Modeling, Analysis, Synthesis, and Performance Evaluation of Multioperational Production Systems With Hybrid Timed Petri Nets

George J. Tsinarakis, *Student Member*, Nikos C. Tsourveloudis, *Associate Member, IEEE*, and Kimon P. Valavanis, *Senior Member*

***Abstract*—Hybrid timed Petri nets (HTPNs) are derived to study random topology and complexity multioperational production systems where parts of one type follow the same route to produce a final product. Each production system is first decomposed into a fundamental multiproductive machine, multiassembly and multidisassembly modules, followed by derivation of their corresponding HTPN models. The overall system HTPN model is obtained via individual module synthesis, satisfying system constraints. Individual module and overall HTPN models nodes (places and transitions) are calculated. Individual module and overall system model invariants are derived mathematically. Performance and possible tradeoffs due to varying operational constraints (buffer capacity, work in process, machine utilization, backlog, etc.) are investigated through extensive simulations. Results show the applicability of the proposed methodology and justify its modeling power and generality.**

***Note to Practitioners*—This paper was motivated by the problem of analysis and performance evaluation of multioperational production systems. In these systems, machines are not dedicated but, at different time intervals, perform different operations. The overall complexity of most existing Petri net modeling and analysis approaches for production systems increases significantly with the size and the complexity of the considered system. This paper suggests a new general modular method for modeling, analysis, synthesis, and performance evaluation of random topology and complexity multioperational production systems using hybrid timed Petri nets (HTPNs). In this, the overall model construction is greatly simplified and its properties are easily obtained with respect to the features of three fundamental modules. The performance of our approach is evaluated by simulations of realistic, in terms of assumptions, manufacturing systems. The method presented provides a promising general use tool for studying multioperational production systems which, if appropriately modified, may be also applied to other types of discrete event systems.**

***Index Terms*—Complexity analysis, discrete and continuous components, hybrid timed Petri nets (HTPNs), modules, multioperational production systems, P-invariants.**

Manuscript received December 28, 2004; revised March 30, 2005. This paper was recommended for publication by Associate Editor M. Zhou and Editor N. Viswanadham upon evaluation of the reviewers' comments. This work was supported in part by the Technical University of Crete through "Herakleitos" Fellowships for research EPEAEK II 88727 and in part by the Greek Secretariat for Research and Technology/European Union, under Grant E.P.A.N. M.4.3.6.1, 2000ΣE 01330005.

G. J. Tsinarakis and N. C. Tsourveloudis are with the Department of Production Engineering and Management, Machine Tools Laboratory, Technical University of Crete, Chania GR 73 100, Greece (e-mail: tsinar@dpem.tuc.gr; nikost@dpem.tuc.gr).

K. P. Valavanis is with the Department of Computer Science and Engineering, Center for Robot-Assisted Search and Rescue, University of South Florida, Tampa, FL 33620-5399 USA (e-mail: kvalavan@csee.usf.edu).

Digital Object Identifier 10.1109/TASE.2005.857655

I. Introduction

HYBRID dynamic systems are characterized by the existence of nontrivial interactions between their continuous and discrete components, reflecting two distinct types of behavior: one described by a continuous function with respect to time, the other characterized by a sequential or discontinuous nature. Continuous or discrete valued variables or signals depend on independent variables such as (continuous or discrete) time [41]. Both behaviors, continuous and discrete, are complementary and essential to deriving the overall system model, called the hybrid model. A hybrid model represents simultaneously detailed evolutions of the system's continuous variables and the system's functional phase sequences. Discrete variables are used to describe switching or controlling network or changing environmental states, while continuous variables represent physical or other types of quantities, such as temperature, processed parts, fluids, and electrical potential. To incorporate hybrid system model unified specification, analysis, and synthesis, several mathematical and formal expressions have been proposed, including hybrid automata, ordinary differential equations, duration calculus, object-oriented paradigms, semantic networks, task trees, the *Branicky* model, and *Bond-Graphs* with commutation [34], [35], [41].

Hybrid Petri nets (HPNs) [30] are a viable mathematical and graphical tool, suitable to model and study hybrid systems. HPNs are capable of modeling sequences of phases of materials continuous flow with elementary discrete controllers [42]. HPNs consisting of a discrete PN and a constant speed continuous PN (CCPN) comprise a popular tool for practical applications, since they may be used for both system simulation and performance evaluation [21].

Multioperational production systems are considered as hybrid systems composed of a network of machines/workstations and buffers, where machines (that fail and are repaired randomly) may produce multiple product types at given time periods (not in parallel). Different part types follow different routes through the system but all parts of the same type follow the same route in the system—they cannot be used in alternative processes leading to different final products.

Ordinary Petri nets (OPNs) and their modifications and extensions have been widely used for modeling discrete event dynamic systems, production systems, and networks [2]–[8], [10]–[14], [40], in addition to several other control policies that have been introduced and tested [15], [17], [18], [22]–[24],

[28], [29]. However, multioperational systems are high-volume systems where machines dedicate different time intervals (of their total operational time) to produce a variety of product/part types according to their relative demand and followed production policies. Production times are short and the number of produced items/products are quite large. When this is the case, OPNs are rather inefficient as an analysis and synthesis tool. Reachability trees and graphs cannot be used in practice because the number of reachable discrete states increases exponentially ("explodes") and the number of events that need to be considered for realistic system simulation is extremely large [38]. Hence, the need to use HPNs is justified.

HPNs are defined by combining an OPN with a continuous Petri net (CPN) [9], [30]–[32], [51]. CPNs result from timed discrete PNs by "fluidification"; that is, by relaxing the condition that a marking is an integer vector [39]. In CPNs, tokens represent a real quantity of token fragments; this quantity unit is called a mark. Transitions move with the velocity of token fragments from the input to the output places [33]. The state space becomes infinite, allowing continuous dynamics modeling [36].

Our previously reported research studied the scheduling of single/multiple-part-type and re-entrant-dedicated production systems and networks of random topology and complexity [15], [17], [18]; it also provided a step-by-step methodology using t-timed ordinary modular PNs for studying such random topology-dedicated production systems [26].

The research work reported in this paper presents a novel general framework for modeling, analysis, synthesis, and performance evaluation of multioperational random topology and complexity production systems (single- or multiple-part-type or cyclically scheduled, with finite or infinite buffers), using hybrid timed Petri nets (HTPNs). The HTPNs introduced in this paper are an enhancement of the ones introduced in [16]; they consist of a discrete part modeled by a t-timed PN (TPN) and a CPN modeled by a constant speed continuous PN (CCPN). The main CCPN analysis tool is the evolution graph [1], [9], [30].

After introducing the three fundamental subsystems that are of general use in any system under study, for any random topology, and complexity multioperational production system, the following are accomplished.

- ***Production system decomposition—analysis***: The multioperational production system is decomposed into three sets of modules, corresponding to multiproductive machine, multiassembly module, and multidisassembly module. Their respective HTPN models are derived.
- ***Production system composition—synthesis***: The overall multioperational production system HTPN model is obtained via synthesis of the component models considering component connectivity and complexity: Component connectivity is determined by the system topology and module characteristics demonstrated by PN places fusion at the respective module connecting points. Component complexity and overall system complexity is determined by calculating the number of the HTPN module nodes and the overall multioperational system nodes. The overall HTPN models nodes are calculated with respect to the nodes of the used fundamental modules and places fusion, taking place at modules connection points. Also, the number as well as the form of the overall systems P-invariants is calculated with respect to the respective quantities of the fundamental subsystems.
- ***Production system constraints*** are considered in terms of the individual HTPN module as well as the overall model continuous and discrete invariants. This is carried out even for fused places where the exact determination/calculation of buffer capacities is essential to performance evaluation.
- ***Production system performance evaluation*** is achieved through simulations of a system's HTPN model. Buffer levels, machine utilization (up and downtime, idletime), production rates, cycle times, and overall production time are calculated and, if necessary, modified based on specifications and constraints. Simulations are carried out using the Visual Object Net software package [20], [37].

The most important contribution of the reported research to the best of the authors' knowledge is its originality since it is the first comprehensive effort to use HTPNs for creating a well-defined framework for detailed and systematic study of generalized production systems providing, at the same time, a justifiable mathematical foundation for HTPN-generalized invariant calculation. Another major contribution concerns the generalization of the methodology presented in [25] and [26] that makes it applicable to any dedicated and multioperational production system. Also, only three fundamental subsystems are necessary to represent in order to construct the HTPN model of any multioperation production system independently of its complexity. Additional contributions are that the HTPN's based system modeling, analysis, synthesis, and performance evaluation are independent of the original system architecture, topology, and structure; system analysis and synthesis is accomplished in terms of the fundamental HTPN modules; the system's complexity is easily obtained since the calculation of the overall HTPN system nodes and the derivation of the HTPN model invariants are done in general (theoretically) without considering a specific topology system; there is no limitation on the production system number of machines that compose the total net or topology, while buffer capacity may be considered to be either finite or infinite; the overall system properties may be calculated with respect to the respective properties of the fundamental HTPN modules models; and the fundamental modules models are simple, based in realistic assumptions, easily applied, and understandable.

A. Related Work

The use of CPNs and HPNs in applications related to manufacturing systems is still rather limited. A review of the most important relative literature follows. In [40], first-order HPNs (FOHPNs), a variation of timed HPNs, is used to study the first-order continuous behavior of hybrid systems by using linear algebraic tools. In particular, FOHPNs model a manufacturing system, demonstrating how system control can be viewed as a conflict resolution policy that aims at optimizing a given objective function (e.g., production rate, machines use). This is accomplished by solving a linear programming problem and computing an admissible instantaneous firing speeds vector. Use of linear algebra enables the performance of sensitivity analysis in order to study how optimal solutions change according to changes of the given linear problem [perturbations of the elements of the linear programming problem (LPP)]. The whole method is applied to manufacturing systems, modeled with FOHPNs. Machines are presented by continuous transitions, buffers by continuous places, while machine failures are presented as couples of discrete transitions. Multiclass machines, whose overall production rate is bounded, are considered by using two distinct transitions, one for each type of process.

In [43], FOHPNs and the instant firing speed vector calculation are introduced. Then, a simulation algorithm is provided to determine the state vector at the beginning of each macro-period of net's operation. A job-shop FOHPN model consisting of four machines, an assembly station, and seven buffers that finally produce two types of products is implemented using the multiclass machine, machine failure, and buffer models described in [40]. The proposed simulation algorithm is applied in order to compute the state vectors maximizing system throughput while minimizing the number of input parts during four macro-periods.

In [9], Petri net variations (ordinary, continuous, hybrid, and colored PNs) are introduced and their main properties and analysis tools are presented with applications examples. Then, *Grafcet*, a tool designed to represent logic controllers with the use of Boolean algebra is also introduced. A *Grafcet* is a graph having two types of nodes, steps, and transitions that is inspired by Petri nets with some differences. The main goal of the book is to introduce formally all of the described PN types and *Grafcet* and to define formally their features.

In [38], HPNs are used for modeling and simulation of large-scale semiconductor manufacturing systems. A Motorola plant workshop characterized by high throughput and a unique components routing is modeled and simulated using SIRPHYCO software. A general HPN model represents the majority of the system's machines. Only the furnace is represented by a different HPN model. Machines are reliable and may be in two states, assigned, or idle, while buffers are represented by continuous places. The furnace that is used for electronic components heat treatment can be seen as a server that has two states. The overall model is simulated for given values and the average buffer levels and machine's throughputs are calculated.

In [21], HPNs are combined with hybrid automata. The hybrid system under study is modeled with HPNs and then using an appropriate algorithm, the hybrid automaton that models the net's behavior is constructed for qualitative analysis of the system. The hybrid automaton is used as an alternative to the evolution graph to describe the net's behavior. It is stated that the proposed method can be applied only to HPNs with periodic functioning. The method is applied to a compact version of the Motorola plant described in [38]. Using the automaton mean marking of net's places, the mean firing frequencies of D-transitions and mean sojourn times in D-places are computed, and these results are validated with simulations of the system performed with SIRPHYCO.

In [36], a modification of HPNs is proposed where firing speeds can be functions of continuous net places markings. Hybrid dynamic nets (HDNs) are then combined with an object-oriented paradigm and then applied to model a hybrid manufacturing system, consisting of three basic subsystems: robot, comparator, and conveyor. Individual three-layer subsystems models are created and then combined together to form a manufacturing system's model that is simulated.

In [32], the main features of CPNs with variable speeds are introduced. Then, they are used to model the transient behavior of manufacturing lines. A manufacturing line consisting of two machines (two places and two transitions represents two buffers and two machines) and for a given initial marking evolution of the internal systems buffer is illustrated.

In [47], CPNs are used to model large series assembly workshops in which the final machine is a mono-input mono-output one. This work is concerned with the feedback control design of such systems. In particular, the average production frequencies of the machines are corrected according to the upstream and downstream buffers content to obtain the desired quantities of parts in the output buffers. Three alternative control laws are proposed, a bang-bang, a proportional, and a combinational one and, as an example, an open manufacturing line consisting of three machines and four buffers is considered and marking and speed evolutions of the net are presented. In [48], CPNs model production systems whose production frequencies are estimated from approximation of the firing speeds when they are not directly measurable from buffers content. A case study of a machine workshop of a car factory using the models introduced in [47] is presented.

The rest of the paper is organized as follows: Section II presents information related to the multioperational system fundamental modules summarizing functionality issues. Section III refers to hybrid PN fundamentals and derives the HTPN system modules. Section IV refers to a modules synthesis procedure, while a case study production system is analyzed and simulated in Section V. In Section VI, some significant issues are discussed while Section VII concludes the paper. Table I presents the nomenclature used in this paper.

II. Multioperational Production System Modules

Nondedicated, multioperational production systems consist of multioperational machines with known production and demand rates for every part type they process. Machines divide their time to coordinate different items production, satisfying system constraints. System components interaction, sequence of operations, the operations performed in each machine, and job control or order flow through a system with a given structure need to be determined accordingly [45], [46].

Multioperational production systems may be found in the literature with a variety of names. In [43], they are referred to as multiclass systems and are modeled using FOHPNs while, in [45], machines of this type are referred to as multiproduct machines and are modeled and analyzed using semi-Markov models and their control is optimized using Markov renewal

TABLE I
NOMENCLATURE

Symbol	*Explanation*
(O)PN	(Ordinary) Petri Net
TPN	Timed Petri Net
(C)CPN	(Constant Speed) Continuous Petri Net
H(T)PN	Hybrid (Timed) Petri Net
WIP	Work-in-process
DEDS	Discrete event dynamic system(s)
w_{ij}	Arc weight from node i to node j
P	Set of places
P_d, P_c	Subset of discrete, continuous places
T	Set of transitions
T_d, T_c	Subset of discrete, continuous transitions
I	Set of input arcs (to transitions)
O	Set of output arcs (from transitions)
V	Set of vertices
D / C	Discrete / Continuous node
h	Hybrid function, type of every node (D or C)
t	Maximum firing speed (C-transitions) or delay (D- transitions)
V_j	Maximum firing speed of C transition j
v_j	Instantaneous firing speed
N or $\mathbb{N}$	Set of nonnegative integers
R or $\mathbb{R}$	Set of real numbers
$n_i(t)$	Number of firings of D-transition i in (0,t)
m_0, m_f, m_i	Initial, final marking, marking i
$m_j(p_i)$	Number of tokens in place i in marking j
m_d, m_c	Discrete, continuous part of the marking
$R(m_i)$,$R(m_0)$	Reachability set of marking *i*, initial marking
n_{Pi}	Number of product types processed by multi-productive machine i.
n_{Ai1}, n_{Ai2}	Multi assembly i input, output buffer number
s_i	Number of parts participating in assemblies of multi assembly module i
q_i	Number of non valid combinations of raw material (from the possible) in module i
n_{Di1}, n_{Di2}	Number of input, output buffers of multi disassembly i
d_{ij}	Number of disassembly of jth initial part in module i, products
W	Incidence matrix
k	Upper bound of tokens found in a place
k_i	Number of parts preserved in set of places forming P-invariant i
S	Transition firing sequence
B_i, B_{ji}, B_{il}	Buffer i, upstream buffer, downstream buffer
M_j	Machine j
b_{ji} , b_{il}	Level of upstream, downstream buffer
C_i	Capacity of buffer i
y_j	Arc weight (equal to final buffer i capacity)
n_1	Number of multi-productive machines in the overall system
n_2	Number of multi-assembly modules in the overall system
n_3	Number of multi-disassembly modules in the overall system
n_4	Number of systems (external) input places
n_5	Number of nets output places (final buffers)
d_1	Number of machines of the overall system
d_2	Number of total system's P-invariants that refer to machine's mutually exclusive states
X	(n_p x 1) nonnegative integer vector
Y	(n_t x 1) nonnegative integer vector
p_{i-j}	Fused place arising from fusion of places i, j
a	Constant between zero and a maximum value

programming. Representative applications are found in automotive manufacturing companies, mainly to metal stamping and forming activities [45].

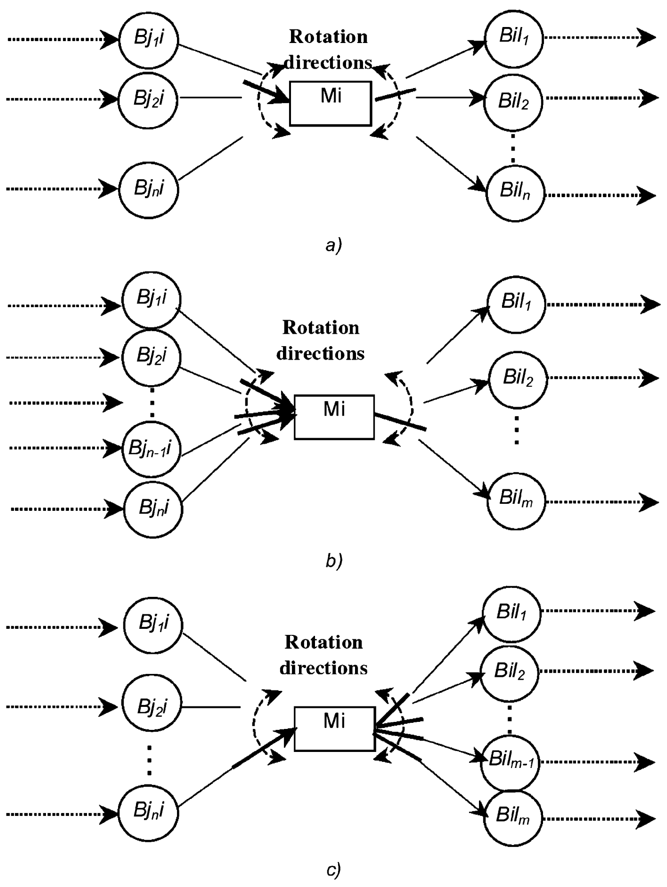

Fig. 1. Multioperational production systems fundamental modules. (a) Generalized multiproductive machine. (b) Generalized multiassembly machine module. (c) Generalized multidisassembly machine module.

Three fundamental modules are derived and are shown in Fig. 1. Circles (rectangles) represent buffers (machines), respectively. These modules, when connected to each other, represent manufacturing networks of various layouts.

A module is a fundamental subsystem with a set of input and output connection arcs defining interactions with other modules and a set of internal discrete and continuous relations that define the module's internal hybrid state. Bold rotating arcs represent a module's "modification basis" demonstrating how unprocessed parts enter machines and how they are removed when processed. They rotate to connect the appropriate fixed input-output buffer combinations. Rotating arcs indicate that machines are not dedicated but at given time periods, they produce different product types.

In the multiproductive machine module, there is one input and one output bold arc. Machine $\boldsymbol{M_i}$ receives parts from one of the $(\boldsymbol{n_{pi}})$ upstream buffers $\boldsymbol{B_{jni}}$ and after processing, sends products to the respective downstream buffer $\boldsymbol{B_{iln}}$ $(\boldsymbol{n_{pi}}$ output buffers exist). In the multiassembly module, the machine obtains parts from two or more from the $(\boldsymbol{n_{Ai1}})$ upstream buffers, assembles them to form a single product, and sends it to one from the $(\boldsymbol{n_{Ai2}})$ downstream buffer $\boldsymbol{B_{ilm}}$. The multiassembly module performs $\boldsymbol{n_{Ai2}}$ different types of assemblies and has multiple input and one output bold arcs. In a multidisassembly module, there is one input and multiple output bold arc. Machine $\boldsymbol{M_i}$ receives unfinished parts from one of the $(\boldsymbol{n_{Di1}})$ upstream buffers $\boldsymbol{B_{jni}}$, separates them, and sends products to a number of the $(\boldsymbol{n_{Di2}})$ output buffers. In multidisassembly, not all disassemblies produce the same number of products.

III. Fundamental Modules Hybrid Petri Net Models

A. Hybrid Petri Net Fundamentals

An OPN is defined as $\mathrm{PN} = \{P, T, I, O, m_0\}$, where $P = \{p_1, p_2 \ldots p_{np}\}$ is a finite set of places, $T = \{t_1, t_2, \ldots, t_{nt}\}$ is a finite set of transitions, $P \cup T = V$ with V the set of vertices, $I : (PxT) \to N$ and $O : (PxT) \to N$ the input and output functions with N a set of non-negative integers, and m_0 the PN initial marking described by tokens residing in places. A t-timed Petri net (TPN) results from the corresponding OPN by associating with each transition t_i a firing delay that may be constant or follow a given distribution, defined as $\mathrm{TPN} = \{P, T, I, O, m_0, D\}$ with D representing time delay, a function from the set of non-negative real numbers $\{0, \mathbb{R}^+\}$ [44].

HPNs are defined in detail in [1], [9], [30], and [51]. An HTPN is described by $\mathrm{HTPN} = \{P, T, I, O, h, \tau, m_0\}$ [21]. P is partitioned in subsets of continuous P_c and discrete places P_d, such that $P_c \cup P_d = P$ and $P_c \cap P_d = \emptyset$. T is also partitioned in continuous T_c and discrete T_d transitions such that $T_c \cup T_d = T, T_c \cap T_d = \emptyset, P \cap T = \emptyset$. I and O

$$I : \begin{cases} P_d xT \to \mathbb{N} \\ P_c xT \to \mathbb{R}_0^+ \end{cases}, O : \begin{cases} P_d xT \to \mathbb{N} \\ P_c xT \to \mathbb{R}_0^+ \end{cases}$$

are the preincidence and postincidence mappings specifying arcs. The set of arcs A is partitioned into two subsets of standard and inhibitor arcs. An inhibitor arc of weight r from a place P_i to a transition T_j allows the firing of T_j only if the marking of P_i is less than r. For all $t_j \in T_c, p_i \in P_d$ connected with standard arcs, $I(p_i, t_j) = O(p_i, t_j)$ must be verified. The hybrid function $h : P \cup T \to \{D, C\}$ indicates whether a place and/or transition is discrete or continuous. $\tau : \mathrm{T} \to \mathbb{R}^+$ associates with each transition a positive real number. For discrete transitions, the associated number corresponds to a time delay d_j, while continuous transitions are associated with maximal firing speeds $V_j = 1/d_j$. The D places initial marking are positive or null integers, while that of C places are positive or null real numbers. The marking and speed vector define completely the state of a CCPN. In HTPNs, for sets I and O, if p_i and t_j are a discrete place and a continuous transition $(h(p_i) = D, h(t_j) = C)$, then $I(p_i, t_j) = O(p_i, t_j)$ must be verified. This condition states that an arc joining a C transition to a D place demands the existence of the reciprocal arc and ensures the preservation of the integral character of discrete marking (marking of discrete places remains integer valued and cannot be modified by the firing of a continuous transition). Input and output places of discrete transitions can be continuous or discrete without restriction [19]. Generally, a discrete transition may have either discrete or continuous input and output places.

With regards to transition enabling and firing, a discrete transition t_j is enabled if every input place p_i to this transition meets the condition $m(p_i) \geq I(p_i, t_j)$. A continuous transition t_j is enabled if every input place meets the following conditions: 1) p_i is a discrete place $m(p_i) \geq I(p_i, t_j)$, or 2) p_i is a continuous place. In this case, there are two alternatives *i)* $m(p_i) > 0$ or *ii)* p_i is fed—p_i has a zero marking but it is supplied with the difference of the input flow less the output flow. The output transition is fired even if the marking of the place is instantaneously zero. An enabled continuous transition is strongly enabled if $m(p_i) > 0$ for every continuous input place p_i; it is weakly enabled otherwise.

Priorities are defined between continuous and discrete transitions for conflict cases. If there is a conflict between a discrete and a continuous transition, the discrete transition has priority over the continuous one. In case of conflict between several continuous transitions with a common empty continuous input place, any solution such that the sum of instantaneous firing speed of transitions feeding the place minus the sum of instantaneous firing speeds of transitions emptying the place is equal to 0 is admissible. When a common input place is discrete and contains a token, any solution such that $\sum_{j=1}^{\alpha}(v_j/V_j) = 1$ is admissible (α is the number of output transitions of the common place). The HTPN marking at time t reached from m_0 after firing a sequence of transitions S is $m(t) = m_0 + W * (n(t) + \int_{u=0}^{t} v(u) * du)$, where W is the incidence matrix, $n(t)$ is the number of times each D transition has fired between the initial time and time t, and $v(t)$ is the instantaneous firing speeds associated with C transitions at time t. This equation separates the discrete evolution from the continuous one and represents a trajectory in the marking space [9], [38].

A vector X of R^{n_p} is a P-invariant if $X^T * W = 0$. A vector Y of R^{n_t} is a T-invariant if $W * Y = 0$ [9]. The existence of P- invariants express a notion of token conservation in sets of places for all reachable markings without enumeration of the reachability set, while T-invariants are a necessary condition for a periodical functioning of an HTPN.

Most HTPN properties are the same with the corresponding OPN properties; however, in some cases, continuous variables require appropriate adaptation to accommodate such features.

Given an HTPN, continuous places are drawn with double circles

◎

discrete places as simple circles

○

continuous transitions are represented with double bars

▯

discrete transitions as simple bars. Immediate transitions are black bars

▮

and timed as empty bars

▯

.

In discrete places, tokens are shown as small black circles, while for continuous places, the number of tokens "residing" in each place is shown. Standard arcs are drawn as usual ($\rightarrow$) while inhibitors are represented by arcs whose end is marked with a small circle ($\multimap$).

HTPN behavior remains an event-driven model, although it contains a continuous functioning [9], [21]. Changes are described by the occurrence of three kinds of events. i) firing of a discrete transition according to a given time structure; ii) emptying of a continuous place; and iii) marking of a continuous place that is input to a discrete transition reaches a value equal to the weight of the arc between the place and the transition. An event of this kind modifies neither the markings of the D places nor the firing speeds of the C transitions but only the enabling conditions of discrete transitions. The state of an HTPN is defined by markings and residual reservation times for reserved markings. Since it is difficult to enumerate all of the reachable states, the evolution graph of an HTPN is represented by nodes corresponding to invariant behavior states (IB states). The IB state of an HTPN corresponds to a time period such that the marking of the discrete part m_d remains constant; the instantaneous speed vector v_c of the C transitions is constant; the set of D transition enabling is constant, and when the IB state is reached, m_c always has the same value.

B. HTPN Modules Fundamentals

Modules are derived based on realistic assumptions: 1) buffers have finite capacities and they are dedicated (one type of product is found in each buffer); 2) machines operate at given speeds that are redefined at specific moments according to events taking place; 3) machine breakdowns happen infinitely often; and 4) each machine changes the type of product produced at given time moments according to specific criteria after the selection of the appropriate machine setup.

The parts transfer to machines and machine setup after the changes of a produced part-type event are represented by timed transitions or continuous transitions with a given speed. When machine breakdowns occur, there is an immediate interruption of the process implemented at that time in the machine.

Tokens are shown for demonstration purposes. All transition input and output arc weights are equal to one except the ones leading to redefinition of the performed process type. Continuous places describe resource availability; discrete places correspond to discrete system states; discrete transitions describe state changes; and continuous transitions correspond to the speed of continuous events (e.g., operations). The discrete part may be considered as a type of "controller" that in given time moments redefines the product type manufactured with respect to specific predefined criteria (minimization of work-in-process (WIP), first-in–first-out (FIFO), buffer capacities, etc.), while the continuous part describes processes followed in the production phase that transform raw materials to final products. Discrete states are "configurations" of the process in a qualitative model used to change the control policy that is based on a continuous model.

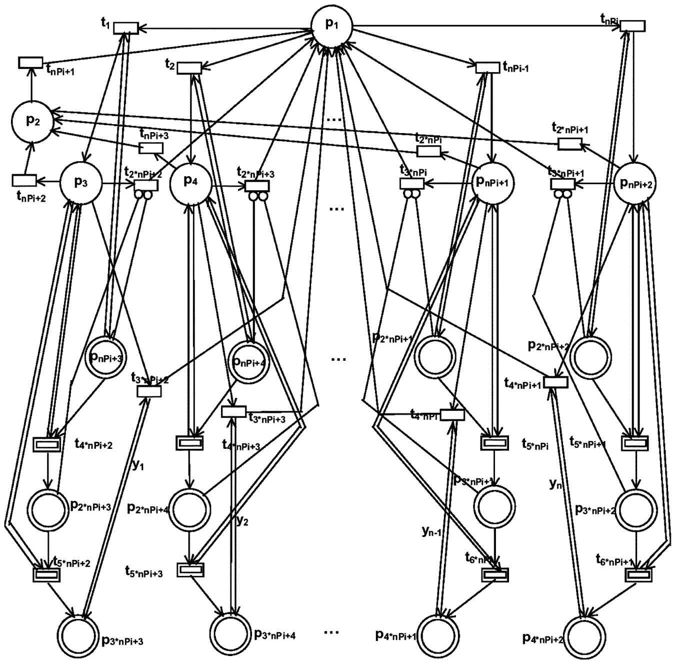

Fig. 2. Multiproductive machine module HTPN model.

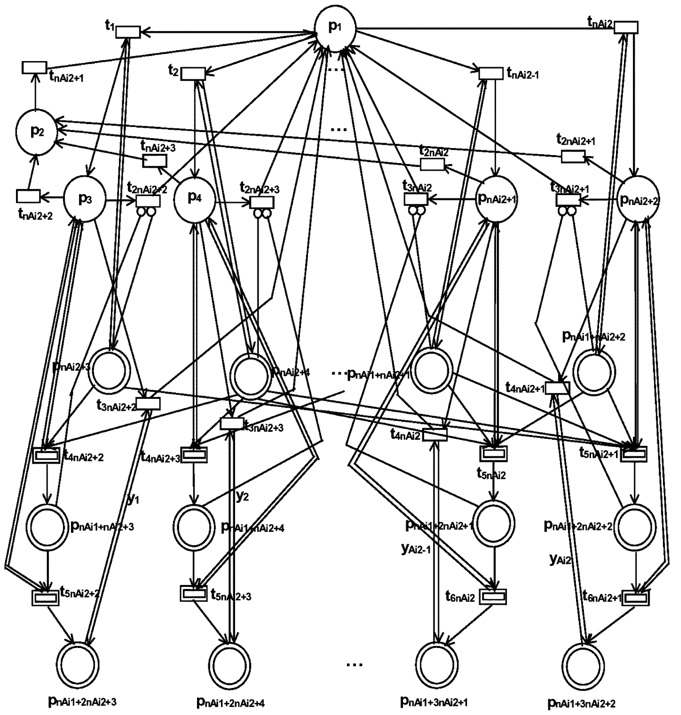

Fig. 3. Multiassembly machine module HTPN model.

C. Modules HTPN Models and Invariant Calculation

Multiproductive systems modules HTPN models are shown in Figs. 2–4. Table II explains the meaning of fundamental modules HTPN models' places and transitions.

The multiproductive machine module refers to a machine performing n_{pi} process types, each corresponding to a different incoming part. The discrete part of the net is modeled so that it is not possible for a machine to produce in parallel product types from different raw materials (this practically means that

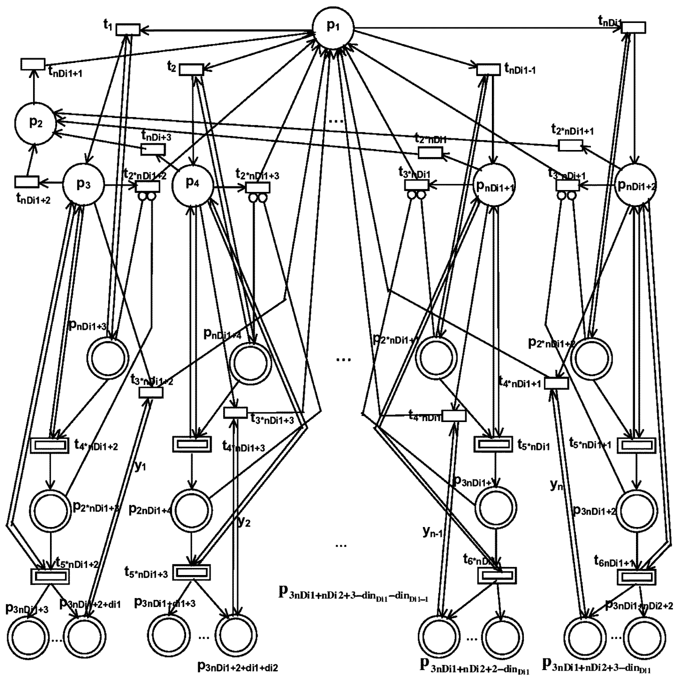

Fig. 4. Multidisassembly machine module HTPN model.

two continuous transitions of a module never fire concurrently). Net's discrete part has one token defining at each time instant its state (type of process performed, machine operational, and ready to produce or out of order). Place p_1 represents machine operational and ready to produce (waiting to define next process) and is connected with places $p_3 - p_{nPi+2}$ representing process types performed. p_2 represents a machine out of order and all net's discrete places representing process types are connected to it. When the machine is repaired after a breakdown, the process performed is not always the same as before; hence, p_1 is connected through timed transitions to all of the process types. After the repair of a breakdown, there is a structural conflict as all part types can be theoretically produced, but in fact, only one is produced at each time instant. This conflict is resolved with respect to system quantities (e.g., number of tokens in input buffers) and assigned priorities. In the net's continuous part, three continuous places for each process type represent initial buffers, parts at a machine for process, and final product buffers.

When all parts of a type have been processed or a final buffer reaches its maximum capacity, the process performed has to change in order not to lose time and production capabilities. The first event happens through transitions $t_{2*nPi+2} - t_{3*nPi+1}$. These are connected with places representing process types and through inhibitor arcs with initial buffers and places representing parts at machines, so that as soon as all parts of a type have been processed, no firing of these transitions can take place, and lead token to p_1 to redefine the process performed. The second event happens through $t_{3*nPi+2} - t_{4*nPi+1}$ that are also connected with places representing process types and with pairs of arcs of weight y_j with final buffers (y_i is equal to the maximum capacity of the respective final buffer). Transitions $t_{4*nPi+2} - t_{5*nPi+1}$ are connected with initial buffers and with pairs of arcs with process types places in order to represent parts supply at the machine. Finally, pairs of arcs connect process types places with process transitions since the process of a type in a machine has to stop when the net's state has changed.

The multiassembly module performs assemblies consisting of combinations of different numbers of initial parts, from two to n_{Ai1} = number of initial parts. Multiple parts of a type never participate in an assembly. The HTPN model has the same basic features with multiproductive machine module model. Its discrete part has one token since each possible assembly is represented by a place. Not all theoretical assemblies have a practical meaning; nonvalid assemblies are excluded. Moreover, parts resulting in a nonvalid assembly are never supplied concurrently to a machine. In nonvalid assemblies, the continuous transition

TABLE II
FUNDAMENTAL HTPN MODULES NODES (P AND T) MEANINGS

Node type	**Node**	**Model**	**Meaning**
P_d	p_1	All models	Machine operational and ready to process
	p_2	All models	Machine out of order
	$p_3 - p_{n_i+2}$	Productive, disassembly	Machine setup for type $1-n_i$ parts
	$p_3 - p_{n_{Ai2}+2}$	Assembly	Machine setup for type $1-n_{A_{i2}}$ parts
P_c	$p_{n_i+3} - p_{2n_i+2}$	Productive, disassembly	Initial parts buffers
	$p_{n_{Ai2}+3} - p_{n_{A_i}+n_{Ai2}+2}$	Assembly	
	$p_{2n_i+3} - p_{3n_i+2}$	Productive, disassembly	Type $1-n_i$ part process completed
	$p_{n_{A_i}+n_{Ai2}+3} - p_{n_{Ai}+2n_{Ai2}+2}$	Assembly	Type $1-n_{A_2}$ assembly completed
	$p_{3n_{P_i}+3} - p_{4n_{P_i}+2}$	Productive	Final product buffers
	$p_{n_{A_i}+2n_{Ai2}+3} - p_{n_{Ai}+3n_{A_{i2}}+2}$	Assembly	
	$p_{3n_{D_i}+3} - p_{3n_{D_i}+n_{Di2}+2}$	Disassembly	
T_d	$t_1 - t_{n_i}$	Productive, disassembly	Set machine for $1-n_i$ type process
	$t_1 - t_{n_{Ai2}}$	Assembly	Set machine for $1-n_{A_{i2}}$ type process
	t_{n_i+1}	Productive, disassembly	Breakdown repair
	$t_{n_{Ai2}+1}$	Assembly	
	$t_{n_i+2} - t_{2n_i+1}$	Productive, disassembly	Machine breakdown while performing $1-n_i$ type process
	$t_{nA_{i2}+2} - t_{2nA_{i2}+1}$	Assembly	Machine breakdown while performing $1-n_{A_{i2}}$ type assembly
	$t_{2n_i+2} - t_{3n_i+1}$	Productive, disassembly	Finish of $1-n_i$ type initial parts - change process type
	$t_{2nA_{i2}+2} - t_{3nA_{i2}+1}$	Assembly	Finish of $1-n_{A_{i2}}$ type initial parts -change assembly type
	$t_{3n_i+2} - t_{4n_i+1}$	Productive, disassembly	Final buffer for type n_i products is full - change process type
	$t_{3nA_{i2}+2} - t_{4nA_{i2}+1}$	Assembly	
T_c	$t_{4n_i+2} - t_{5n_i+1}$	Productive, disassembly	Process part
	$t_{4nA_{i2}+2} - t_{5nA_{i2}+1}$	Assembly	Perform assembly $1-n_{A_2}$
	$t_{5n_i+2} - t_{6n_i+1}$	Productive, disassembly	Move product from machine to output buffer
	$t_{5nA_{i2}+2} - t_{6nA_{i2}+1}$	Assembly	

describing their performance has zero firing speed, meaning that no token flow takes place. Final buffers of nonvalid assemblies have zero capacities (inhibitor arcs may alternatively be used to prohibit firing of the transitions). All discrete place capacities are equal to one. At different times

$$n_{Ai_2} = \sum_{s_i=2}^{n_{A_{i1}}} \frac{n_{A_{i1}}}{s_i\,(n_{A_{i1}} - s_i)} - q_i$$

types of assembly are performed ($q_i \geq 0$ is the number of nonvalid assemblies). Each part type participates at most in

$$c_1 = \sum_{s_i=2}^{nA_{i1}} \frac{(n_{A_{i1}} - 1)}{(s_i - 1)(n_{A_{i1}} - s_i)}$$

different types of assemblies. Arc connections are the respective ones described for the multiproductive machine.

A multidisassembly module is very similar to the multiproductive machine module. The main difference is that each initial part is disassembled in multiple final products (two or more final items are produced from the disassembly of one initial part). The most general case of the multidisassembly module is considered. In this, in each one of the $n_{D_{i1}} \geq 2$ processes performed d_{ij}, different product types are produced ($d_{ij} \geq 2, j = 1, \ldots, n_{D_{i1}}$). For each product, a different output buffer exists. The overall output buffers number is $n_{D_{i2}} = \sum_{i_2=1}^{n_{D_{i1}}} d_{ii_2}$. The number of tokens in the net's discrete part remains constant and equal to one; capacities of all discrete places are equal to one.

Considering the HTPN modules shown in Figs. 2–4 with any finite initial marking m_0, one may observe that 1) conflicts exist in their discrete net part since machines are not dedicated and produce multiple product types. Such conflicts are resolved during simulation by assigning priorities for the operational period. Conflict exists between parts process and machine breakdown and between different process types performance when a machine is operational and ready to process. The first type of conflict is easily resolved since the occurrence of a machine breakdown is not continuous and when the discrete transition leading to machine breakdown is enabled, it has the highest priority. In the second case, each time decision is made according to net status between the part types that are available in the input buffers. 2) As long as there is parts availability in the input buffer(s), operations continue until a breakdown occurs. 3) Modules are not generally live. No deadlock occurs in their discrete part as there is parts availability—each transition has only one preceding place and there is always one token in the discrete part of the net (discrete part reaches a steady state—machine ready to process—as soon as the entire initial parts process has ended). The continuous parts of the nets are also not generally live. They remain live as long as parts in some of the initial buffers have not been exhausted. The initial parts in buffers define the duration that the net remains live. 4) All modules are *k-bounded*. The absence of self-loops in combination with the fact that modules are totally covered by P-invariants ensuring token preservation, guarantees *k-boundedness*. 5) Only the multiproductive machine module is conservative. The multiassembly module uses multiple parts for the production of one product, while the multidisassembly module produces multiple products from one initial part. Thus, they have at least one nonconservative continuous transition for each product type). 6) All modules are nonpersistent due to the existence of conflict transitions in the discrete parts of their nets. Further, in the multiassembly machine module, there is an additional conflict in the continuous part since some initial parts types may participate in different products. 7) Token preservation and the machine mutually exclusive states are described by the respective P-invariants. 8) Modules are not repetitive and not consistent—there are no repetitive sequences of transitions whose firing results in the initial marking or in the periodic appearance of a restricted number of markings. Thus, there are no T-invariants.

The upper limit of tokens that may be found concurrently in continuous net places is defined with respect to initial markings $m_0(p)$ and place capacities C_i.

For the multiproductive machine module, this upper limit is $\max_{j=n_{pi}+3_j}^{2n_{Pi+2}}\{F_{jP}\}$ where

$$F_{jP} = \min\left\{\max\left\{C_j, C_{j+n_{P_i}}, C_{j+2n_{P_i}}\right\},\right.$$
$$\left. m_0(p_j) + m_0\left(p_{j+n_{P_i}}\right) + m_0\left(p_{j+2n_{P_i}}\right)\right\}.$$

To calculate F_{jP}, the three places representing final and initial buffers and the machine for each product type are considered, and their initial tokens are added. This number is compared to the maximum of the capacities of these places and the minimum of the two values is calculated. The maximum calculated F_{jP} is the upper limit of tokens found in the net.

The respective quantities for multiassembly and multidisassembly modules are calculated by similar equations that are omitted due to space limitations. The main differences are that in multiassembly, all of the input buffers contain raw materials for each assembly while in multidisassembly, the d_{ij} final product buffers of each initial part are taken into account in the equation. Calculations of these quantities ensure the k-boundedness of fundamental modules and production systems.

Fundamental modules P-invariants are derived as a function of the number of processes performed in each machine, the number of input parts in each assembly, the number of parts produced by each disassembly, and the number of allowed raw material combinations in assemblies. Each module's structural complexity (number of nodes) is also derived.

The multiproductive machine module has $(n_{pi} + 1)$ P-invariants. One P-invariant refers to the mutually exclusive states of the machine; n_{pi} states correspond to the machine setups for the different processes performed, one refers to machine breakdown, and one to machine being operational and ready to produce. The other P-invariants refer to the preservation of tokens (parts) within the system, where $k_i, i = 1, \ldots, n_{pi}$ is the initial sum of tokens in the respective set of places. The P-invariants are

$$\begin{gathered}
m(p_1) + m(p_2) + \cdots + m(p_{nPi+1}) + m(p_{nPi+2}) = 1\\
m(p_{nPi+3}) + m(p_{2nPi+3}) + m(p_{3nPi+3}) = k_1\\
m(p_{nPi+4}) + m(p_{2nPi+4}) + m(p_{3nPi+4}) = k_2\\
\vdots\\
m(p_{2nPi+1}) + m(p_{3nPi+1}) + m(p_{4nPi+1}) = k_{nPi-1}\\
m(p_{2nPi+2}) + m(p_{3nPi+2}) + m(p_{4nP+2}) = k_{\mathrm{nPi}}.
\end{gathered}$$

The first P-invariant refers to the discrete part of the net, while the remaining n_{pi} refers to the continuous one. In the discrete part of the net (Fig. 2), the n_{pi} places corresponding to the machine setups must be connected with the place representing machine operational, be ready to produce both directions, and must also be connected with the place representing machine breakdown since breakdowns may happen at any net state.

The multiassembly machine module has $(n_{Ai1}+1)$ P-invariants. One refers to the discrete part of the net describing the mutually exclusive machine states (n_{Ai2} states represent machine setups for different assembly types, one refers to machine breakdown, and one to the machine being operational and ready to produce). The other P-invariants refer to the continuous part of the net, describing tokens preservation for each initial part type within the system. In these, $k_i, i = 1, \ldots, n_{Ai1}$ is the initial sum of tokens in the respective set of places. Each such invariant consists of two places representing the initial buffer and i type of parts entering the machine, and by c_{2i} places representing final buffers in which products whose initial part is the component with

$$c_{2i} \le c_1 = \sum_{s_i=2}^{nA_{i1}} \frac{(n_{A_{i1}}-1)}{(s_i-1)(n_{A_{i1}}-s_i)}.$$

The equality stands when all of the assemblies in which the initial part i participates are valid. Letting $m(p_{n_{Ai1}+2n_{Ai2}+3}) = A_1, \ldots, m(p_{n_{Ai1}+3n_{Ai2}+2}) = A_{n_{Ai2}}$ the P-invariants of the multiassembly machine module are

$$\begin{aligned}
&m(p_1)+m(p_2)+\cdots+m(p_{nAi2+1})+m(p_{nAi2+2})=1\\
&m(p_{nA_{i2}+3})+m(p_{nA_{i1}+nA_{i2}+3})+A_1\\
&\quad+A_2+\cdots+A_{c_{21}-1}+A_{c_{21}}=k_1\\
&m(p_{nA_{i2}+4})+m(p_{nA_{i1}+nA_{i2}+4})+A_1\\
&\quad+A_2+A_4+\cdots+A_{c_{22}}+A_{c_{22}+1}=k_2\\
&\qquad\qquad\vdots\\
&m(p_{nA_{i1}+nA_{i2}+2})+m(p_{nA_{i1}+2nA_{i2}+2})+A_{n_{Ai2}-c_{2n_{Ai1}}}+\cdots\\
&\quad+A_{n_{Ai2}-1}+A_{n_{Ai2}}=k_{n_{A_{i1}}}.
\end{aligned}$$

The multidisassembly machine module has $(n_{Di2}+1)$ P-invariants. The first P-invariant refers to the mutually exclusive machine states (n_{Di1} states represent the different types of disassemblies, one refers to machine breakdown, and one to the machine being operational and ready to produce). The remaining invariants refer to token preservation within the system with $k_i\ i = 1, \ldots, n_{Di2}$ the initial sum of tokens in the respective set of places. Each d_{ij} P-invariant refers to the same initial buffer and part types in a machine with the final buffer only being different. The P-invariants are

$$\begin{aligned}
&m(p_1)+m(p_2)+\cdots+m(p_{nDi1+1})+m(p_{nDi1+2})=1\\
&m(p_{nDi1+3})+m(p_{2nDi1+3})+m(p_{3nDi1+3})=k_1\\
&m(p_{nDi1+3})+m(p_{2nDi1+3})+m(p_{3nDi1+4})=k_2\\
&\qquad\qquad\vdots\\
&m(p_{nDi1+3})+m(p_{2nDi1+3})+m(p_{3nDi1+2+di1})=k_{di1}\\
&m(p_{nDi1+4})+m(p_{2nDi1+4})+m(p_{3nDi1+3+di1})=k_{di1+1}\\
&m(p_{nDi1+4})+m(p_{2nDi1+4})+m(p_{3nDi1+4+di1})=k_{di1+2}\\
&\qquad\qquad\vdots\\
&m(p_{nDi1+4})+m(p_{2nDi1+4})\\
&\quad+m(p_{3nDi1+2+di1++di2})=k_{di1+di2}\\
&\qquad\qquad\vdots\\
&m(p_{2nDi1+2})+m(p_{3nDi1+2})\\
&\quad+m(p_{3nDi1+nDi2+3-din_{Di1}})=k_{nDi2-din_{Di1}+1}\\
&\qquad\qquad\vdots\\
&m(p_{2nDi1+2})+m(p_{3nDi1+2})\\
&\quad+m(p_{3nDi1+nDi2})=k_{nDi2-1}\\
&m(p_{2nDi1+2})+m(p_{3nDi1+2})\\
&\quad+m(p_{3nDi1+nDi2+1})=k_{nDi2}.
\end{aligned}$$

TABLE III
HTPN MODULES NODE COMPLEXITY

Model	*Node type*	*Nodes number*
Multi-productive machine	*DP*	$n_{Pi}+2$
	DT	$4n_{Pi}+1$
	CP	$3n_{Pi}$
	CT	$2n_{Pi}$
Multi-Assembly	*DP*	$n_{Ai2}+2$
	DT	$4n_{Ai2}+1$
	CP	$n_{Ai1}+2n_{Ai2}$
	CT	$2n_{Ai2}$
Multi-Disassembly	*DP*	$n_{Di1}+2$
	DT	$4n_{Di1}+1$
	CP	$2n_{Di1}+n_{Di2}$
	CT	$2n_{Di1}$

From the above P-invariants, the first refers to the net's discrete part while the remaining $n_{Di2} * d_i$ refer to the continuous one.

The HTPN module complexity is shown in Table III. The number of discrete places for all modules is equal to the number of processes performed in the machine, increased by two (machine breakdown and machine operational and ready to produce). The number of discrete transitions is also common for all modules and is equal to the number of processes multiplied with four and increased by one. The continuous nodes numbers of the modules are also calculated with respect to the numbers of the corresponding input and output buffers of the modules.

IV. HTPN MODULE SYNTHESIS

Module synthesis may be more easily understood once systems P-invariants are derived. Therefore, given a production system, P-invariants are theoretically calculated at first as a function of the modules, followed by synthesis rule justification.

It is emphasized that P-invariants are of two main types: one referring to the mutually exclusive machine(s) states and the other referring to preservation of parts in the system.

A. *Theoretical Calculation of Invariants*

Initially, the calculated P-invariants are distinguished into two types: the ones referring to machine mutually exclusive states (d_1) and the rest referring to parts preservation in the buffers (d_2). Machine mutually exclusive states P-invariants are as many as the modules used to build the overall model, since each module has one such P-invariant as it has one machine. Considering a production system HTPN model consisting of n_1 multiproductive machines, n_2 multiassembly modules, and n_3 multidisassembly modules, d_1 is calculated as $d_1 = n_1 + n_2 + n_3$. The form of this P-invariant arises from the used submodel since it describes the mutually exclusive machine states (machine setups for production of different products, machine operational, and ready to produce or machine out of order). So d_1 P-invariants, as a whole, are obtained from the individual submodels and their number is calculated according to the number of modules used.

In addition, there are d_2 P-invariants from the second type (parts preservation). In general, for an overall system, each of the d_2 P-invariants arises by appropriately connecting the respective P-invariants of the individual modules so that they describe a route from an initial buffer to the final products buffer, containing all of the in-process buffers where a part is found. In the overall systems P-invariants in modules connection points, the places representing first modules final buffer and second modules initial buffer are substituted by a fused place representing their common buffer in the system.

It is assumed that in a multiproductive system, a part type (initial or in process) cannot be used for the production of multiple product types. This means that the whole content of a buffer follows a unique route in a system. The only exception is multiassembly module, where different raw material combinations are possible. d_2 calculation is derived independently of system topology and structure. The use of a machine for multiple processes does not affect token preservation since, at each time period, a machine processes at the most one part type. Disassembly and assembly processes are responsible for the generation of multiple invariants. So d_2 is

$$d_2 = \sum_{i=1}^{n_4} \left[n_{D_{i,1}} - \sum_{i_1=1}^{o_1} n_{A_1,i_1} + n_{A_{1,1}} * E_{1,1} + n_{A_{1,2}} * E_{1,2} + \cdots + n_{A_{1,o_1}} * E_{1,o_1} \right]$$

$$E_{1,1} = \left[n_{D_{i,2,1}} - \sum_{i_{2,1}=1}^{o_{2,1}} n_{A_2,i_{2,1}} + n_{A_{2,1,1}} * E_{2,1} + \ \ldots + n_{A_{2,1,o_{2,1}}} * E_{2,o_{2,1}} \right]$$

$$E_{1,o_1} = \left[n_{D_{i,2,o_1}} - \sum_{i_{2,o_1}=1}^{o_{2,o_1}} n_{A_{2,i_{2,o_1}}} + n_{A_{2,o_1,1}} * E_{2,o_1} + \cdots + n_{A_{2,o_{2,o_1}},1} * E_{2,o_{2,o_1}} \right].$$

As it is obvious from the above-calculated equations, the calculation of d_2 takes into account the respective numbers of invariants of the submodels $(n_{pi}, n_{Di2}, \text{and}\, mn_{Ai1})$.

For each of the n_4 initial part types, processes (throughout the system) are sequentially considered. The number of parts produced by the first received disassembly $(n_{Di,1})$ are calculated first. For each part type, two sequential disassemblies define a level. The first disassembly defines a level's starting point and the second is the ending point being at the same time, the next levels' starting point. From $n_{Di,1}$, the number of first-level disassembly products participating in assemblies of the same level are subtracted $(\sum_{i_1=1}^{o_1} n_{A_1,i_1}$, where o_1 is the number of these level assemblies). When no assembly takes place in a level, multiproductive machine processes are considered (in the equation) as one-part assemblies. For every first-level assembly, the product of the number of inputs to assembly parts $(\mathrm{n_{A1,k}})$ and the respective factor $E_{1,k}, k = 1, \ldots, o_1$ is added to the already calculated quantity. $E_{1,k}$ refers to the part of the equation that has been described, adapted to second-level quantities (products of second-level disassembly; second-level disassembly products participating in second-level assemblies + the product of the number of input to second-level assembly parts $(n_{A2,k})$ and the respective factor $\mathrm{E_{2,j}}$. $\mathrm{E_{2,j}}$ is received in a similar way for third-level quantities. This process is repeated for the number of levels defined by the disassemblies of each products process sequence. Respective factors $\mathrm{E_{3,1}, E_{3,2}, \ldots, E_{n,1,\ldots}}$, and $\mathrm{E}_{\mathrm{n},l}$ are calculated in a similar way, where l is the last disassembly of the last level). If an initial part does not participate in any disassembly, $n_{Di,1}$ is one. For the system of Fig. 7, $\mathrm{d_2} = (2 - 2 + 1 * 1 + 1 * 1) + (2 * 2 - 0) + (1 * 2 - 1 + 1 * 1) = 8$.

B. Theoretical Nodes Calculation

Considering the model of a multioperational production system consisting of n_1 multiproductive machine modules, n_2 multiassembly modules, n_3 multidisassembly modules, n_4 input places, and n_5 output places, its overall transitions number is $\sum_{i=1}^{n_1}(6n_{P_i} + 1) + \sum_{j=1}^{n_2}(6n_{A_{j2}} + 1) + \sum_{k=1}^{n_3}(6n_{D_{k1}} + 1)$ where n_{Pi} refers to the number of processes performed by the multiproductive machine, i, n_{Aj1} refers to the raw materials used by multiassembly machine j, and n_{Dk2} refers to the number of product types produced by the multidisassembly machine k. From these, $\sum_{i=1}^{n_1}(4n_{P_i} + 1) + \sum_{j=1}^{n_2}(4n_{A_{j1}} + 1) + \sum_{k=1}^{n_3}(4n_{D_{k2}} + 1)$ are discrete and $2\sum_{i=1}^{n_1} n_{P_i} + 2\sum_{j=1}^{n_2} n_{A_{i2}} + 2\sum_{k=1}^{n_3} n_{D_{k1}}$ are continuous. The overall places number is $\sum_{i=1}^{n_1}(4n_{Pi} + 2) + \sum_{j=1}^{n_2}(n_{A_{i1}} + 3n_{A_{i2}} + 2) + \sum_{k=1}^{n_3}(3n_{D_{i1}} + n_{D_{i2}} + 2)$. Considering place fusion, this number is reduced by

$$\frac{1}{2}\left[2\sum_{i=1}^{n_1} n_{P_i} + \sum_{j=1}^{n_2}\left(n_{A_{j1}} + n_{A_{j2}}\right) + \sum_{k=1}^{n_3}\left(n_{D_{k1}} + n_{D_{k2}}\right) - n_4 - n_5\right]$$

as the final buffers of the preceding modules are fused with the initial buffers of the modules that follow, except the overall net's external input and output buffers. Thus, the overall number of the combined net's places is

$$\sum_{i=1}^{n_1}(3n_{P_i} + 2) + \sum_{j=1}^{n_2}\left[\frac{n_{A_{j1}}}{2} + \frac{5n_{A_{j2}}}{2} + 2\right] + \sum_{k=1}^{n_3}\left[\frac{5n_{D_{k1}}}{2} + \frac{n_{D_{k2}}}{2} + 2\right] + \frac{n_4 + n_5}{2}.$$

From these, $\sum_{i=1}^{n_1}(n_{Pi}+2)+\sum_{j=1}^{n_2}(n_{A_{j2}}+2)+\sum_{k=1}^{n_3}(n_{D_{k1}}+2)$ are discrete (since no fusion of discrete places occurs), and the remaining

$$\sum_{i=1}^{n_1}(2n_{P_i})+\sum_{j=1}^{n_2}\left[\frac{n_{A_{j1}}}{2}+\frac{3n_{A_{j2}}}{2}\right]+\sum_{k=1}^{n_3}\left[\frac{3n_{D_{k1}}}{2}+\frac{n_{D_{k2}}}{2}\right]+\frac{n_4+n_5}{2}$$

are continuous.

In the multiassembly module, multiple initial parts produce one final product, but the number of nonvalid assembly types q_i must also be taken into account. So, from n_{Ai1} initial buffers, n_{Ai2} final ones arise. In a disassembly module, the number of output buffers increases compared to the number of input buffers and from n_{Di1} initial buffers, n_{Di2} final ones arise. After this, it is possible to compute n_5 with respect to n_4 and the number of modules' input and output buffers are as follows:

$$n_5 = n_4+\sum_{i=1}^{n_3}(n_{Di2}-n_{Di1})+\sum_{j=1}^{n_2}(n_{Aj2}-n_{Aj1}).$$

From the above calculations, it is obvious that the number of net's transitions is equal to the sum of transitions of the individual modules. The respective places' number is slightly reduced due to the fusion of places, but it is also calculated as a function of the numbers of places of the used subsystems.

C. Synthesis Procedure

The synthesis procedure is presented through the modeling of a simple multiproductive production system, shown in Fig. 5. Generalizations are obvious.

The multioperational production system is composed of two multiproductive machine modules, each performing two processes (in different time intervals and not in parallel). Two final products (type 1 and type 2 products) are manufactured from two different initial parts, each receiving two operations, one in each machine. Type 1 parts enter first machine 1 and then machine 2, while type 2 parts follow the opposite route. From Fig. 5, it is obvious that in the overall HTPN model, two place fusions occur (in individual modules connection points). Specifically, places p_6 and p_{20} are fused to form p_{6-20} while places p_9 and p_{15} form p_{9-15}. In modules connection points, the output buffer of each preceding module and the input buffer of the succeeding module are fused to form a common buffer holding in process parts. This results in the reduction of the overall number of continuous places by two, while the respective number of transitions is equal to the sum of module transitions. The combined HTPN model input places are reduced by two (fused places are not external since they have preceding places from which they receive in process parts).

Maximum capacities refer to the maximum number of tokens that can be theoretically found in a place, although they may never be found in a net's operational phase. Maximum capacities of fused places are defined with respect to the capacities of the individual places from which they arise. For example, for

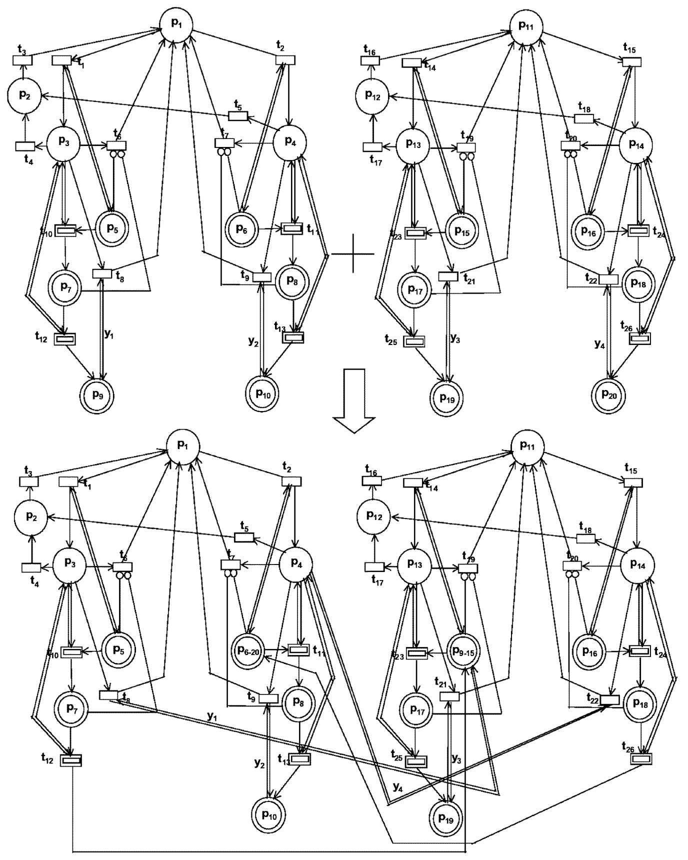

Fig. 5. Generic multiproductive machine modules synthesis [27].

$p_{6-20}, C_{6-20} = \min\{C_6, C_{20}\}$ or $C_{6-20} = \max\{C_6, C_{20}\}$, or $C_{6-20} = a*C_6+(1-a)*C_{20}, 0 < a < 1)$. The fusion of places according to the sequence in which parts receive processes is done before assigning tokens (defining initial marking) in the places. In any other case, the initial marking of the fused place may be derived as the sum of tokens of the places from which the fused one results (e.g., $m_0(p_{6-20}) = m_0(p_6) + m_0(p_{20})$).

The properties of the individual modules are preserved in the overall system after their connections. This holds for the majority of the possible topologies of production systems obtained by connecting the fundamental modules. This can be detected accordingly by observation and verified by simulation using the appropriate analytical tools. For the system of Fig. 5, four linear-independent P-invariants exist, which refer to mutually exclusive machine states (d_1) : $m(p_1) + m(p_2) + m(p_3) + m(p_4) = 1$ and $m(p_{11}) + m(p_{12}) + m(p_{13}) + m(p_{14}) = 1$. The other two (d_2) refer to the parts number preservation in the HTPN $m(p_5)+m(p_7)+m(p_{9-15})+m(p_{17})+m(p_{19}) = n_1$, where n_1 is the initial sum of tokens in places $p_5, p_7, p_{9-15}, p_{17}$ and p_{19} and $m(p_{16})+m(p_{18})+m(p_{6-20})+m(p_8)+m(p_{10}) = n_2$, where n_2 is the initial sum of parts in $p_{16}, p_{18}, p_{6-20}, p_8$, and p_{10}. It is obvious that these P-invariants of the overall model arise by "adding" the respective invariants of the two modules $(m(p_5) + m(p_7) + m(p_9) = n_{1,1}$ and $m(p_{15}) + m(p_{17}) + m(p_{19}) = n_{2,1}, m(p_6)+m(p_8)+m(p_{10}) = n_{1,2}$ and $m(p_{16})+m(p_{18})+m(p_{20}) = n_{2,2})$ and by substituting places p_9 and p_{15} with the fused place p_{9-15} in the first and places p_6 and p_{20} with the fused place p_{6-20} in the second.

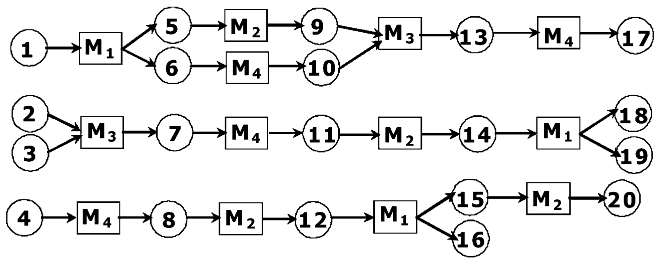

Fig. 6. Multioperational production system and its module decomposition.

D. Generalizations

A random topology system may be derived in terms of individual modules, adjusted to specific features. However, the common basic synthesis rules that may be followed for any multiproductive system are 1) module connections are formed between module buffers. The output buffer of the preceding module and the input buffer of the succeeding module are fused to form a common buffer holding in process parts. 2) Node numbers are calculated from the corresponding modules and from the number of system output buffers. The latter term is necessary to compute the number of fused places in connection points. 3) Fused place markings and capacities are calculated with respect to the corresponding ones of the initial places participating in fusion. 4) Invariants corresponding to part preservation are obtained from the corresponding invariants of the individual modules by appropriate conjunction and replacement of the initial places with the fused ones. 5) System invariants referring to the mutually exclusive states of the machines are derived from the individual module ones. 6) Each module is connected at the most with $n_4 + n_5$ other modules and they have the same maximum number of fused places. 7) The module discrete parts do not participate in any active way in the synthesis procedure.

V. Case Study and Simulation Results

The production system of Fig. 6 composed of four machines and 20 buffers (4 input, 5 output, and 11 internal buffers) is used as a case study. Five types of final products, stored in output buffers 16–20, are produced. Each machine allocates a percentage of its operational time for each product. Initial parts in buffers 2 and 3 follow the same route; two final products result from each disassembly operation in machine M_1. Each machine performs at the minimum two and at the maximum four types of processes in different time intervals. In total, 12 types of processes are performed in the system. Each initial part receives at most four different processes to become a final product. Fig. 7 shows the corresponding overall system HTPN model.

The overall HTPN model consists of two multiproductive machines (M_2 and M_4), one multiassembly (M_3), and one multidisassembly (M_1). In a multiassembly machine module, four types of raw materials are used. Two types of initial parts participate in each assembly but none of them are common in two. All other combinations of raw materials lead to nonvalid assemblies and the respective products are not considered.

Parts enter the system through the input buffers 1, 2, 3, and 4 represented by places p_6, p_{24}, p_{25}, and p_{39}, respectively. Final products in output buffers 16–20 are represented by places $p_{14}, p_{15}, p_{17}, p_{45}$, and p_{65}. All other buffers contain in-process parts.

The overall system HTPN model consists of 54 places (sum of modules places–fused places $= 65 - 11 = 54$) and 82 transitions. Twelve P-invariants exist, four referring to mutually exclusive machine states (d_1) and eight referring to part preservation in sets of places (d_2). Knowing that $k_i, \mathrm{i} = 1, \ldots, 8$ refers to the sum of tokens in respective sets of places given m_0, the invariants are

$$\begin{aligned}
&m(p_1) + m(p_2) + m(p_3) + m(p_4) + m(p_5) = 1\\
&m(p_{18}) + m(p_{19}) + m(p_{20}) + m(p_{21}) = 1\\
&m(p_{30}) + m(p_{31}) + m(p_{32})\\
&\quad + m(p_{33}) + m(p_{34}) + m(p_{35}) = 1\\
&m(p_{48}) + m(p_{49}) + m(p_{50})\\
&\quad + m(p_{51}) + m(p_{52}) + m(p_{53}) = 1\\
&m(p_6) + m(p_9) + m(p_{13-36})\\
&\quad + m(p_{40}) + m(p_{22-44}) + m(p_{26})\\
&\quad + m(p_{28-37}) + m(p_{41}) + m(p_{45}) = k_1\\
&m(p_6) + m(p_9) + m(p_{12-54}) + m(p_{58})\\
&\quad + m(p_{23-62}) + m(p_{26})\\
&\quad + m(p_{28-37}) + m(p_{41}) + m(p_{45}) = k_2\\
&m(p_{24}) + m(p_{27}) + m(p_{29-38})\\
&\quad + m(p_{42}) + m(p_{46-55})\\
&\quad + m(p_{59}) + m(p_{7-63}) + m(p_{10}) + m(p_{14}) = k_3\\
&m(p_{24}) + m(p_{27}) + m(p_{29-38})\\
&\quad + m(p_{42}) + m(p_{46-55}) + m(p_{59})\\
&\quad + m(p_{7-63}) + m(p_{10}) + m(p_{15}) = k_4\\
&m(p_{25}) + m(p_{27}) + m(p_{29-38})\\
&\quad + m(p_{42}) + m(p_{46-55})\\
&\quad + m(p_{59}) + m(p_{7-63}) + m(p_{10}) + m(p_{14}) = k_5\\
&m(p_{25}) + m(p_{27}) + m(p_{29-38})\\
&\quad + m(p_{42}) + m(p_{46-55})\\
&\quad + m(p_{59}) + m(p_{7-63}) + m(p_{10}) + m(p_{15}) = k_6\\
&m(p_{39}) + m(p_{43}) + m(p_{47-56})\\
&\quad + m(p_{60}) + m(p_{8-64})\\
&\quad + m(p_{11}) + m(p_{17}) = k_7\\
&m(p_{39}) + m(p_{43}) + m(p_{47-56})\\
&\quad + m(p_{60}) + m(p_{8-64}) + m(p_{11})\\
&\quad + m(p_{16-57}) + m(p_{61}) + m(p_{65}) = k_8.
\end{aligned}$$

For any finite marking m_0, the HTPN model of the overall system is not generally live, is k-bounded, not conservative, nonpersistent, not repetitive, and not consistent. These properties arise from the fundamental modules that have been used to construct the overall systems HTPN model.

As stated, multioperational production systems under study are typically high-volume systems where machines perform a variety of processes with short process times and times to move processed parts from machines. The case considered describes the production of 100 items of each of the five final product

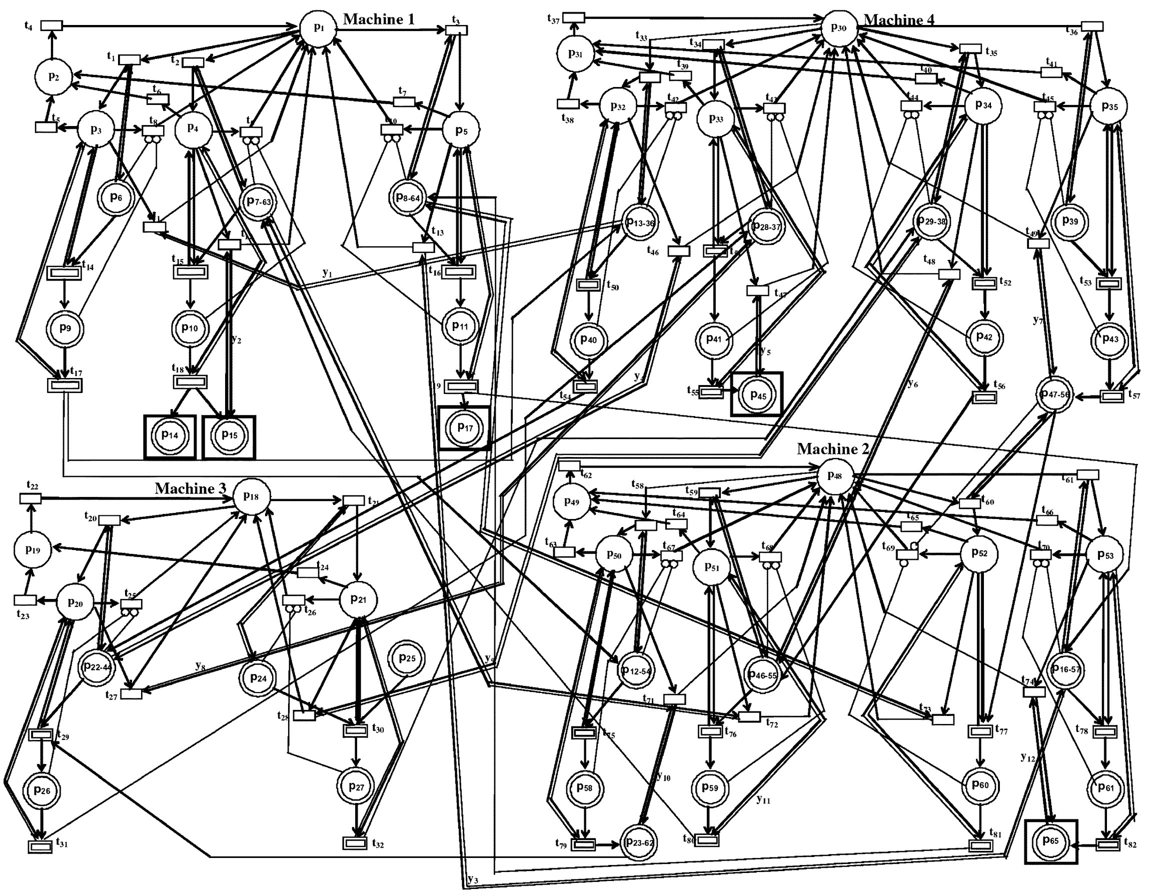

Fig. 7. Multioperational production system HTPN module.

types. This means that each machine considered processes from 200 (multiassembly) to 400 parts (both multiproductive machine modules) of different types in different time intervals.

In the net's initial marking m_0, there are 100 parts in each of the continuous places p_6, p_{24}, p_{25}, and p_{39} that represent the net's initial buffers. Initially, all machines are considered operational and ready to produce. This means that places p_1, p_{18}, p_{30}, and p_{48} contain one token each in m_0. The initial marking of all other places is equal to zero. The net's operational phase is completed when places representing the final buffers $p_{14}, p_{15}, p_{17}, p_{45}$, and p_{65} contain 100 tokens each. In the final net state m_f, places p_1, p_{18}, p_{30}, and p_{48} contain one token each (machines are operational and ready to produce but do not have raw materials and so remain idle) while all other discrete and continuous places are empty.

The system's performance is studied through simulation. Simulations are performed using Visual Object Net software [20]. Some of the possible goals of the simulation may be the optimization of the system's performance through minimizing the machines idleness, maximizing total throughput, minimizing total working time, optimizing buffer capacities, or even finding and replacing noneffective systems components. This can be accomplished by simulating the systems' behavior in alternative scenarios and selecting the best solution. For systems operation, it is necessary to define additionally the firing speeds of the net's continuous and discrete transitions. Firing speeds of all the continuous transitions are considered constant. The delays associated with the discrete transitions are also considered constant, with the only exception being the transitions representing machine breakdowns and breakdown repairs, which are considered to follow normal distributions with given features, as breakdowns and repairs cannot be considered to have always the same duration. In some cases, it may be necessary to define priorities to resolve conflicts in an efficient way. In addition, the maximum capacities of the net's buffers containing inprocess parts must be defined.

Firing speeds of continuous transitions representing parts process speeds are $\{t_{14}, t_{15}, t_{16}, t_{29}, t_{30}, t_{50}, t_{52}, t_{53}, t_{75}, t_{76}, t_{77}, t_{78}\} = \{1.5, 1.3, 1.2, 1, 1.3, 1.9, 1.1, 2, 1.5, 0.8, 1.5, 1.3, 2\}$ parts/time unit. Firing speeds of transitions representing the move of processed parts from machines are $\{t_{17}, t_{18}, t_{19}, t_{31}, t_{32}, t_{54}, t_{55}, t_{56}, t_{57}, t_{79}, t_{80}, t_{81}, t_{82}\} = \{2, 2, 2.5, 2.5, 2.5, 2, 2, 2, 2, 1.5, 1.5, 1.5, 1.5\}$ parts/time unit.

Discrete transitions that refer to machine repair follow normal distributions (normally distributed random numbers between zero and a maximum number x defined here $[\mathrm{rnd}(x)]$, are generated describing the duration of breakdown repairs. So, $\{t_1, t_{22}, t_{37}, t_{62}\} = \{\mathrm{rnd}(5), \mathrm{rnd}(6), \mathrm{rnd}(5), \mathrm{rnd}(7)\}$. The respective stands for the transitions referring to machine breakdown while producing each product type. Each machine duration between breakdowns is considered equal since the probability of a breakdown while processing a part in a machine is the same for all part types. These are $\{t_5, t_6, t_7, t_{23}, t_{24}, t_{38}, t_{39}, t_{40}, t_{41}, t_{63}, t_{64}, t_{65}, t_{66}\} = \{\mathrm{rnd}(20), \mathrm{rnd}(20), \mathrm{rnd}(20), \mathrm{rnd}(22), \mathrm{rnd}(22), \mathrm{rnd}(20), \mathrm{rnd}(20), \mathrm{rnd}(20), \mathrm{rnd}(20), \mathrm{rnd}(18), \mathrm{rnd}(18),$

$\mathrm{rnd}(18), \mathrm{rnd}(18)\}$. All other discrete transitions are immediate since their occurrence must be immediate when all of the necessary preconditions (full-machine product buffers or empty initial buffers) are satisfied.

The maximum capacity of the buffers containing in-process parts (5–15) must be defined before beginning the simulations. It is considered that all of these buffers (represented by $p_{7-63}, p_{8-64}, p_{13-36}$, $p_{28-37}, p_{29-38}, p_{47-56}$, $p_{22-44}, p_{12-54}, p_{46-55}$, p_{16-57}, and p_{23-62}) may contain up to ten parts during the net's operation. The same number is the weight of the couples of arcs connecting each machine's output buffers with transitions representing full final buffers and leading to the change of performed processes (e.g., t_{11}).

After the definition of these parameters, the simulation can begin. It must be noted that the final marking of the PN is common for all cases as the machines' operational parameters and features that have an impact in product cycle time (parts process speeds, internal buffer maximum capacities, mean breakdown durations, etc.) change and not the number of final products.

With the given values of the parameters, the simulation is terminated after 465.4 time units. Machine 1 completes parts processes after 465.4 time units, machine 2 after 464, machine 3 after 462, and machine 4 after 462.7 time units. Type 1 products production is completed after 442.3 time units, type 2 and 3 items production after 465.4 time units, type 4 products production after 310 time units, and type 5 products production after 280 time units. From Table IV, it is obvious that all internal buffers reach their maximum capacities instantly or for longer time periods. The time interval and duration for which the maximum capacity of each buffer is reached can be used for conclusions regarding systems operation and changes that are necessary to improve systems performance. Also, interesting conclusions may arise from studying machine idleness.

A modification is attempted by increasing the firing speed of t_{75} from 0.8 to 1.5 parts/time unit. Change of a transition's firing speed represents changes in the operational speed of a machine. This may result from the change of the machine tools or by substituting the machine with a more efficient one. Interpretation of the changes is to make clear the interaction between subsystems and that by changing a module's features results in changes of the overall net's operation. Simulation is repeated with all other parameters being the same and is completed in 419.9 time units; a significant reduction of overall production time by about 11%. In this simulation, machine 1 completes the parts processes after 418.9 time units, machine 2 after 419.8 time units, machine 3 after 380.5 time units, and machine 4 after 419.8 time units. Type 1 items production is completed after 348.8 time units, type 2 and 3 items production after 397.2 time units, type 4 items production after 419.8 time units, and type 5 after 418.9 time units. Once again, all of the net's internal buffers reach their maximum capacity for longer or shorter time intervals.

A next modification step is attempted by doubling the speed of t_{29} from 1 to 2 parts/time unit. In this case, simulation is terminated after 382.9 time units, meaning that there is an extra reduction of the simulation time by 8.6% compared to the previous one. In this third simulation, machine 1 completes parts processes after 382.8 time units, machine 2 after 381.7 time units, machine 3 after 380.7 time units, and machine 4 after 361.7 time units. Type 1 items production is completed after 361.8 time units, type 2 and 3 items production after 382.8 time units, type 4 products production after 379.7 time units, and type 5 products production after 374.1 time units.

Another modification step concerns changing the time between machine breakdown appearances of machine M_2. In this case, the breakdown appearances of machine 2 are again normally distributed between 0 and 30. That is, the time between firings of the respective transitions are $\{t_{63}, t_{64}, t_{65}, t_{66}\} = \{\mathrm{rnd}(30), \mathrm{rnd}(30), \mathrm{rnd}(30), \text{ and } \mathrm{rnd}(30)\}$. The changes performed ensure that the appearance of breakdowns is rarer than before (this can be achieved by regular maintenance of the equipment when, for example, it is idle to avoid possible breakdowns). In this case, simulation is completed in 358.7 time units, meaning that there is an additional reduction of production time of almost 6.32%. It must be noted that the overall reduction of production time with all of the described changes is almost 23%. In this simulation, machine 1 completes all of the parts processes that perform after 335.2 time units, machine

TABLE IV
CALCULATION OF QUANTITATIVE FEATURES OF MULTIOPERATIONAL PRODUCTION SYSTEM FOR THE DIFFERENT SIMULATIONS DESCRIBED

PARAMETER	**SIMULATION**			
	Initial	**Second**	**Third**	**Fourth**
Duration				
In time units	465.4	419.8	382.9	358.7
Mean production time (time units)				
Type 1 products	4.423	3.488	3.618	3.587
Type 2 products	4.654	3.972	3.828	3.352
Type 3 products	4.654	3.972	3.828	3.352
Type 4 products	3.1	4.198	3.797	2.737
Type 5 products	2.8	4.189	3.741	2.66
Maximum number of parts in buffer i, i = 5 – 15				
Buffer 5 (p_{12-54})	10	10	10	10
Buffer 6 (p_{13-36})	10	10	10	10
Buffer 7 (p_{29-38})	10	10	10	10
Buffer 8 (p_{47-56})	10	10	10	10
Buffer 9 (p_{23-62})	10	10	10	10
Buffer 10 (p_{22-44})	10	10	10	10
Buffer 11 (p_{46-55})	10	10	10	10
Buffer 12 (p_{8-64})	10	10	10	10
Buffer 13 (p_{28-37})	10	10	10	10
Buffer 14 (p_{7-63})	10	10	10	10
Buffer 15 (p_{16-57})	10	10	10	10
% of the simulation time that buffer i is full, i = 5 – 15				
Buffer 5 (p_{12-54})	59.1	26.6	35.28	41.65
Buffer 6 (p_{13-36})	6.27	22.75	22.51	33.43
Buffer 7 (p_{29-38})	34.33	50.69	41.68	52.05
Buffer 8 (p_{47-56})	44.97	51.11	35.75	15.86
Buffer 9 (p_{23-62})	3.1	12.34	9.01	22.14
Buffer 10 (p_{22-44})	45.94	13.84	16.95	21.27
Buffer 11 (p_{46-55})	43.88	40.64	5.51	19.24
Buffer 12 (p_{8-64})	4.34	15.97	0.08	6.97
Buffer 13 (p_{28-37})	1.12	3.76	15.8	20.23
Buffer 14 (p_{7-63})	0.13	9.72	2.27	1
Buffer 15 (p_{16-57})	15.94	25.87	11.12	13.02

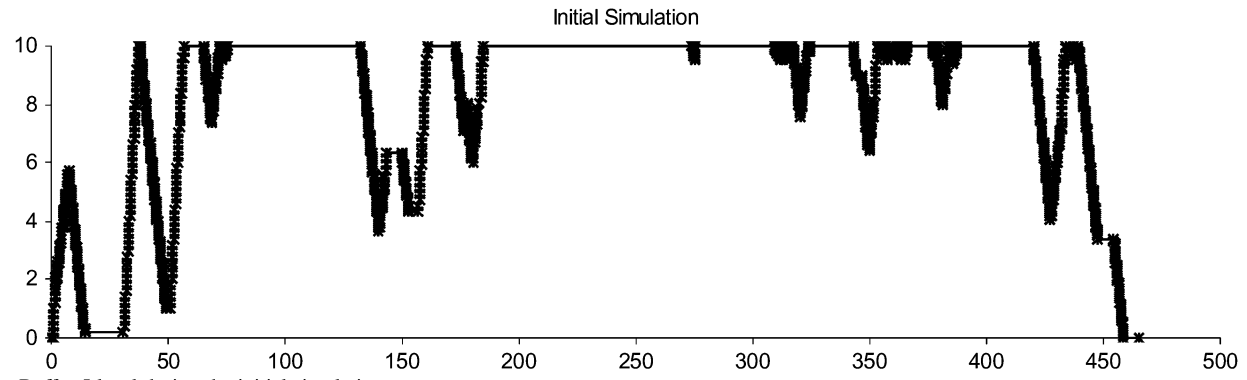

Fig. 8. Buffer 5 level during the initial simulation.

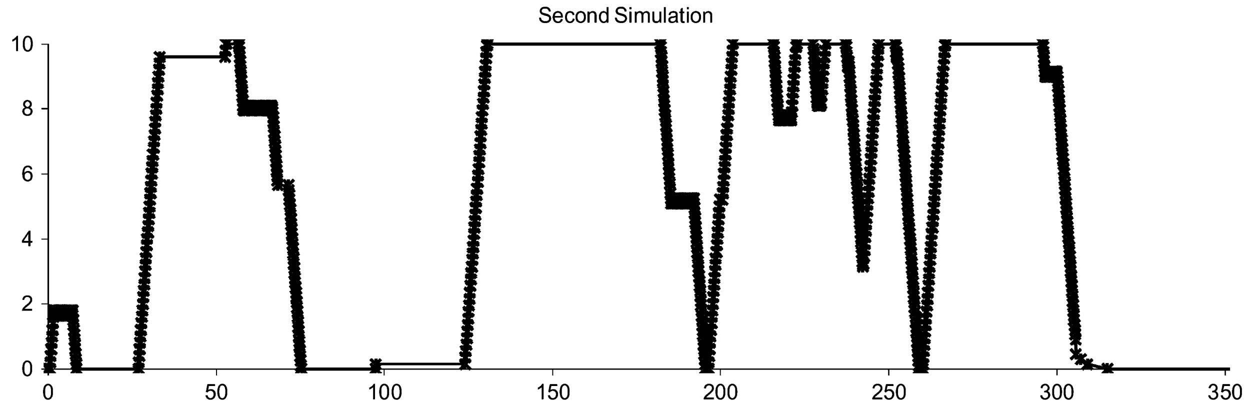

Fig. 9. Buffer 5 level during the second simulation.

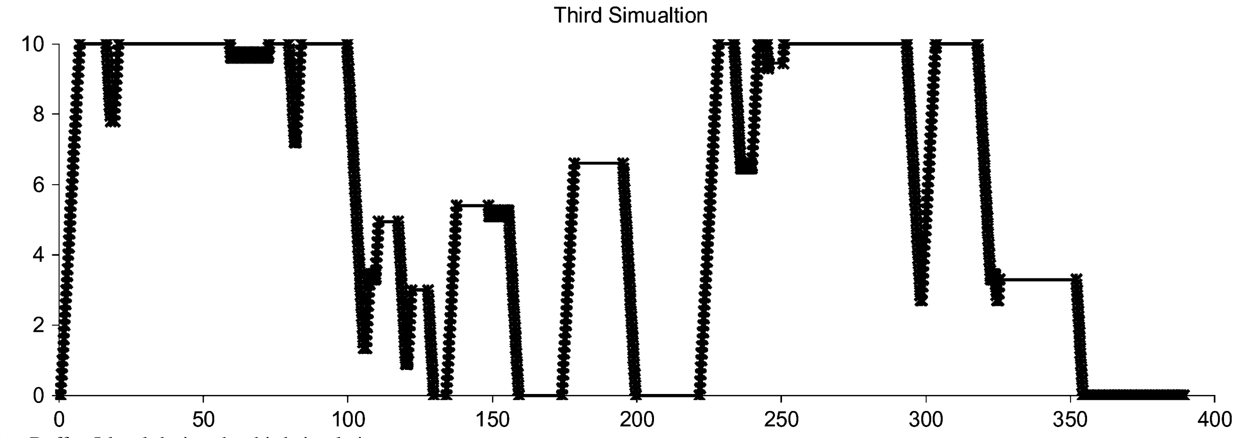

Fig. 10. Buffer 5 level during the third simulation.

2 after 332.7 time units, machine 3 after 354.6 time units, and machine 4 after 358.6 time units. Type 1 products production is completed after 358.7 time units, type 2 and 3 items production after 335.2 time units, type 4 products production after 273.7 time units, and type 5 products production after 266 time units.

Simulations may be continued until the optimization of a given objective function (minimization of overall simulation time, minimization of machine idleness, maximization of throughput, etc.). The impact of other factors in the overall simulation time may be also studied, such as assigning different priorities in the operations performed in a machine, control strategies followed, and change of transport times. An overview of the calculated quantitative parameters and features for the described simulations of multioperational production system is presented in Table IV. From this, it is obvious that the mean production times of the five types of products change according to the changes performed in the net's features. In some cases (e.g., type 5 product), changes that result in reduction of the overall simulation duration may result in an increase of a product's mean production time, since the sequences in which processes are performed are different because of the changes in the net's operations. Also, it is interesting to note that in all cases, all internal buffers reach their maximum capacities (ten parts) instantly or for periods that may overcome even 50% of the net's overall operation time. The calculation of the percentage of the simulation time that each buffer is full is critical for possible modifications of the buffers capacities that may result in further improvement of the net's performance.

In Figs. 8–11, buffer 5 levels during the performed simulations are presented. The respective diagrams may be produced

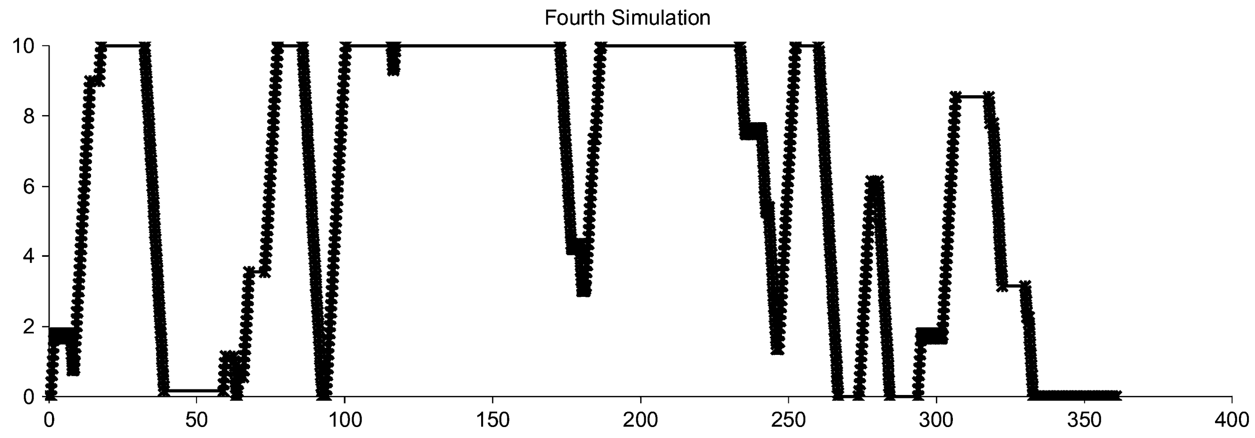

Fig. 11. Buffer 5 level during the fourth (final) simulation.

for all other in-process parts buffers but are omitted here due to space limitations. Because of the continuous transitions firing speed and buffer levels that are considered real numbers, in certain simulation time, instants buffer 5 is shown to contain a noninteger number of parts. This is not possible in real systems, but is a common drawback of discrete event dynamic systems (DEDS) fluid approximations. However, it does not significantly influence the results since the evaluation parameters can be calculated with accuracy (only in some cases it is necessary to take the closest integer number).

VI. Discussion

The methodology proposed in this paper is suggested for the modeling and study of random topology and complexity production systems. This means that even for small markings, the number of reachable states is high (explodes) and the OPN analysis tools such as reachability graphs are hard to derive and do not have practical value. This combined with the fact that the overall systems markings in the considered example are not typical for real applications and much larger markings may be met in practice, strengthen the necessity to use HPNs for modeling and analysis purposes. The already presented advantages of HPNs make them the best selection.

Calculations of systems P-invariants and nodes numbers are not trivial, since we are not familiar with any papers, in which nodes and P-invariants complexities of HPN models built from basic subsystems are calculated just by knowing how many modules are used and what the operational features of each module are. No reduction is used for a modules connection, since reduction techniques result in loss of detail. Even the implemented fundamental subsystems are complicated since they refer to nondedicated production systems that produce different items at different time intervals. The used subsystems are as general as possible (in a number of papers, the simplest possible models are considered) and no restrictions are made in their structural features (e.g., how many different types of processes may be implemented by a machine).

The calculated nodes and P-invariants are necessary for the complexity analysis of the HTPN models of systems composed of the fundamental subsystems. P-invariants express a notion of token conservation in sets of places (e.g., buffers) for all reachable markings without enumeration of the reachability set $R(m_0)$ and independently of any dynamic process. In order to calculate the P-invariants, we must know the number and type of places that compose the discrete (P-invariants describe mutually exclusive machine states) and the continuous (P-invariants describe preservation of parts) part of the net.

These calculations are also necessary for future research topics. One such step concerns the implementation of algorithms for the automatic construction of the HTPN models of random topology and structure. These algorithms will also calculate the numbers as well as the form of the overall systems P-invariants by knowing its topology, number of machines used, process sequences, and the structural features of each machine. The theoretical calculation of the nodes number in this case will be used for naming the overall systems nodes and for checking the correct functioning of the algorithms. Another possible future research topic concerns the extension of the proposed method in order to create a well-defined framework for modeling and studying different types of manufacturing flexibility (operation, routing, etc.).

Calculated P-invariants are also appropriate for the modular supervisory control of the overall system, which will be used to optimize its behavior and performance according to a given objective and ensure its deadlock freeness and liveness as long as tokens exist in the net's discrete part. This is a common approach since a number of authors have used place invariants to compute a feedback [49] or supervisory [50] controller.

A drawback of the proposed method is the inherent complexity of the HTPN models (e.g., Fig. 7). However, we decided not to use any reduction techniques to simplify the HTPN modeling since this would have resulted in a loss of significant detail. An alternative which would have simplified the representation is the use of colored Petri nets that under certain conditions result in a more compact model (however, in this case, the operation of continuous model elements has to be considered discrete). Some implementation problems arise from the Visual Object Net simulation package, since its capabilities are limited and its speed decreases significantly as net complexity increases. In a next phase, it will be necessary to develop a simulation tool with improved features (e.g., variety of distributions followed, included analysis tools, data exchange with other programs, a more friendly user interface, and an automatic calculation of performance measures).

The suggested approach is considerably different from other approaches where HPNs and their variations have been used to model manufacturing systems, the most important of which have been already presented. The differences are demonstrated by direct comparison with the relevant literature in the sequel.

The goals of the work reported in [40] differ significantly from the ones in this manuscript since FOHPNs are used there only to model the system under study, while the control and scheduling of the system's operation is done using LPP methods. Also, the models are nondetailed compared to the ones introduced here and no properties, P-invariants, and nodes complexities are calculated. The main differences described for [40] also stand for [43]. In addition, no analytical modular approach is considered and the subsystems modules are built in a completely different logic since discrete nodes represent only breakdowns. The main differences of the method of [38] from the one proposed here are no properties, nodes complexity, and P-invariants are calculated; no modules connection procedure is suggested; the manufacturing systems under study are of a specific type; and no alternative simulations are performed to check how a model's performance changes according to the implemented changes. Also, a small number of performance measures are calculated and the modules are not the general but the simplest possible cases. The method of [21] uses HPNs only as a modeling tool, as its main goal is to introduce an algorithm for constructing the system automaton based on the respective HPN used for the computation of specific performance measures. No properties, nodes complexity, and P-invariants are calculated; no modules connection procedure is suggested, and the proposed approach is not modular. The presented example concerns a dedicated and non multioperational system. In [36], no analysis, properties, and nodes complexity calculation is performed, while a different approach for manufacturing systems is followed. Even the main features of the used PNs classes are significantly different.

In [32], the authors claim that a lot of things have to be done to extend this method for transient analysis of more complicated systems and that they have to do a properties analysis as well. Also, P-invariants and nodes complexities are not calculated and the representation of the manufacturing systems is the simplest possible with major assumptions concerning the net's features. Transient analysis is used similarly to simulation. In [47], no properties and invariants calculation is done or simulations are performed since the only goal of this paper is to study the different possible feedback control designs. The manufacturing systems representation is the simplest possible since machines are represented by transitions and buffers by places, and no breakdowns or other facts take place in the system. In [48], the author claims that his main contribution is to work out the firing frequencies of the transitions by reversing the evolution equations of the model when the marking is known. The differences between this paper and our method are the same as those presented above for [47].

VII. Conclusion

HTPNs have been used for modeling, analysis, synthesis, and performance evaluation of random topology multioperational production systems. Three fundamental multioperational production modules are considered, their HTPN models are obtained, and their P-invariants and properties are calculated. Calculation of the overall HTPN model nodes and derivation of the HTPN model invariants are done theoretically without considering a specific topology system. Overall system nodes numbers, properties, and P-invariants are obtained with respect to the respective quantities of the individual modules, according to the modules used, and the topology and structure of the overall system. The proposed methodology is complete and general covering the aspects of system decomposition/composition, constraint satisfaction, complexity, and performance evaluation with minimum initial assumptions regardless of considered system topology. The proposed methodology may be used for the analysis of multioperational production systems behavior and optimization of systems performance measures under specific conditions.

References

[1] R. David and H. Alla, "On hybrid Petri nets," *Discrete Event Dyn. Syst.: Theory Appl.*, vol. 11, pp. 9–40, Jan. 2001.

[2] G. Frey, K. Mossig, and M. Schnabel, "Assembly line sequencing based on Petri-net T-invariants," *Contr. Eng. Practice*, vol. 8, pp. 63–69, Jan. 2000.

[3] M. Domg and F. F. Chen, "Process modeling and analysis of manufacturing supply chain networks using object-oriented Petri nets," *Robot. Comput. Integr. Manuf.*, vol. 17, pp. 121–129, 2001.

[4] J. Wern-Kueir, "Petri net models applied to analyze automatic sequential pressing systems," *J. Mater. Process. Technol.*, vol. 120, pp. 115–125, 2002.

[5] M. D. Jeng and X. Xie, "Modeling and analysis of semiconductor manufacturing systems with degraded behavior using Petri nets and siphons," *IEEE Trans. Robot. Automat.*, vol. 17, no. 5, pp. 576–588, Oct. 2001.

[6] K. Valavanis, "On the hierarchical modeling analysis and simulation of flexible manufacturing systems with Extended Petri nets," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 1, pp. 94–110, Jan./Feb. 1990.

[7] A. A. Desrochers and R. Al-Jaar, *Applications of Petri Nets in Manufacturing Systems—Modeling, Control, and Performance Analysis*. Piscataway, NJ: IEEE Press, 1995.

[8] J. Moody and P. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*. Norwell, MA: Kluwer, 1998.

[9] R. David and H. Alla, *Petri Nets & Grafcet—Tools for Modeling Discrete Event Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[10] J. M. Proth and X. Xiaolan, *Petri Nets, A Tool for Design and Management of Manufacturing Systems*. New York: Wiley, 1996.

[11] M. C. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Norwell, MA: Kluwer, 1993.

[12] T. Murata, "Petri Nets: Properties, analysis, and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[13] T. Gu and P. Bahri, "A survey of Petri net applications in batch processes," *Comput. Ind.*, vol. 47, pp. 99–111, 2002.

[14] R. Valette, B. Chezalviel-Pradin, and F. Girault, "An introduction to Petri net theory," in *Fuzziness Petri Nets, Studies Fuzziness, Soft Computing*. Heidelberg, Germany: Physica Verlag, 1999, vol. 22, pp. 3–24.

[15] N. Tsourveloudis, E. Dretoulakis, and S. Ioannidis, "Fuzzy work-in-process inventory control of unreliable manufacturing systems," *Inform. Sci.*, vol. 27, pp. 69–83, 2000.

[16] S. Christensen and L. Petrucci, "Modular analysis of Petri Nets," *Comput. J.*, vol. 43, no. 3, pp. 224–242, 2000.

[17] S. Ioannidis, N. Tsourveloudis, and K. Valavanis, "Fuzzy supervisory control of manufacturing systems," *IEEE Trans. Robot. Automat.*, vol. 29, no. 3, pp. 379–389, Jun. 2004.

[18] ——, "Supervisory control of multiple-part-type production networks," in *Proc. 10th IEEE Med. Conf. Control Automation*. Lisbon, Portugal, 2002.

[19] M. Svadova, "Modeling hybrid dynamic systems using hybrid Petri nets," in *Proc. 13th Int. Conf. Process Control*. Bratislava, Slovakia, 2001.

[20] Visual Object Net Software Package, v.2.07 a 2002 [Online]. Available: http://www.ParamSoft.de/

[21] A. T. Sava and H. Alla, “Combining hybrid Petri nets and hybrid automata,” *IEEE Trans. Robot. Automat.*, vol. 17, no. 5, pp. 70–678, Oct. 2001.
[22] R. Akella and P. R. Kumar, “Optimal control of production rate in a failure prone manufacturing system,” *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 2, pp. 116–126, Feb. 1986.
[23] S. X. Bai and S. B. Gershwin, “Scheduling manufacturing systems with work-in-process inventory control: Formulation of single part type systems,” in *Proc. 29th IEEE Conf. Decision Control*, 1990, pp. 557–564.
[24] ——, “Scheduling manufacturing systems with work-in-Process inventory control: Multiple-part-type systems,” *Int. J. Prod. Res.*, vol. 32, pp. 365–386, 1994.
[25] G. Tsinarakis, K. Valavanis, and N. Tsourveloudis, “Modular Petri net based modeling, analysis, and synthesis of dedicated production systems,” in *Proc. IEEE Int. Conf. Robotics Automation*, Taipei, Taiwan, R.O.C., 2003, pp. 3559–3564.
[26] G. J. Tsinarakis, N. C. Tsourveloudis, and K. P. Valavanis, “Modular Petri Net based modeling, analysis, synthesis, and performance evaluation of random topology dedicated production systems,” *J. Intell. Manuf.*, vol. 16, pp. 67–92, 2005.
[27] G. J. Tsinarakis, K. P. Valavanis, and N. C. Tsourveloudis, “Studying multi-operational production systems with modular hybrid Petri nets,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Las Vegas, NV, 2003, pp. 3028–3034.
[28] S. B. Gershwin, *Manufacturing Systems Engineering*. Upper Saddle River, NJ: Prentice-Hall, 1994.
[29] ——, “Design and operation of manufacturing systems: The control-point policy,” *Inst. Elect. Eng. Trans.*, vol. 32, pp. 891–906, 2000.
[30] J. Le Bail, H. Alla, and R. David, “Hybrid Petri nets,” in *Proc. Eur. Control Conf.*, Grenoble, France, Jul. 1991, pp. 1472–1477.
[31] H. Alla and R. David, “A modeling and analysis tool for discrete events systems: Continuous Petri net,” *Perf. Eval.*, vol. 33, pp. 175–199, 1998.
[32] N. Zerhouni, M. Ferney, and A. El Moudni, “Transient analysis of manufacturing systems using continuous Petri nets,” *Math. Comput. Simul.*, vol. 39, pp. 635–639, 1995.
[33] V. Duridanova and T. Hummel, “Modeling of embedded mechatronic systems using hybrid Petri nets,” in *Proc. IASTED Int. Conf. Modeling, Identification, Control*, Innsbruck, Austria, 2001, pp. 521–526.
[34] G. Horton and M. Kowarschik, “Discrete-continuous modeling using hybrid stochastic Petri nets,” in *Proc. Eur. Simulation Symp.*, 1999.
[35] L. Thevenon and J. M. Flaus, “Modular representation of complex hybrid systems: Application to the simulation of batch processes,” *Simul. Pract. Theory*, vol. 8, no. 5, pp. 283–306, 2000.
[36] R. Drath, U. Engmann, and S. Schwuchow, “Hybrid aspects of modeling manufacturing systems using modified Petri nets,” presented at the 5th Workshop Intelligent Manufacturing Systems, Granado, Brazil, 1999.
[37] R. Drath, “Hybrid object nets: An object oriented concept for modeling hybrid systems,” in *Proc. Hybrid Dynamical Systems. 3rd Int. Conf. Automation Mixed Processes*, Reims, France, 1998, pp. 437–442.
[38] M. Allam and H. Alla, “Modeling and simulation of an electronic component manufacturing system using hybrid Petri nets,” *IEEE Trans. Semicond. Manuf.*, vol. 11, no. 3, pp. 374–383, Aug. 1998.
[39] A. Di Febbraro, A. Giua, and G. Menga, “Guest editorial for the special issue on hybrid Petri nets,” *Discr. Event Dyn. Syst.*, vol. 11, no. 1&2, 2001.
[40] F. Balduzzi, A. Giua, and G. Menga, “First-order hybrid Petri Nets: A model for optimization and control,” *IEEE Trans. Robot. Automat.*, vol. 16, no. 4, pp. 382–399, Aug. 2000.
[41] P. Antsaklis, X. Koutsoukos, and J. Zaytoon, “On hybrid control of complex systems: A survey,” in *Proc. 3rd Int. Conf. Automation Mixed Processes: Dynamic Hybrid Systems*, Reims, France, 1998, pp. 1–8.
[42] S. Pettersson and B. Lennartson, “Hybrid modeling focused on Hybrid Petri Nets,” in *Proc. 2nd Eur. Workshop on Real-Time Hybrid Systems*, Grenoble, France, 1995, pp. 303–309.
[43] F. Balduzzi, A. Giua, and C. Seatzu, “Modeling and simulation of manufacturing systems with first-order hybrid Petri nets,” *Int. J. Production Res., (Special Issue Modeling, Specification, Analysis of Manufacturing Systems)*, vol. 39, no. 2, pp. 255–282, 2001.
[44] J. Wang, *Timed Petri Nets: Theory and Application*. Norwell, MA: Kluwer, 1998.
[45] T. Hodgson, G. Ge, R. King, and H. Said, “Dynamic lot size/sequencing policies in a multi-product single-machine system,” *Inst. Elect. Eng. Trans.*, vol. 29, pp. 127–137, 1997.
[46] J. Buzacott, “The structure of manufacturing systems: Insights on the impact of variability,” *Int. J. Flex. Manuf. Syst.*, vol. 11, pp. 127–146, 1999.
[47] D. Lefebvre, “Feedback control designs for manufacturing systems modeled by continuous Petri nets,” *Int. J. Syst. Sci.*, vol. 30, no. 6, pp. 591–600, 1999.
[48] ——, “Estimation of the firing frequencies in discrete and continuous Petri nets models,” *Int. J. Syst. Sci.*, vol. 32, no. 11, pp. 1321–1332, 2001.
[49] K. Yamalidou, J. Moody, M. Lemmon, and P. Antsaklis, “Feedback control of Petri nets based on place invariants,” *Automatika*, vol. 32, no. 1, pp. 15–28, 1996.
[50] J. Moody and P. Antsaklis, “Petri net supervisors for DES with uncontrollable and unobservable transitions,” *IEEE Trans. Automat. Contr.*, vol. 45, no. 3, pp. 462–476, Mar. 2000.
[51] R. David and H. Alla, *Discrete, Continuous, and Hybrid Petri Nets*. New York: Springer-Verlag, 2005.

George J. Tsinarakis (S'04) received the B.S. and M.S. degrees in 2000 and 2002, respectively, from the Technical University of Crete, Chania, Greece, where he is currently pursuing the Ph.D. degree in industrial engineering.

His research interests include Petri nets theory, discrete event systems, modeling, analysis, performance evaluation and optimization of production systems, and manufacturing flexibility.

Nikos C. Tsourveloudis (A'95) received the B.S. degree in production engineering and management and the Ph.D. degree in industrial engineering from the Technical University of Crete (TUC), Chania, Greece, in 1990 and 1995, respectively.

Currently, he is an Assistant Professor and Director of the Machine Tools Lab and the Intelligent Systems and Robotics Lab at TUC. From 1992 to 1995, he was President of the Association of Industrial Engineers of Greece. He was a Research Scientist with the University of Louisiana, Lafayette, in 1997 and 1998. His research interests include intelligent control, modeling and scheduling of flexible and computer integrated manufacturing systems, autonomous operation/navigation of robotic vehicles, virtual reality, and fuzzy-logic applications.

Dr. Tsourveloudis is a member of numerous professional and scientific organizations. The American National Science Foundation (NSF), the European Union and the Hellenic General Secretariat for Research and Technology (GSRT) have funded his research.

Kimon P. Valavanis (SM'91) received the Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1986.

He was a Faculty Member at Northeastern University, Boston, MA, holding the Analog Devices Career Development Chair for Assistant Professors from 1986 to 1990. He was also with the University of Louisiana, Lafayette, from 1991 to 1999 serving as Regents Professor in Manufacturing from 1995 to 1999 and as A-CM associate Director for Research. He was with the Technical University of Crete, Chania, Greece from 1999 to 2003 and was Director of the Graduate Program and Chair of the University Industrial Advisory Board. Since 2003, he has been with the Department of Computer Science and Engineering and the Center for Robot-Assisted Search and Rescue (CRASAR), University of South Florida, Tampa, where he serves as Professor and Deputy Director of CRASAR. He also holds a Guest Professorship at the Faculty of Electrical Engineering and Computing, Department of Telecommunications, University of Zagreb, Zagreb, Croatia. His research is in the area of intelligent systems, robotics, and manufacturing. However, during the last few years, he has concentrated on control and coordination of unmanned systems. He has published more than 250 book chapters, technical journal/transactions, and conference papers, and has co-authored three books.

Dr. Valavanis has been the Editor-In-Chief of the *IEEE Robotics Automation magazine* since 1996. He was the General Chair (with F. Lewis) of the 11th Mediterranean Conference on Control and Automation in 2003 and the Program Chair of the 2004 IEEE ICRA. He is also the General Chair of the 2007 Mediterranean Conference on Control and Automation. In 1998, he was elected Vice President Administration of the IEEE Mediterranean Control Association. He is a former Fulbright Scholar (Senior Lecturing/research Award) and a Distinguished Speaker in the IEEE Robotics and Automation Society (since 2003).