importance for a usable scheduling system. The above described automatic generation of Petri net models can be effectively used for these purposes.

The proposed algorithm also allows for specific process sequence structures typical for batch processes. E.g., the scheduling literature typically addresses a type of problems where a resource required by an operation is released as soon as the operation is finished. This is typical in the discrete industry, where intermediate products are stored in buffer zones in between machines. In batch processes the situation is different in the sense that a resource, e.g. a reactor is used both for processing and intermediate storage. The resource can be occupied by a number of successive operations, which can be easily modelled by Petri nets. Furthermore, the use of timestamped tokens provides a convenient representation of the time status of the processed work orders.

4.3. Evaluation of schedules

As the majority of commonly used scheduling objective functions are based on completion times of the jobs or work orders and due dates, the timed Petri net modelling framework yields a possibility to use the same kind of model with an objective function tailored to the needs of the particular production scenario.

In the field of deterministic scheduling the objective to be minimised is always a function of the completion times of the jobs or work orders [16]. This fits well in the timed Petri net scheduling framework where the time evolution of the net marking depends on timestamps associated with the tokens. If the schedule is modelled properly, the work order completion times can be read from the timestamps of tokens in the final marking obtained by timed Petri net simulation.

Let o_{ji} denote the i-th operation of work order j. Let C_{ji} denote the completion time of the operation o_{ji}. The completion time of the work order, i.e. the completion of the last operation of the work order is denoted by C_j.

During the timed Petri net marking evolution, start of o_{ji} corresponds to triggering of a related transition t_{ji}. Associated delay $f(t_{ji})$ corresponds to duration of o_{ji}. Following the firing rule, the transition output places are marked with tokens whose time attribute is set to $@(\rho_{ji} + f(t_{ji}))$ if ρ_{ji} denotes the moment of transition firing, i.e., the release time of o_{ji}. The generated timestamp equals the completion time of o_{ji}: $C_{ji} = \rho_{ji} + f(t_{ji})$.

Assuming the timed Petri net model of scheduling problem as described above, let $p_{WO\_j\_end} \subset P$ denote the j-th work order end place, i.e. the place that holds a token representing finished status of the work order. Let M_f denote the final marking reached after all the operations had been finished. If a token in $p_{WO\_j\_end}$ corresponds to finishing the last operation of work order WO_j then $M_f(p_{WO\_j\_end}) = 1@C_j$. Therefore the completion times can be read from $M_f(p_{WO\_j\_end})$: $C_j = M_f(p_{WO\_j\_end}) \subset TS_{MS}$.

4.3.1. Makespan

Makespan C_{max} is equivalent to the completion time of the last finished work order: $C_{max} = max(C_1, \ldots, C_n)$. Considering the above notation

$$C_{max} = max(M_f(p_{WO\_j\_end})), j = 1 \ldots n \tag{1}$$

4.3.2. Total weighted completion time

The sum of weighted completion times gives an indication of the inventory costs related to a schedule [16]. Given a final marking M_f the cost can be calculated as

$$\sum w_j C_j = \sum_{j=1}^{n} w_j M_f(p_{WO\_j\_end}) \tag{2}$$

4.3.3. Tardiness

If a set of due dates d_j is adjoined to the work orders, the tardiness of a work order is defined as the difference $C_j - d_j$ if positive, and 0 otherwise:

$$T_j = max(M_f(p_{WO\_j\_end}) - d_j, 0) \tag{3}$$

In contrast to objective measures, which are related to final marking, the initial marking can be used to specify release dates of work orders. If a token in $p_{WO\_j\_st}$ corresponds to the initial request of work order WO_j, and r_j is a corresponding release date then $M_0(p_{WO\_j\_st})$ should contain a token $1@r_j$.

5. A case study: Multiproduct batch plant

The applicability of our approach will be demonstrated on the model of a multiproduct batch plant designed and built at the Process Control Laboratory of the University of Dortmund. The demonstration plant is relatively simple compared to industrial-scale plants, but poses complex control tasks. A detailed description of the plant can be found in [17].

In the following a brief description of the plant will be given. From the data given in production management systems a timed Petri-net model is built. With the help of a simulator/scheduler using different scheduling rules, different schedules can be achieved. At the end the results for the given problem are presented and compared with the results achieved with other techniques.

5.1. Description of the plant

The process under consideration is a batch process that produces two liquid substances, one blue, one green, from three liquid raw materials. The first is coloured yellow, the second red and the third is colourless. The colourless sodium hydroxide (NaOH) will be referred to below as white. The chemical reaction behind the change of colours is the neutralisation of diluted hydrochloric acid (HCl) with diluted NaOH. The diluted HCl acid is mixed with two different pH indicators to make the acid look yellow if it is mixed with the first one and red when mixed with the second one. During the neutralisation reaction the pH indicators change their colour when the pH value reaches approximately 7. The first indicator changes from yellow to blue, and the second from red to green.

The plant consists of three different layers, see Figure 11. The first layer consists of the buffering tanks B11, B12 and B13, which are used for holding the raw materials "Yellow", "Red" and "White". The middle layer consists of three stirred tank reactors, R21, R22 and R23. Each reactor can be filled from any raw-material buffer tank. The production involves filling the reactor with one batch of "Yellow" or "Red" and then neutralising it with one batch of "White". The lower layer consists of two buffer tanks, B31 and B32, in which the products are collected from the middle layer. Each of them is used exclusively for "Blue" or "Green" and can contain three batches of product. The processing times of the plant are presented in Table 3. The system can be influenced through different inputs, pumps P1-P5 and valves V111-V311.

5.2. The scheduling problem

The plant provides a variety of scheduling problems. In our case we are dealing with a problem, where we have a demand to produce a certain amount of finished products. Also the starting times of every work order are given. Our task is to determine when raw materials must be available and when the overall production process will be finished. Our goal is to finish the production in the shortest time.

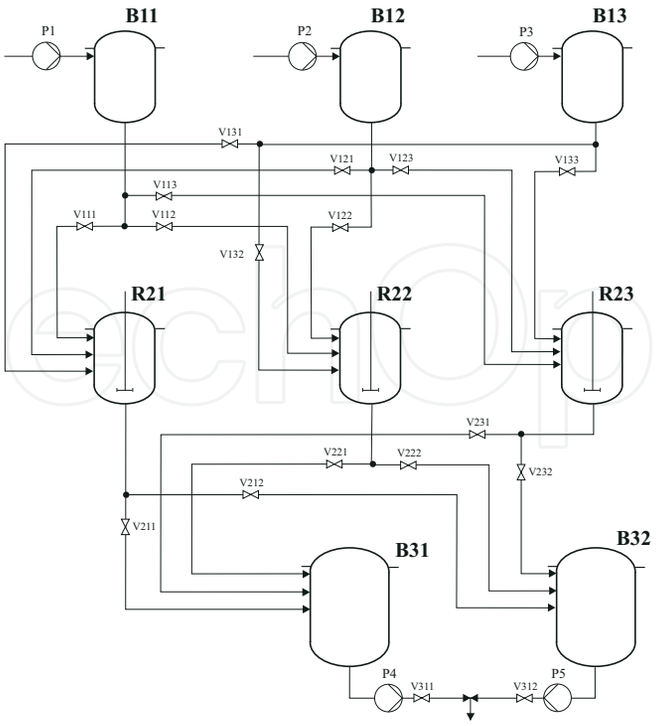

Figure 11. Multiproduct batch plant.

Process	Time(s)
Pumping 1 batch "Yellow" into B11	12
Pumping 1 batch "Red" into B12	12
Pumping 1 batch "White" into B13	12
Draining 1 batch "Yellow" into R21	15
Draining 1 batch "Red" into R21	11
Draining 1 batch "White" into R21	10
Draining 1 batch "Yellow" into R22	12
Draining 1 batch "Red" into R22	13
Draining 1 batch "White" into R22	9
Draining 1 batch "Yellow" into R23	12
Draining 1 batch "Red" into R23	14
Draining 1 batch "White" into R23	13
Draining 1 batch "Blue" from R21 into B31	12
Draining 1 batch "Green" from R21 into B32	13
Draining 1 batch "Blue" from R22 into B31	12
Draining 1 batch "Green" from R22 into B32	12
Draining 1 batch "Blue" from R23 into B31	12
Draining 1 batch "Green" from R23 into B32	12
Pumping 3 batches "Red" out of B31	30
Pumping 3 batches "Green" out of B32	30

Table 3. Processing times.

Work orders, given in table 4, illustrate the problem, where six batches of "Blue" (PB) and six batches of "Green" (PG) products have to be produced. It follows that we need six batches of "Yellow" and "Red" raw materials and twelve batches of "White" raw material.

Product	Code	Amount	Starting time
Blue	PB	6	0
Green	PG	6	0

Table 4. Work orders.

5.3. Structure of the production facility given in production management system

Data about the production process and its structure can be obtained from the production management information systems. These data can be presented in a form of a recipe and the formula as described in chapter 3.4.

There are two recipes available, which are specifying a sequence of operations needed to produce product "PB" and product "PG". They are given with tables 5 and 6.

	Operation	Duration	Resources
PB	Op10	–	BOM_B
	Op20 draining R2x in B31	12/12/12	[R_1](R21/R22/R23), B31
	Op30 pumping from B31	30	$(3 \times 1) \times$B31
Y	Op10 pumping Y in B11	12	[BY_1](B11)
	Op20 draining B11 in R2x	15/12/12	[R_3](R21/R22/R23), [BY_2](B11)
WB	Op10 pumping W in B13	12	[BW_1](B13)
	Op20 draining B13 in R2x	10/9/13	[R_2](R21/R22/R23), [BW_2](B13)

Table 5. Recipe of a product "PB".

	Operation	Duration	Resources
PG	Op10	–	BOM_ G
	Op20 draining R2x in B32	13/12/12	[R_ 1](R21/R22/R23), B32
	Op30 pumping from B32	30	$(3 \times 1) \times$B32
R	Op10 pumping Y in B12	12	[BR_ 1](B12)
	Op20 draining B12 in R2x	11/13/14	[R_ 3](R21/R22/R23), [BR_ 2](B12)
WG	Op10 pumping W in B13	12	[BW_ 1](B13)
	Op20 draining B13 v R2x	10/9/13	[R_ 2](R21/R22/R23), [BW_ 2](B13)

Table 6. Recipe of a product "PG".

To produce Blue product ("PB"), firstly operation *Op10* has to be carried out. This operation determines only that a sub-product, defined with formula "B", is needed. When this sub-product is ready, two more operations are needed. *Op20* gives information about draining the product into the buffer tank *B31* and *Op30* about pumping the final product out of that buffer tank. The fact, that buffer tank could not be emptied before three batches of sub-product "B" are poured in it, is described with this notation: $(3 \times 1) \times B31$.

In batch manufacturing situations when one task has to be executed with a (group of) resource(s), that were used to execute a previous task already are common. These resources are in our case labelled with an additional mark, i.e. code with serial number in square brackets is added in front of this resource(s). For example, this occurs in our case when one of the reactor is being used. Code $[R\_1]$ is assigned to the reactor (from a group of reactors $R2x$) which is needed for operation $Op20$ when producing product "B" and indicates that this resource can now be released. Note, that this resource was assigned with an operation where code $[R\_3]$ was used already (*Op10* of a sub-product "Y").

Formula given in Table 7 lists the raw materials needed to produce items "B" and "G". Item "B" represents a sub-product that is needed for operation *Op10* of a recipe "PB". The production of one item "B" requires one "Yellow" ("Y") and one "White" ("WB") batches (items). Each of these two items are produced with two operations. Note that some batch-specific precedence constraints exists. Operation *Op20* of item "Y" precede operation *Op20* of item "WB". The structure of item "G" is given in a similar way.

Item	Sub-item	Quantity	Preced. constr.			
B	Y, Op10	1	0	0	0	0
	Y, Op20		0	0	0	1
	WB, Op10	1	0	0	0	0
	WB, Op20		0	0	0	0
G	R, Op10	1	0	0	0	0
	R, Op20		0	0	0	1
	WG, Op10	1	0	0	0	0
	WG, Op20		0	0	0	0

Table 7. Formula for the items "B" and "G".

5.4. Modelling

In this subchapter production process described previously is modelled with timed Petri nets. To build a model the algorithm from Chapter 3.5 is used. This model can later be used to schedule all the tasks, that are necessary to produce as much final products as required by work orders.

From work orders, given in table 4 it is recognised which and how much of each products are required. Let start with the procedure on building the Blue product ("PB"). When applying the first step of our algorithm, the PN structure shown in figure 12 is achieved.

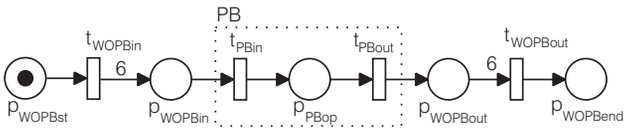

Figure 12. PN model of PB product.

In the second step, additional information are added into this model (framed part of figure 12). Data about all the details are gathered from recipe list of product "PB" (table 5). Information about emptying the buffer tank *B*31 (*Op30*) and draining the reactor *R*2*x* (*Op20*) are added. As operation (*Op20*) needs resources, that are used by previous operations, not all details are added yet at this place. A model, shown in Figure 13 is achieved.

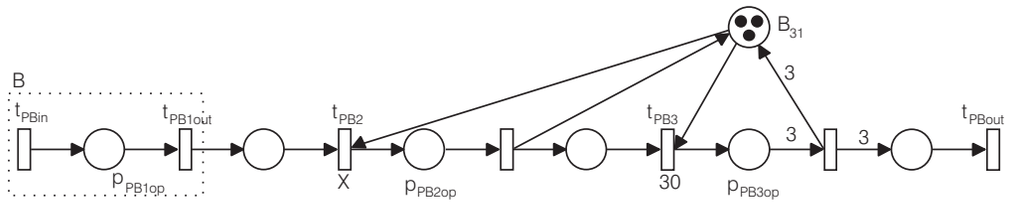

Figure 13. PN model of PB product.

Operation *Op10* (place P_{PB1op}) is defined with formula for item "B" (see table 7). It represents a mixing operation of two raw materials "Y" and "WB". Both sub-items are described with two operations, where precedence constraints are included as given with formula. Figure 14 shows how this formula information is modelled with Petri nets.

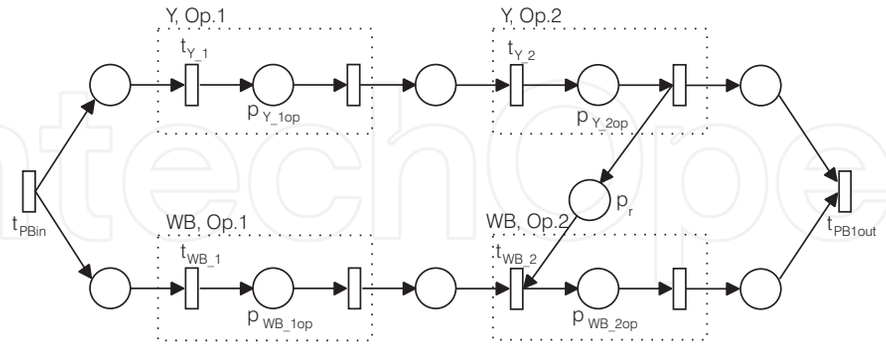

Figure 14. PN model of PB product with inserted BOM information.

In figure 15 some parts of a model given in figure 14 are simplified and information about the usage of rectors R_{2x} is added.

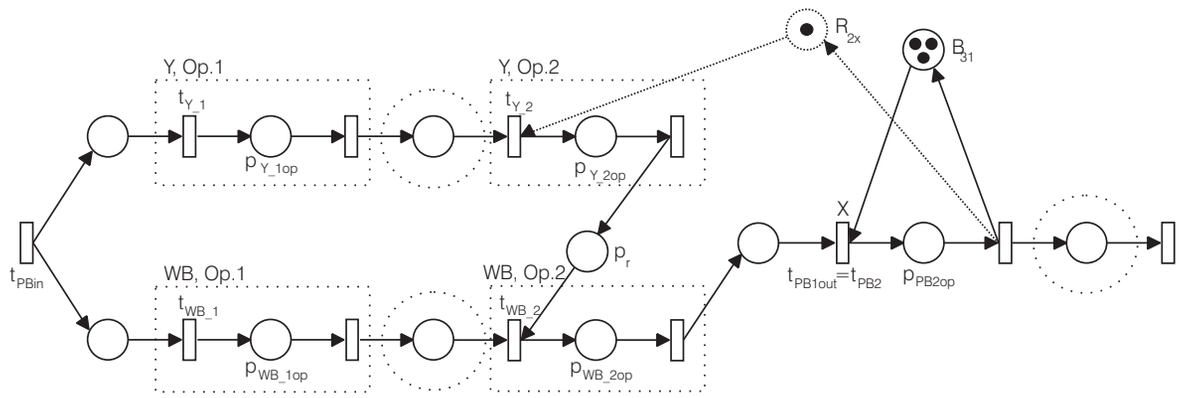

Figure 15. PN model of PB product.

As there are three possible reactors (R_{21}, R_{22} or R_{23}) that can be used to perform these operations this model is extended with the structure given in figure 16.

With this procedure a detailed timed Petri-net model of the production of the blue product ("PB") is obtained. The same procedure was performed to model also the production of green product ("PG"), and a Petri net model given in figure 17 is achieved.

5.5. Results

The resulting model was at the end verified using P-invariant analysis. We can find out eleven P-invariant, where eight of them refer to resources and three of them to the production routes.

In this way a Petri net model of a multiproduct batch plant was achieved on which different scheduling algorithms can be performed in order to obtain the most effective production. Petri-net simulation was used to evaluate different schedules of tasks that are needed to produce the desired amount of final products. Makespan was the performance measure of interest. The schedule allows an easy visualisation of the process and ensures that sufficient raw materials ("Yellow", "Red" and "White" batches) are available at the right time. It respects

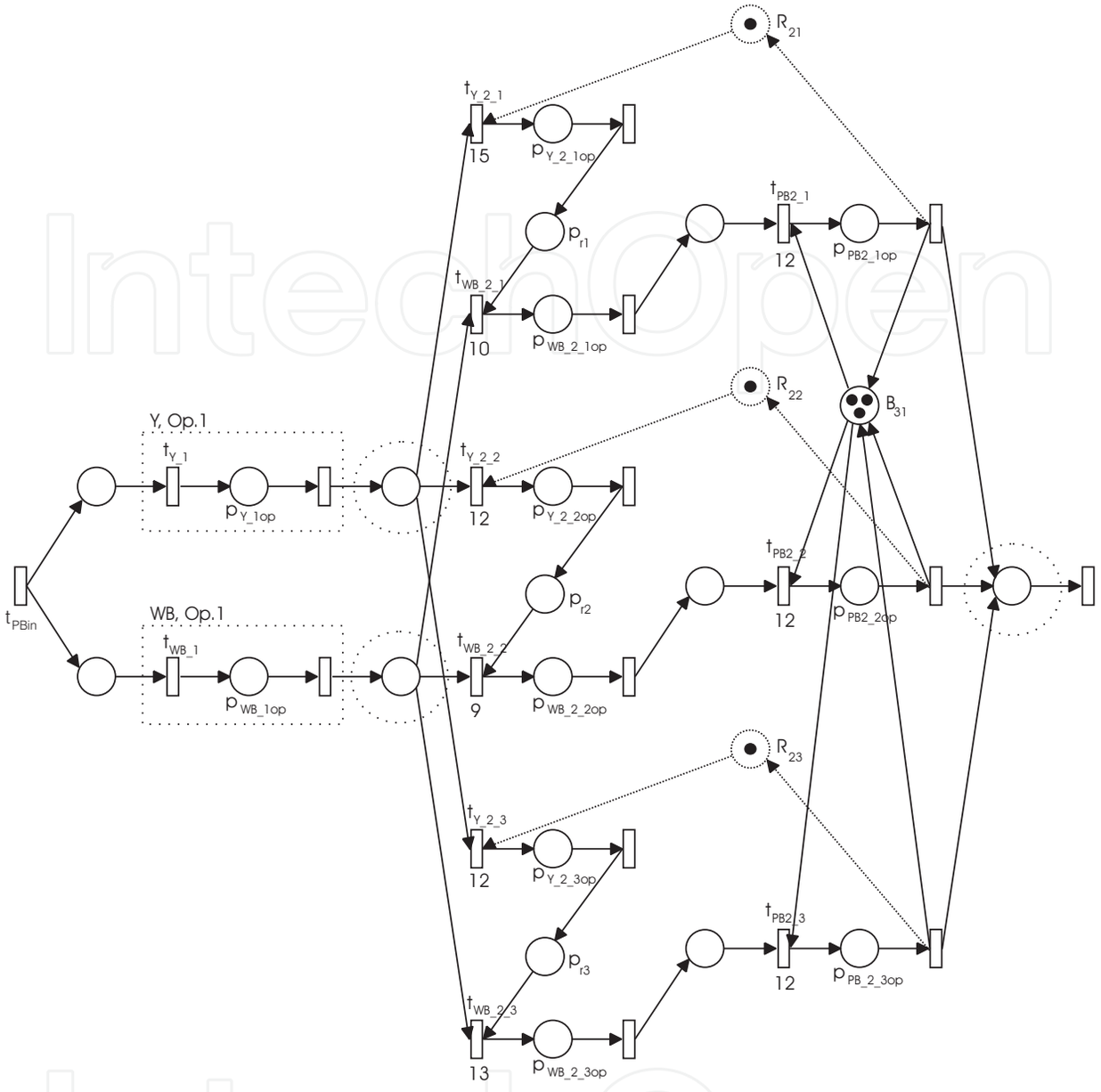

Figure 16. PN model of PB product.

all the production constraints and the duration of the whole process can be identified. A schedule of the batch process using SPT priority rule is given with Gantt chart (figure 18).

The results were compared with the results obtained using various algorithms and are presented in table 8.

Algorithm	Makespan
SPT rule	315s
LPT rule	331s
Branch and Bound ([17])	323s
MS Project ([3])	329s

Table 8. Results.

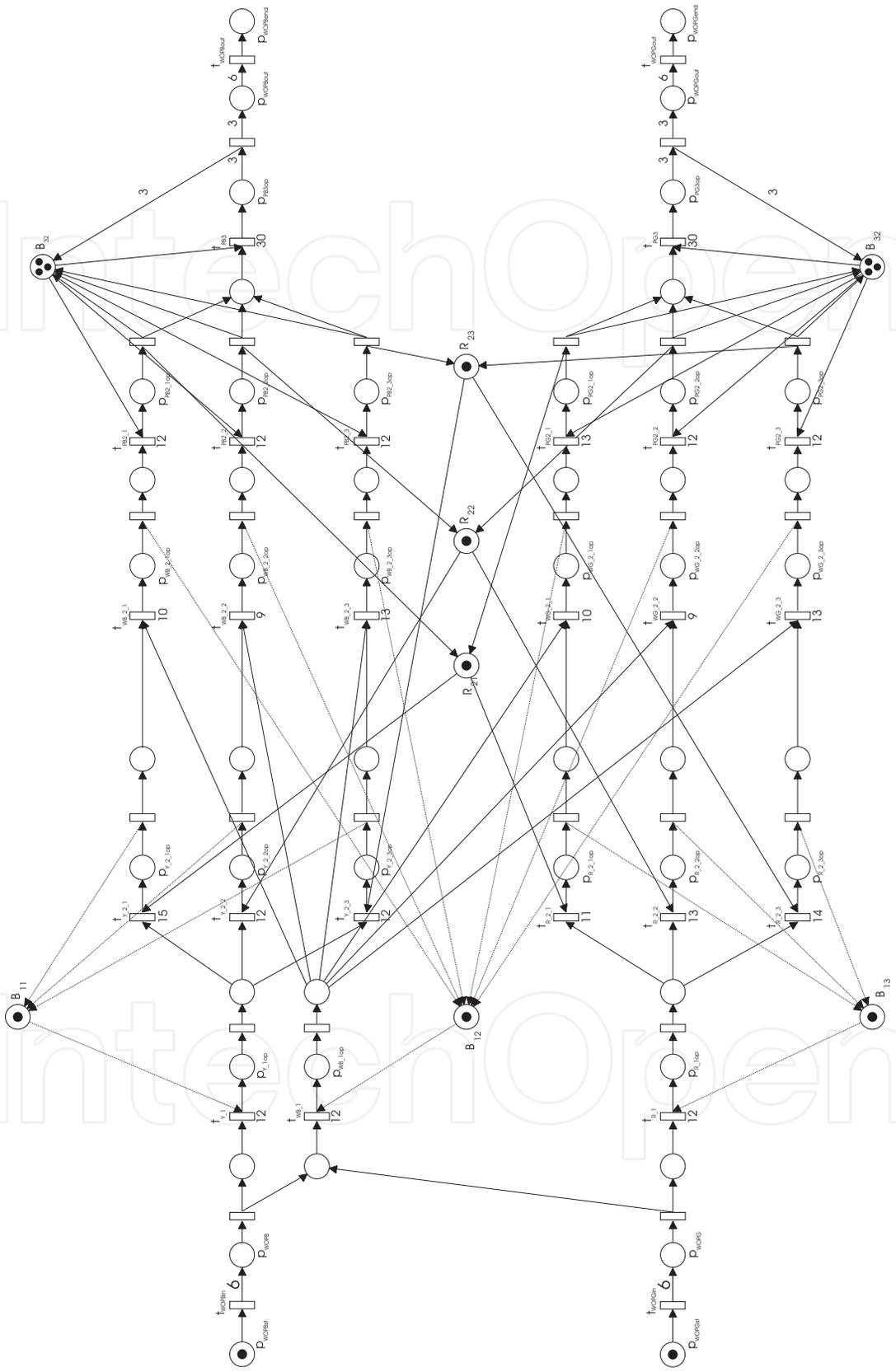

Figure 17. PN model of the production plant.

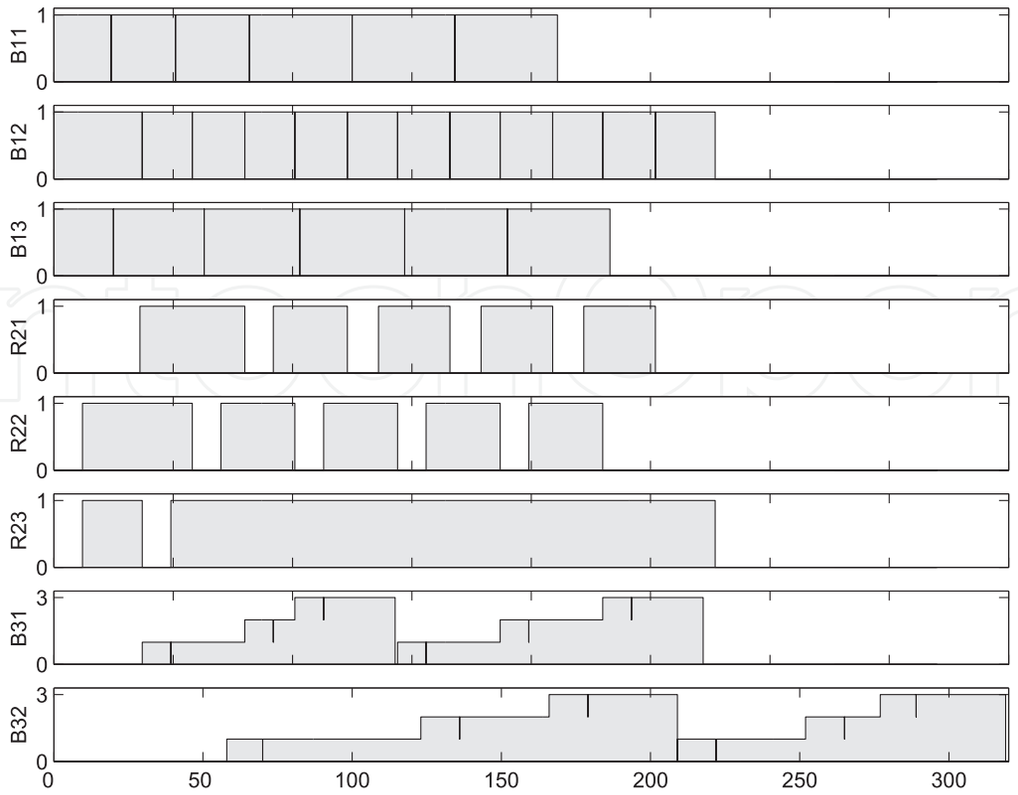

Figure 18. Production schedule.

6. Conclusion

A procedure for using existing data from production management systems to build the Petri-net model was developed. Timed Petri nets with the holding-duration principle of time implementation were used to model basic production activities. For the purposes of scheduling, different heuristic rules can be used within Petri net simulation. The applicability of the proposed approach was illustrated on a practical scheduling problem, where the data about the production facility is given with the formula and recipe. The model achieved with the proposed method was used to determine a schedule for production operations. The proposed method is an effective way to get an adequate model of the production process, which can be used to develop different analyses of the treated system, e.g. schedules.

Acknowledgements

The work was done in the frame of the Competence Centre for Advanced Control Technologies. Operation is partly financed by the Republic of Slovenia, Ministry of Education, Science, Culture and Sport and European Union (EU) - European Regional Development Fund.

Author details

Dejan Gradišar
Jožef Stefan Institute, Slovenia

Gašper Mušič
Faculty of Electrical Engineering, University of Ljubljana, Slovenia

7. References

[1] ANSI/ISA [1995]. *ANSI/ISA-88.01-1995 Batch Control Part 1: Models and Terminology (Formerly ANSI/ISA-S88.01-1995)*, ANSI/ISA.

[2] Bowden, F. D. J. [2000]. A brief survey and synthesis of the roles of time in Petri nets, *Mathematical and Computer Modelling* 31(10-12): 55–68.

[3] Gradišar, D. & Mušič, G. [2004]. Scheduling production activities using project planning tool, *Electrotechnical Review* 71(3): 83–88.

[4] Gu, T. & Bahri, P. A. [2002]. A survey of Petri net applications in batch processes, *Computers in Industry* 47(1): 99–111.

[5] Haupt, R. [1989]. A survey of priority rule-based scheduling, *OR Spectrum* 11(1): 3–16.

[6] Hillah, L., Kindler, E., Kordon, F., Petrucci, L. & Treves, N. [2009]. A primer on the Petri net markup language and ISO/IEC 15909-2, *Petri Net Newsletter* 76: 9–28.

[7] ISA [2008]. *ISA-TR88.95.01 Using ISA-88 and ISA-95 Together*, ISA.

[8] Jain, A. & Meeran, S. [1999]. Deterministic job-shop scheduling: Past, present and future, *European Journal of Operational Research* 113(2): 390–434.

[9] Jensen, K. [1997]. *Coloured Petri nets. Basic concepts, analysis methods and practical use*, Springer-Verlag, Berlin.

[10] Lakos, C. & Petrucci, L. [2007]. Modular state space exploration for timed Petri nets, *International Journal on Software Tools for Technology Transfer* 9: 393–411.

[11] Lee, D. & DiCesare, F. [1994]. Scheduling flexible manufacturing systems using Petri nets and heuristic search, *IEEE Trans. on Robotics and Automation* 10(2): 123–132.

[12] Méndez, C., Cerdá, J., Grossmann, I. E., Harjunkoski, I. & Fahl, M. [2006]. State-of-the-art review of optimization methods for short-term scheduling of batch processes, *Computers & Chemical Engineering* 30(6-7): 913–946.

[13] Mušič, G. [2009]. Petri net base scheduling approach combining dispatching rules and local search, *21st European Modeling & Simulation Symposium*, Vol. 2, Puerto de La Cruz, Tenerife, Spain, pp. 27–32.

[14] Nortcliffe, A. L., Thompson, M., Shaw, K. J., Love, J. & Fleming, P. J. [2001]. A framework for modelling in S88 constructs for scheduling purposes, *ISA Transactions* 40(3): 295–305.

[15] Piera, M. A. & Mušič, G. [2011]. Coloured Petri net scheduling models: Timed state space exploration shortages, *Math.Comput.Simul.* 82: 428–441.

[16] Pinedo, M. L. [2008]. *Scheduling: Theory, Algorithms, and Systems*, 3rd edn, Springer Publishing Company.

[17] Potočnik, B., Bemporad, A., Torrisi, F., Mušič, G. & Zupančič, B. [2004]. Hybrid modelling and optimal control of a multi product bach plant, *Control Engineering Practice* 12(9): 1127–1137.

[18] Silva, M. & Teruel, E. [1997]. Petri nets for the design and operation of manufacturing systems, *European Journal of Control* 3(3): 182–199.

[19] Tuncel, G. & Bayhan, G. [2007]. Applications of petri nets in production scheduling: a review, *The International Journal of Advanced Manufacturing Technology* 34(7-8): 762–773.

[20] Wijngaard, J. & Zijlstra, P. [1992]. MRP application the batch process industry, *Production Planning & Control* 3(3): 264–270.

[21] Wortmann, H. [1995]. Comparison of information systems for engineer-to-order and make-to-stock situations, *Computers in Industry* 26(3): 261–271.

[22] Yu, H., Reyes, A., Cang, S. & Lloyd, S. [2003]. Combined Petri net modelling and AI based heuristic hybrid search for flexible manufacturing systems-part II: Heuristic hybrid search, *Computers and Industrial Engineering* 44(4): 545–566.

[23] Zuberek, W. M. [1991]. Timed petri nets: definitions, properties and applications, *Microelectronics and Reliability* 31(4): 627–644.