ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 332 (2005) 461–485
Theoretical Computer Science
www.elsevier.com/locate/tcs

Property-preserving subnet reductions for designing manufacturing systems with shared resources[☆]

H.J. Huang[a,*], L. Jiao[b], T.Y. Cheung

[a] Department of Computer Science and Information Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 518055, China

[b] Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

Received 1 January 2004; accepted 13 October 2004
Communicated by D.-Z. Du

Abstract

This paper handles two problems in manufacturing system design: resource sharing and system abstraction. In a manufacturing system, resources such as robots, machines, etc. are shared by several processes. When the resources are switched from one process to another, they may need some modifications such as cleaning oil, adding equipments and so on. Previous designing methods assume that the resources have no intermediate modifications. Hence, they need to be extended to handle such kinds of resource-sharing problems. As for abstraction, modeling operations with single places in manufacturing system design is very popular. From the viewpoint of verification, the objective is to verify whether the reduced model has the same desirable properties as the original one. This paper presents three kinds of property-preserving subnet reduction methods. For each reduction method, conditions are presented for ensuring that the properties liveness, boundedness and reversibility are preserved. Applications of these reduction methods to handling the above resource sharing and system abstraction problems are illustrated with an example from the manufacturing system.

Keywords: Manufacturing system; Petri net; Property-preserving; Reduction; System design; Transformation; Verification

☆ This work was financially supported by National Natural Science Foundation of China under Grants No. 60435020, No. 60473007 and No. 60223005.

* Corresponding author.
E-mail addresses: hjhuang@hitsz.edu.cn (H.J. Huang), ljiao@ios.ac.cn (L. Jiao).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.12.010

1. Introduction

1.1. Motivation and problems

Two problems arise frequently in system design, namely, resource sharing and system abstraction. Every system needs some resources. In software engineering, buffers, data-type libraries, servers, software agents, databases, etc. are examples of computing resources. In manufacturing engineering, a resource may be a robot, a machine, an assembly line, etc. In software coding, a subprogram or an operation may also be regarded as a resource. For various reasons, these resources often have to be shared among several parts of the system. In software engineering, for example, the well-known Mutual Exclusion Problem deals with the issues of how to share the access to some common data resources arising in many practical applications. In manufacturing engineering, in order to reduce the idling time of the expensive robots and machines, their utilization is often shared among several processes [20]. Abstraction plays an important role in system development. In component-based system design, for example, a component is abstracted into a single function. In a programming language, a library is abstracted into a data type. In manufacturing engineering, a set of resources is represented by a single 'super' resource.

Both problems are complex and error-prone. Note that 'sharing' does not imply simultaneous usage. While simultaneous usage can be handled by re-enterable codes in software systems and is not allowed in manufacturing systems, 'sharing' requires a resource to be occupied by some part(s) exclusively during utilization and is released afterwards. For multiple-resource systems, a wrong order in occupying and releasing these resources may cause deadlocks or overflow. As for abstraction, from the viewpoint of verification, the objective is to check whether a system is still valid when every replaced part operates as a single function while ignoring its internal logic. This is not a simple task either, especially if the parts under abstraction form subsystems with multiple entries and exits. A logical mishandling will make an abstraction erroneous.

Based on Petri nets, this paper presents a unified approach for the modeling and verification of these two different problems. Briefly, both problems are modeled as property-preserving subnet-reducing transformations. According to the structure of the shared resources or the replaced parts, three classes of transformations are formulated. For each transformation, conditions are presented for ensuring that properties liveness, boundedness and reversibility are preserved. This approach is summarized in terms of four specification or verification problems as follows:

1. *Modeling the system*—The type of a Petri net used for modeling the system under design not only determines its scope of application but also affects the process of verification [15]. In manufacturing engineering, most of the systems are modeled as finite state machines or marked graphs [33]. In use-case-based software system design, the use cases may be specified as case nets [6]. This paper investigates general Petri nets.
2. *Representing the resources and abstracted parts*—In the literature, a resource is uniquely represented as a place. Zhou's exclusions [33,35] and Chu's augmented marked graphs [9] are formal descriptions of such representations. Also, it is assumed that a resource is switched from one user to another without any intermediate modification. In this paper, it is assumed that the given system is composed of connected or disconnected parts.

(For the sake of flexibility, a part in this paper has no fixed definition.) Each resource is originally represented by a set of places (called *resource-places* hereafter), one in each of the parts it is involved in. Also, a resource may go through some intermediate processing when switching from one user to another. This implies that the resource-places may form a connected subnet whose transitions represent the intermediate processes. As for abstraction, sequential systems are represented as directed paths and non-sequential systems having multiple entries and exits as state-machine subnets.

3. *Formulating resource sharing and subsystem abstraction as subnet-reducing transformations*—In all the models appearing in the literature, a resource is represented uniquely as a place. This paper takes a synthesis approach. When a resource is shared by several parts or a part is abstracted into a single function, its representation (i.e., a subnet) will be merged into a single place or a single transition. Formally, this is a transformation that reduces a subnet to a single place or transition. Three transformations are formulated according to the structure of the shared resources or abstracted parts.
4. *Verifying the system*—To verify a system is to show whether it possesses certain properties or not. For example, the deadlock and overflow issues mentioned above are investigated as the liveness and boundedness properties of the system's Petri net representation. In the literature dealing purely with resource-sharing or system abstraction, rarely any 'specific and systematic' methods for verification have been reported. Most of the time, just general techniques are used. By viewing these two problems as transformations, this paper proposes a property-preserving approach. First, it is assumed that the system possesses certain properties before the transformation. For each of the transformations, conditions are proposed so that it will preserve the system's properties.

1.2. Property-preserving transformations

Petri nets are well known for their graphical and analytical capabilities in specification and verification, especially for concurrent systems. Many properties can be analytically defined and many techniques are available for development and verification. In particular, the approach based on property-preserving transformations will be described in more detail below as it is the main theme of this paper.

Usually, a design may be subject to many transformations, such as compositions, refinements, place-reductions, etc. A transformation may be used for system generation or system verification. For the former, a transformation creates a needed and 'permanent' modification on a design. For the latter, a transformation is purely temporary so that verification may proceed more easily under the transformed specification. Naturally, for both purposes, it is important that a transformation should not destroy or create those properties under investigation.

Some relevant issues concerning a property-preserving transformation are discussed below:

1. *Forward preservation and backward preservation*—A transformation may preserve a property in two directions. Forward (resp., backward) preservation guarantees that a property of the original (resp., transformed) system is satisfied by the transformed (resp., original) system while being unable to guard against the creation of new and probably

undesired properties in it. Backward preservation is particularly useful if a transformation serves purely verification purposes.

Although highly desirable, it is uncommon that a transformation can preserve a property in both directions. In fact, even for one-way preservation, additional conditions often have to be imposed. In this paper, for each transformation, conditions are presented for two-way preservation.

2. *Preservation of multiple properties*—Very often, a system has several desirable properties. Then, for both system generation and verification, it is a challenge to discover a *single* transformation that can preserve all of them. Recent research aims at exploring for transformations which can preserve as many properties as possible [6,19,26].

Brief review on property-preserving transformations (see [26] for more detailed reviews):

Transformations on Petri nets may be roughly classified into three groups, namely reduction, refinement and composition. Most of the early works belong to the first two groups. Research in reduction methods began with simple pattern modifications on Petri nets [10,12,23–25,30]. Desel [10] showed that a live and safe FC net without frozen tokens can be reduced either to a live and safe marked graph or to a live and safe state machine. Esparza [12] provided reduction rules that reduce a live and bounded FC net to a circuit containing only one place and one transition. A well-known recent result is the preservation of well-formedness and Commoner's property under the merge of places within a free-choice net [11,13,14] or an asymmetric-choice net [21]. As for refinement methods, [30] introduced a refinement method for expanding a Petri net to the desired level of detail. Variations on refinement were studied in [5,30,31]. Brauer et al. [5] provided a survey on behavior and equivalence-preserving refinement methods. Recently, Huang et al. [7] showed the preservation of 19 properties under the refinement of a single transition or a single place. As for the third group (i.e., composition), [2] considered the 1-way merge of a set of non-neighboring places. P-invariants are shown to be preserved under such merge operations. Narahari et al. [28] investigated the following properties of the merged system: absence of deadlocks, conservativeness and boundedness. Soussi et al. [29] proposed the constraints for the preservation of liveness. Cheung [6] considered the problem of merging the places of two marked graphs. He has proposed a condition called cycle-inclusion property for checking the liveness, boundedness and reversibility of the integrated net. This condition was proved to be equivalent to the ST-property. However, his method has not been extended to augmented marked graphs and hence cannot be applied iteratively. Recently, Huang et al. [19] extended this approach to augmented marked graphs and provided a different method for checking the preservation of liveness, boundedness and reversibility. Mak [26] and Best et al. [4] showed that many properties are preserved under several kinds of composition that are induced by various operators, such as parallelism, choice, disable, etc.

Some papers studied a mixture of transformations. For example, Zeng and Cheung [8] proposed the conditions for preserving place invariants under five classes of transformations. [1] presented seven property-preserving transformation rules. Both papers include reduction, refinement and composition. Mak's work [26], while mainly for compositions, also included a path reduction problem which is a special case of the problem studied in Sections 3 and 4 of this paper.

Summary and organization of this paper

This paper first formulates the resource-sharing problem and system abstraction problem as a subnet-reducing transformation in Petri nets. Then, according to the structure of the shared resources or the abstracted parts, the following three transformations are presented for investigation in more detail. All transformations can be applied to both problems in principle.

Reducing a transition-bordered path to a single transition and reducing a place-bordered path to a single place (Sections 3 and 4): These two transformations have major applications in abstracting programs with a single entry and a single exit into single functions. They have been studied in the literature [25,12,26] but under much more restrictive conditions on the start and end transitions or places. Conditions are proposed in this paper for preserving liveness, boundedness and reversibility. Preservations of another seventeen properties such as siphon, trap, P-invariant, T-invariant and so on are not presented in this paper. They can be found in [18].

Reducing a place-bordered subnet to a single place (Section 5): This transformation is an extension of the above two path reductions wherein the place-bordered path or transition-bordered path is changed to a place-bordered subnet N_S. This transformation has major application in abstracting subprograms with multiple entries and/or multiple exits into single functions. Conditions are proposed for preserving conservativeness, structural boundedness, consistency, repetitiveness, boundedness, liveness and reversibility.

2. Fundamentals of petri nets

This section presents the preliminaries needed for the rest of the paper.

Definition 1. A *net* is a 4-tuple $N = (P, T, F, W)$, where P is a finite set of *places*, T is a finite set of *transitions* such that $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ is the *flow relation* and W is a *weight function* such that $W(x, y) \in \mathcal{N}^+$ (positive integers) if $(x, y) \in F$ and $W(x, y) = 0$ if $(x, y) \notin F$. A net is said to be *ordinary* if $W = 0$ or 1 for all arcs. In this case, W will be omitted.

For any $x \in P \cup T$, the *pre-set* of x is defined as ${}^\bullet x = \{y \in P \cup T \mid (y, x) \in F\}$ and the *post-set* of x is defined as $x^\bullet = \{y \in P \cup T \mid (x, y) \in F\}$. Similarly, for any subset $Y \subseteq P \cup T$, ${}^\bullet Y$ (resp., $Y^\bullet$) denotes the union set of ${}^\bullet y$ (resp., $y^\bullet$) for all $y \in Y$. A net is said to be *pure* or *self-loop-free* if ${}^\bullet x \cap x^\bullet = \emptyset$, $\forall x \in P \cup T$. The *incidence matrix* V of a net N is a $|P| \times |T|$ matrix whose element v_{ij} at row p_i and column t_j is calculated by $v_{ij} = W(t_j, p_i) - W(p_i, t_j)$. If it is clear from the context, symbols between column vectors and row vectors are not distinguished.

A *marking* of a net $N = (P, T, F, W)$ is a mapping $M : P \to \{0, 1, 2, \ldots\}$. A *Petri net* is a couple (N, M_0) where N is a net and M_0 is a marking of N called the initial marking. A place p is *marked* by M if $M(p) > 0$. Suppose $P' \subseteq P$, then P' is marked by M if there exists $p \in P'$ such that $M(p) > 0$.

A transition t of a net $N = (P, T, F, W)$ is *enabled* or *firable* at a marking M if $M(p) \geqslant W(p, t)\ \forall p \in {}^\bullet t$. A transition t may be *fired* if it is enabled. Firing transition t

results in changing marking M to a new marking M', where $M'(p) = M(p) - W(p,t) + W(t,p)\ \forall p \in P$. The process is denoted by $M[N,t\rangle M'$. For a sequence $\sigma = t_1 \dots t_k \in T^*$, $M[N,\sigma\rangle$ means that there exist markings M_i, $i = 1,\dots,k$ such that $M_0 = M$ and $M_{i-1}[N,t_i\rangle M_i$ and $M_{k-1}[N,t_k\rangle$. $L(N,M_0)$ denotes the language of (N,M_0), i.e., $L(N,M_0) = \{\sigma \mid M_0[N,\sigma\rangle\}$. $M[N,\sigma\rangle M'$ means that M' is *reachable* from M by firing sequence σ. If σ is not explicitly specified, the notation $M[N,{}^*\rangle M'$ is used. $R(N,M_0)$ denotes the set of all markings reachable from an initial marking M_0.

A *place invariant*, i.e., *P-invariant* (resp., *transition invariant, i.e., T-invariant*) of a net $N = (P,T,F,W)$ is a non-negative integer $|P|$-vector α (resp., $|T|$-vector β) satisfying the equation $\alpha V = 0$ (resp., $V\beta = 0$), where V is the incidence matrix of N.

Definition 2 (*Liveness*). A transition t is said to be *live* in (N,M_0) iff, for any $M \in R(N,M_0)$, there exists an $M' \in R(N,M)$ such that t can be fired at M'. (N,M_0) is said to be live iff all transitions are live in (N,M_0).

Definition 3 (*Reversibility*). A net (N,M_0) is said to be reversible iff $M_0 \in R(N,M)$ for any $M \in R(N,M_0)$.

Definition 4 (*Boundedness*). A place p is said to be *bounded* (resp. *safe*) in (N,M_0) iff, for any $M \in R(N,M_0)$, there exists a positive integer k such that $M(p) \leqslant k$ (resp., $M(p) \leqslant 1$). (N,M_0) is said to be bounded (resp., safe) iff all places of N are bounded (resp., safe). N is said to be *structurally bounded* iff (N,M_0) is bounded for any marking M_0.

The following characterization holds for structural boundedness [27]: N is structurally bounded iff there exists a $|P|$-vector $\alpha > 0$ such that $\alpha V \leqslant 0$.

Definition 5 (*Conservativeness, consistency and repetitiveness*). A net N is said to be *conservative* (resp., *consistent*, *repetitive*) iff there exists a $|P|$-vector $\alpha > 0$ such that $\alpha V = 0$ (resp., $|T|$-vector $\beta > 0$ such that $V\beta = 0$, $V\beta \geqslant 0$), where V is the incidence matrix of N.

Definition 6 (*State machine (SM) and marked graph (MG)*). A net $N = (P,T,F)$ is said to be a *state machine* iff $\forall t \in T : |t^\bullet| = |{}^\bullet t| = 1$. N is said to be a *marked graph* iff $\forall p \in P : |p^\bullet| = |{}^\bullet p| = 1$.

Definition 7 (*Subnet, connectedness and strongly connectedness*). A net $N_1 = (P_1, T_1, F_1)$ is said to be a subnet of another net N (in notation $N_1 \subseteq N$) iff $P_1 \subseteq P$, $T_1 \subseteq T$ and $F_1 = F \cap ((P_1 \times T_1) \cup (T_1 \times P_1))$. A subnet N_1 of N is said to be *induced* (or *generated*) by P_1 (resp., T_1) iff $T_1 = {}^\bullet P_1 \cup P_1^\bullet$ (resp., $P_1 = {}^\bullet T_1 \cup T_1^\bullet$). N is *connected* [11] iff it is not composed of two disjoint and non-empty subnets. N is *strongly connected* iff, for every pair of nodes (x,y), there exists a directed path from x to y.

Definition 8 (*Siphon and trap*). Let $N = (P,T,F)$ be a net and D be a subset of P. D is called a *siphon* (resp., trap) iff ${}^\bullet D \subseteq D^\bullet$ (resp., $D^\bullet \subseteq {}^\bullet D$). A siphon is said to be *minimal* if it does not contain any other siphons.

It is easy to show that (1) the union of siphons (resp., traps) is still a siphon (resp., trap), (2) a siphon remains token-free once it becomes free of tokens, and (3) a trap remains marked once it becomes marked.

3. Reducing a transition-bordered path to a transition

This section studies a transformation that reduces an elementary path to a single transition. The path both starts and ends at a transition. This transformation is formally stated below, where the place p may represent an elementary directed path starting and ending at a place. The entire path may also represent a subsystem that has a single entry and a single exit.

Reduce-T-Path (*reducing a transition-bordered path to a single transition*) (Fig. 1): Let (N, M_0), where $N = (P, T, F)$, is an ordinary Petri net. Suppose there exist $\varepsilon_i, \varepsilon_o \in T$ and $p \in P$ such that $\varepsilon_i \neq \varepsilon_o$, ${}^\bullet p = \{\varepsilon_i\}$, $p^\bullet = \{\varepsilon_o\}$ and ${}^\bullet\varepsilon_i \cap {}^\bullet\varepsilon_o = \varepsilon_i^\bullet \cap \varepsilon_o^\bullet = \phi$. Reduce-T-Path transforms (N, M_0) to (N', M_0') as follows:

$$P' = P - \{p\},$$

$$T' = (T - \{\varepsilon_i, \varepsilon_o\}) \cup \{\varepsilon'\},$$

$$\begin{aligned} F' = F &- (\{(x, \varepsilon_i) \mid x \in {}^\bullet\varepsilon_i\} \cup \{(\varepsilon_i, x) \mid x \in \varepsilon_i^\bullet\} \cup \{(x, \varepsilon_o) \mid x \in {}^\bullet\varepsilon_o\} \cup \{(\varepsilon_o, x) \mid x \in \varepsilon_o^\bullet\}) \\ &\cup (\{(x, \varepsilon') \mid x \in {}^\bullet\varepsilon_i \cup {}^\bullet\varepsilon_o - \{p\}\} \cup \{(\varepsilon', x) \mid x \in \varepsilon_o^\bullet \cup \varepsilon_i^\bullet - \{p\}\}) \end{aligned}$$

and

$$M_0'(p) = M_0(p) \quad \forall p \in P'.$$

The reduction rules studied in [25,3,26] are special cases of Reduce-T-Path in the sense that they satisfy an additional set of conditions: (a) $\varepsilon_o^\bullet \neq \emptyset$, ${}^\bullet\varepsilon_o = \{p\}$ and $M_0(p) = 0$ or (b) ${}^\bullet\varepsilon_i \neq \emptyset$, $\varepsilon_i^\bullet = \{p\}$ and $M_0(p) = 0$.

In the following Theorem 1, conclusions for preservation of liveness, boundedness and reversibility are presented. Some results about preserving siphon, trap, P-invariant, T-invariant and so on refer to [18].

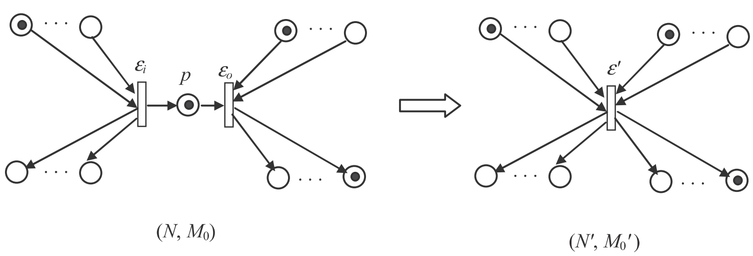


Fig. 1. Petri nets before and after applying Reduce-T-Path.

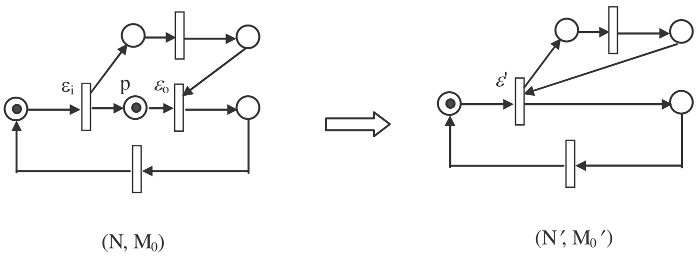


Fig. 2. An example showing that the properties liveness and reversibility are not preserved under Reduce-T-Path.

Theorem 1 (*Property preservation under Reduce-T-Path*). *Let* (N, M_0) *and* (N', M_0') *be the two Petri nets defined in Reduce-T-Path. Then,* (1) *If* (N, M_0) *is bounded, so is* (N', M_0'); (2) *If* (N, M_0) *is live* (*resp., reversible*), *in general,* (N', M_0') *is not live* (*resp., reversible*). (N', M_0') *is live* (*resp., reversible*) *if any of the following two conditions holds*: (a) $\varepsilon_o^\bullet \neq \emptyset$, ${}^\bullet\varepsilon_o = \{p\}$ *and* $M_0(p) = 0$ *or* (b) ${}^\bullet\varepsilon_i \neq \emptyset$, $\varepsilon_i^\bullet = \{p\}$ *and* $M_0(p) = 0$.

Proof. (1) Suppose (N', M_0') is unbounded. Then, there exists an infinite firing sequence $\sigma' = \sigma_1\varepsilon'\sigma_2\varepsilon' \ldots$ (or $\sigma' = \sigma_1\sigma_2 \ldots$) (where every σ_i does not contain ε') such that $M_0'[\sigma'\rangle M'$ and M' become unbounded. Let $\sigma = \sigma_1\varepsilon_i\varepsilon_o\sigma_2\varepsilon_i\varepsilon_o \ldots$ be obtained from σ' by replacing each ε' with $\varepsilon_i\varepsilon_o$ (or let $\sigma = \sigma_1\sigma_2 \ldots$). Obviously, σ is firable in N and $M_0[\sigma\rangle M$, where $M(q) = M'(q)$ for every $q \in P - \{p\}$ and $M(p) = M_0(p)$. Hence, (N, M_0) is unbounded—a contradiction.

(2) In general, liveness and reversibility are not preserved. For example, in Fig. 2, (N, M_0) is live and reversible but (N', M_0') is not. Refer to [26] for the proof of liveness under the particular cases. In the following, preservation of reversibility is proved under the particular cases. $\forall M' = M_0'[\sigma'\rangle$, suppose $\sigma' = \sigma_1\varepsilon'\sigma_2\varepsilon' \ldots$, where every σ_i does not contain ε'. $\exists M = M_0[\sigma\rangle$, where $\sigma = \sigma_1\varepsilon_i\varepsilon_o\sigma_2\varepsilon_i\varepsilon_o \ldots$ such that $M(s) = M'(s)$ for every $s \in P - \{p\}$ and $M(p) = 0$. Since N is reversible, $\exists\sigma_r = \sigma_1'\varepsilon_i\sigma_2'\varepsilon_o\sigma_3'\varepsilon_i\sigma_4'\varepsilon_o \ldots$ such that $M[\sigma_r\rangle M_0$. For Condition 1, firability of ε_i in N implies firability of ε' in N' and firing ε' in N' has the same effect as firing both ε_i and ε_o. Hence, in σ_r, ε_i is replaced by ε', ε_o is ignored and the resulting $\sigma_r' = \sigma_1'\varepsilon'\sigma_2'\sigma_3'\varepsilon'\sigma_4' \ldots$ is firable in N'. Also, $M'[\sigma_r'\rangle M_0'$. Similarly, under Condition 2, by letting $\sigma_r' = \sigma_1'\sigma_2'\varepsilon'\sigma_3'\sigma_4'\varepsilon' \ldots$, $M'[\sigma_r'\rangle M_0'$ follows.

The following Propositions 1 and 2 in the next section are obtained from Suzuki and Murata [30] and Cheung et al. [7] without proof.

Proposition 1. *Suppose ordinary Petri net* (N, M_0) *is obtained from* (N', M_0') *by splitting a transition* $t' \in T'$ *into a path* $\varepsilon_i p \varepsilon_o$ *such that* $({}^\bullet t'$ *in* $N') = ({}^\bullet\varepsilon_i$ *in* $N)$, $(t'^\bullet$ *in* $N') = (\varepsilon_o^\bullet$ *in* $N)$, $M_0(p) = 0$ *and the other parts remain unchanged. If* (N', M_0') *is live* (*resp., bounded, reversible*), *then* (N, M_0) *is live* (*resp., bounded, reversible*).

Corollary 1. *Let* (N, M_0) *be an ordinary Petri net and* (N', M_0') *be obtained from* (N, M_0) *by applying Reduce-T-Path. Suppose* $|\varepsilon_i^\bullet| = |{}^\bullet\varepsilon_o| = 1$. *Then* (N, M_0) *is live* (*resp., bounded, reversible*) *iff* (N', M_0') *is live* (*resp., bounded, reversible*).

4. Reducing a place-bordered path to a place

This section studies a transformation that reduces an elementary path to a single place. The path both starts and ends at a place. This transformation is formally stated below, where the transition ε may represent an elementary directed path starting and ending at a transition. The entire path may also represent a subsystem that has a single entry and a single exit.

Conclusions for preservation of liveness, boundedness and reversibility are presented. Some results about preserving siphon, trap, P-invariant, T-invariant and so on are refer to [18].

Reduce-P-Path (reducing a place-bordered path to a single place) (Fig. 3): Let (N, M_0), where $N = (P, T, F)$, be an ordinary Petri net. Suppose there exist $p_i, p_o \in P$ and $\varepsilon \in T$ such that $p_i \neq p_o$, ${}^\bullet\varepsilon = \{p_i\}$, $\varepsilon^\bullet = \{p_o\}$ and ${}^\bullet p_i \cap {}^\bullet p_o = p_i^\bullet \cap p_o^\bullet = \phi$. Reduce-P-Path transforms (N, M_0) to (N', M_0') as follows:

$$P' = (P - \{p_i, p_o\}) \cup \{p'\},$$

$$T' = T - \{\varepsilon\},$$

$$\begin{aligned} F' = F &- (\{(x, p_i) \mid x \in {}^\bullet p_i\} \cup \{(p_i, x) \mid x \in p_i^\bullet\} \cup \{(x, p_o) \mid x \in {}^\bullet p_o\} \\ &\cup \{(p_o, x) \mid x \in p_o^\bullet\}) \cup (\{(x, p') \mid x \in {}^\bullet p_i \cup {}^\bullet p_o - \{\varepsilon\}\} \\ &\cup \{(p', x) \mid x \in p_o^\bullet \cup p_i^\bullet - \{\varepsilon\}\}), \end{aligned}$$

$$M_0'(p) = M_0(p) \quad \text{if } p \neq p' \quad \text{and} \quad M_0'(p') = M_0(p_i) + M_0(p_o).$$

The reduction rules studied in [25,26] are special cases of Reduce-P-Path in the sense that they satisfy an additional set of conditions: (a) $p_o^\bullet \neq \emptyset$, ${}^\bullet p_o = \{\varepsilon\}$ and $M_0(p_o) = 0$ or (b) ${}^\bullet p_i \neq \emptyset$, $p_i^\bullet = \{\varepsilon\}$ and $M_0(p_o) = 0$.

Lemma 1. *Let (N, M_0) and (N', M_0') be the two Petri nets defined in Reduce-P-Path. Then, the following propositions hold*:

(1) *For every $M \in R(N, M_0)$, there exists $M' \in R(N', M_0')$ such that $M'(p') = M(p_i) + M(p_o)$ and $M'(p) = M(p)$ for $p \in P' - \{p'\}$.*

(2) *If, for each $t_i \in p_i^\bullet - \{\varepsilon\}$, there exists $t_o \in p_o^\bullet$ such that $t_i^\bullet = t_o^\bullet$ and ${}^\bullet t_i - \{p_i\} = {}^\bullet t_o - \{p_o\}$, then, for every $M' \in R(N', M_0')$, there exists $M \in R(N, M_0)$ such that $M(p_i) + M(p_o) = M'(p')$ and $M(p) = M'(p)$ for $p \in P - \{p_i, p_o\}$.*

Proof. (1) Since $M \in R(N, M_0)$, $\exists \sigma \in L(N, M_0)$ such that $M_0[N, \sigma\rangle M$. $\forall \varepsilon \in \sigma$, let M_1 and M_2 be the two markings just before and just after firing ε in N. Then, according to the way ε is eliminated in Reduce-P-Path, $M_i'(p') = M_i(p_i) + M_i(p_o)$ and $M_i'(p) = M_i(p)$ for $p \in P' - \{p'\}$, $i = 1, 2$ in N'. Suppose that σ' is the transition sequence obtained from σ by deleting all such ε, the above argument shows that $\sigma' \in L(N', M_0')$ and $M_0'[N', \sigma'\rangle M'$.

(2) Since $M' \in R(N', M_0')$, $\exists \sigma' \in L(N', M_0')$ such that $M_0'[N', \sigma'\rangle M'$. Let us try to fire σ' in N. Consider any $t \in \sigma'$. If $t \notin p_i^\bullet - \{\varepsilon\}$, then t is always firable in N. If $t = t_i \in p_i^\bullet - \{\varepsilon\}$, then t_i may or may not be firable in N. Each t_i that is firable in N is kept in σ', possibly

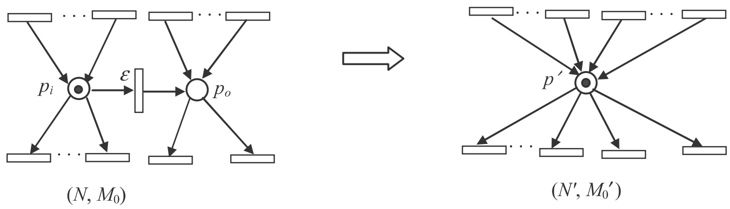


Fig. 3. Petri nets before and after applying Reduce-P-Path.

having to insert a ε into σ' if necessary. For each t_i that is not firable in N, since there exists $t_o \in p_o^\bullet$ such that $t_i^\bullet = t_o^\bullet$ and ${}^\bullet t_i - \{p_i\} = {}^\bullet t_o - \{p_o\}$, t_i is replaced with this t_o. Since t_i is firable in N', t_o is also firable in N', resulting in the same marking as firing t_i. This replacement results in a transition sequence σ such that $M_0[N, \sigma\rangle M$.

Theorem 2 (*Preservation of liveness, boundedness, and reversibility under Reduce-P -Path*). *Let* (N, M_0) *and* (N', M_0') *be the two Petri nets defined in Reduce-P-Path. Suppose at least one of the following conditions is valid*: (a) $p_o^\bullet \neq \emptyset$, ${}^\bullet p_o = \{\varepsilon\}$ *and* $M_0(p_o) = 0$. (b) ${}^\bullet p_i \neq \emptyset$, $p_i^\bullet = \{\varepsilon\}$ *and* $M_0(p_o) = 0$. (c) *For each* $t_i \in p_i^\bullet - \{\varepsilon\}$, *there exists* $t_o \in p_o^\bullet$ *such that* $t_i^\bullet = t_o^\bullet$ *and* ${}^\bullet t_i - \{p_i\} = {}^\bullet t_o - \{p_o\}$. *Then, if* (N, M_0) *is live* (*resp., bounded, reversible*), (N', M_0') *is live* (*resp., bounded, reversible*).

Proof. For Conditions (a) and (b), proof can be found in [25] and [26], respectively. For Condition (c), the proof proceeds as follows: For any reachable marking M' of (N', M_0') and any transition t in N', under the assumption of Condition (c), Lemma 1 implies that $\exists M \in R(N, M_0)$ such that $M(p_i) + M(p_o) = M'(p')$ and $M(p) = M'(p)$ for $p \in P - \{p_i, p_o\}$. Since (N, M_0) is live, $\exists M_1 \in R(N, M)$, such that t is firable at M_1. By Lemma 1, $\exists M_1' \in R(N', M')$ such that $M_1'(p') = M_1(p_i) + M_1(p_o)$ and $M_1'(p) = M_1(p)$ for $p \in P' - \{p'\}$. This implies that $\forall p \in {}^\bullet t$, $M_1'(p) \geqslant M_1(p)$. The fact that t is firable at M_1 implies that t is firable at M_1'. Hence, (N', M_0') is live. The proofs for boundedness and reversibility are similar to the above proof for liveness. They are omitted here.

Example 1. In all cases considered below, the path $p_i \varepsilon p_o$ in N is reduced to p' in N'. In Fig. 4, (N, M_0) is live, bounded and reversible. Since for $t_i \in p_i^\bullet - \{\varepsilon\}$, there exists $t_o \in p_o^\bullet$ such that $t_i^\bullet = t_o^\bullet = \{p\}$ and ${}^\bullet t_i - \{p_i\} = {}^\bullet t_o - \{p_o\} = \emptyset$, it follows from Theorem 2(c) that (N', M_0') is live, bounded and reversible.

Note that, in Theorem 2, each of the three Conditions (a), (b) or (c) is sufficient but not necessary. Figs. 5–7 show that different results may occur if none of these conditions holds. In N of all these figures, since ${}^\bullet p_o \neq \{\varepsilon\}$ and $p_i^\bullet \neq \{\varepsilon\}$, neither Condition (a) nor Condition (b) is satisfied; and, since $t_i^\bullet \neq t_o^\bullet$, Condition (c) is not satisfied either. Hence, (N', M_0') cannot be concluded to preserve all these three properties of (N, M_0). In fact, in Fig. 5, (N, M_0) is bounded and reversible but (N', M_0') is unbounded and not reversible. In Fig. 6, (N, M_0) is live but (N', M_0') is not. After firing $t_1 t_2 t_3 t_i t_i$ in (N', M_0'), transition t is dead.

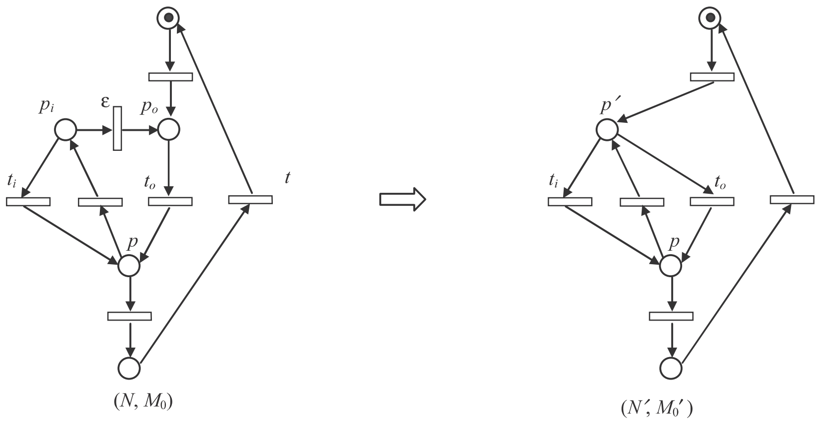


Fig. 4. Both (N, M_0) and (N', M_0') are live, bounded and reversible.

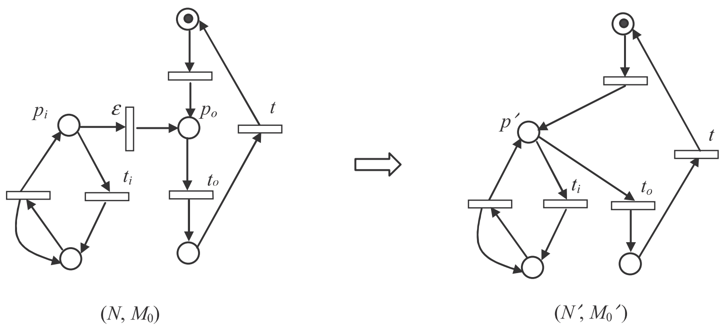


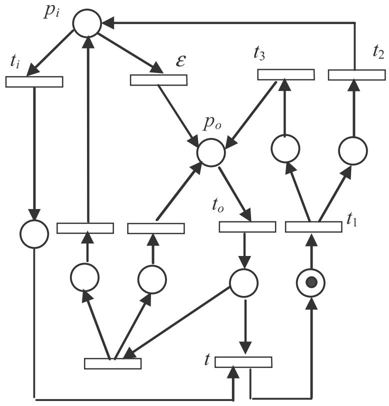
Fig. 5. (N, M_0) is bounded and reversible but (N', M_0') is unbounded and not reversible.

In Fig. 7, both (N, M_0) and (N', M_0') are live. Note that (N', M_0') is not shown in both Figs. 6 and 7.

Proposition 2 (*Suzuki and Murata [30] and Cheung et al. [7]*). *Suppose (N, M_0) is obtained from (N', M_0') by splitting a place $p' \in P'$ into a path $p_i t p_o$ such that $({}^\bullet p_i$ in $N) = ({}^\bullet p'$ in $N')$, $(p_o^\bullet$ in $N) = (p'^\bullet$ in $N')$, $M_0(p_i) + M_0(p_o) = M_0'(p')$ and the other parts remain unchanged. Then,*

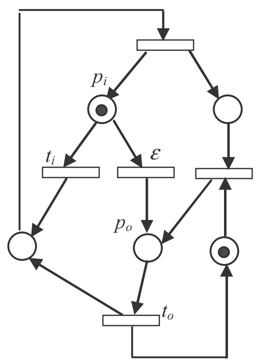
(1) *If (N', M_0') is live and ${}^\bullet p' \neq \phi$, then (N, M_0) is live.*
(2) *If (N', M_0') is bounded (resp., reversible), then (N, M_0) is bounded (resp., reversible).*

Corollary 2. *Let (N, M_0) be an ordinary Petri net and (N', M_0') be obtained from (N, M_0) by applying Reduce-P-Path. Suppose $|p_i^\bullet| = |{}^\bullet p_o| = 1$. Then (N, M_0) is live iff (N', M_0') is live and ${}^\bullet p' \neq \phi$. (N, M_0) is bounded (resp., reversible) iff (N', M_0') is bounded (resp., reversible).*



(N, M_0)

Fig. 6. A live (N, M_0) that become nonlive after applying Reduce-P-Path.



(N, M_0)

Fig. 7. A live (N, M_0) that is still live after applying Reduce-P-Path.

5. Reducing a place-bordered subnet to a place

This section studies a transformation that reduces a subnet N_S within an ordinary Petri net to a single place. This is an extension of the case studied in Sections 3 and 4. Conditions for the preservation of many properties will be derived.

Reduce-Subnet (reducing a place-bordered subnet to a single place) (Fig. 8): Let $N_S = (P_S, T_S, F_S)$ be a place-bordered (i.e., $({}^{\bullet}T_S \cup T_S^{\bullet}) \cap (P - P_S) = \emptyset$) subnet of an ordinary Petri net $N = (P, T, F)$. Suppose there exists a transition set $T_I \subseteq T - T_S$ such that the subnet generated by P_S and $T_S \cup T_I$ forms a strongly connected SM in N. Reduce-Subnet reduces N_S to a single place p_s by transforming (N, M_0) to (N', M_0'), where

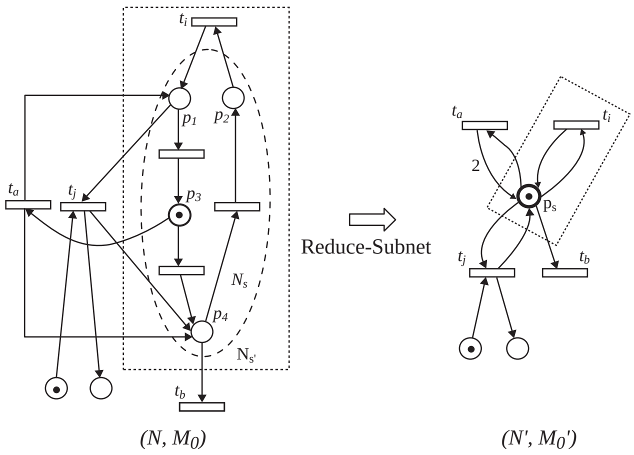


Fig. 8. The Petri nets before and after applying Reduce-Subnet.

$N' = (P', T', F', W')$, as follows:

$$P' = P - P_S \cup \{p_s\},$$

$$T' = T - T_S,$$

$$\begin{aligned} F' = F - F_S &- (\{(t, p), (p, t) \mid t \in T - T_S,\, p \in P_S\} \cap F) \\ &\cup \{(t, p_s) \mid t \in T - T_S,\, t^\bullet \cap P_S \neq \emptyset\} \cup \{(p_s, t) \mid t \in T - T_S,\, {}^\bullet t \cap P_S \neq \emptyset\}, \end{aligned}$$

$\forall t \in T_A = {}^\bullet P_S \cup P_S^\bullet - T_S - T_I$, $W'(p_s, t) = |{}^\bullet t \cap P_S|$ and $W'(t, p_s) = |t^\bullet \cap P_S|$. The weight of every other edge in F' remains 1; and

$$M_0'(p) = M_0(p) \quad \text{for } p \in P' - \{p_s\} \quad \text{and} \quad M_0'(p_s) = \sum_{p \in P_S} M_0(p).$$

Example 2 (*Fig. 8*). In N, the place-bordered subnet N_S lies within the ellipse, $T_I = \{t_i, t_j\}$ and $T_A = \{t_a, t_b\}$. N_S and T_I generate a strongly connected SM. In N', the transitions t_i and t_j form self-loops with p_s and the weight of the arc (t_a, p_s) is 2 because $|t_a^\bullet \cap P_S| = 2$ in N.

Some of the property-preservation results described later depend on the following condition.

Internal Path Condition (*IPC*) (Fig. 9): Consider the subnet $N_S = (P_S, T_S, F_S)$ of (N, M_0) in Reduce-Subnet. $\forall x \in ((T_I \cup T_A)^\bullet \cap P_S) \cup \{p \in P_S \mid M_0(p) > 0\}$, $\forall y \in {}^\bullet(T_I \cup T_A) \cap P_S$, there exists a path γ that starts at x and ends at y such that γ lies entirely within N_S.

Discussion on Reduce-Subnet and Internal Path Condition:

a. The subnet N_S can model subsystems with multiple entries and exits. For example, the subsystem in N_S of Fig. 8 has two entries $\{p_1, p_4\}$ and four exits $\{p_1, p_2, p_3, p_4\}$.

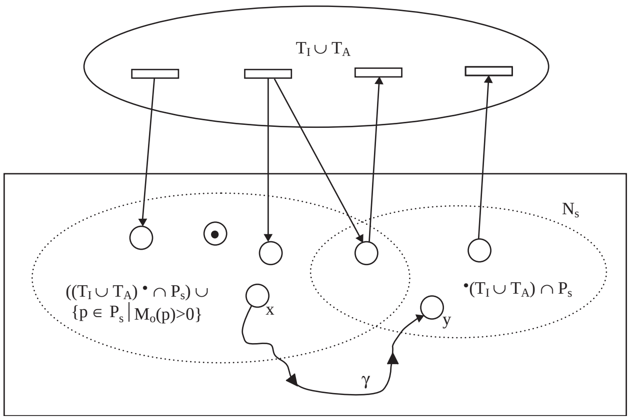


Fig. 9. Internal Path Condition (x is any place in $((T_I \cup T_A)^\bullet \cap P_S) \cup \{p \in P_S \mid M_0(p) > 0\}$, y is any place in $^\bullet(T_1 \cup T_A) \cap P_S$ and γ is a path from x to y within N_S).

b. Reduce-Subnet takes two practical requirements into consideration in its formulation. It is flexible enough so that it can have a large scope of application. First, the subnet N_S to be reduced is ‘almost’ a strongly connected SM, meaning that N_S itself is not required to be a strongly connected SM but must become so when combined with a set T_I of ‘included’ transitions. Second, the choice of T_I and T_A is not unique. For example, for the same N_S in Fig. 8, one can choose $T_I = \{t_i\}$ and $T_A = \{t_a, t_b, t_j\}$. In particular, if N_S is a strongly connected SM itself, T_I may even be empty. For example, for the subnet $N_{S'}$ within the dotted square in Fig. 8, one can let $T_I = \varnothing$ and $T_A = \{t_a, t_b, t_j\}$.
c. In the definition of Internal Path Condition, $(T_I \cup T_A)^\bullet \cap P_S$ denote the set of ‘entry’ places, $\{p \in P_S \mid M_0(p) > 0\}$ is the set of initially marked places and $^\bullet(T_I \cup T_A) \cap P_S$ denote the set of ‘exit’ places. IPC does not require N_S to be strongly connected. It only requires that, within N_S, there exists a directed path from every ‘entry’ place or initially marked place to every ‘exit’ place. Obviously, strongly connected N_S automatically satisfies IPC. A weaker condition IPC allows more flexibility in selecting a subnet for reduction during actual application.

Definition 9 (*Mappings arising from Reduce-Subnet*). Let (N', M_0') and (N, M_0) be the two Petri nets defined in Reduce-Subnet. For a firing sequence σ and a marking M of N where $M_0[N, \sigma\rangle M$, the mappings of σ and M from N onto N' are defined as follows:

$f : T^* \to T'^*$:

$$f(\lambda) = \lambda, \quad \text{where } \lambda \text{ is the null sequence},$$

$$f(\sigma t) = \begin{cases} f(\sigma) & \text{if } t \in T_S, \\ f(\sigma)t & \text{if } t \in T - T_S. \end{cases}$$

M' is the restriction of M from P to P':

$$M'(p) = M(p) \quad \text{if } p \in P' - \{p_s\}$$

$$M'(p_s) = \sum_{p \in P_S} M(p).$$

For the rest of this section, the notations N', M'_0, T_A, T_I, σ, $f(\sigma)$ and M' have the same meanings as defined in Reduce-Subnet or Definition 9. For simplification, the symbols σ and σ' are also used to denote the set of transitions in the sequences σ and σ', respectively.

Lemmas 2 and 3 below describe the relationships of σ and M with their mappings $f(\sigma)$ and M'.

Lemma 2. *Let (N', M'_0) and (N, M_0) be the two Petri nets defined in Reduce-Subnet. For any sequence σ and marking M of N where $M_0[N, \sigma\rangle M$, their mappings $f(\sigma)$ and M' satisfy $M'_0[N', f(\sigma)\rangle M'$.*

Proof (by induction on the length of σ). For $\sigma = \lambda$, obviously $M = M_0$. By Definition 9, $f(\sigma) = \lambda \in L(N', M'_0)$ and $M' = M'_0$. Hence, $M'_0[N', f(\sigma)\rangle M'$. Next, assume the proposition holds for every μ, where $|\mu| \leqslant n$. That is, for such μ and marking M_1, $M_0[N, \mu\rangle M_1$ implies that $M'_0[N', f(\mu)\rangle M'_1$. Let $\sigma = \mu t \in L(N, M_0)$ and marking M satisfy $M_0[N, \mu\rangle M_1[N, t\rangle M$.

To show that $f(\sigma)$ is firable, two cases should be considered:

a. If $t \in T_S$, then $f(\sigma) = f(\mu) \in L(N', M'_0)$ by Definition 9 and the above assumption.
b. If $t \in T - T_S$, then $f(\sigma) = f(\mu)t$. By the above assumption $M'_0[N', f(\mu)\rangle M'_1$, it is sufficient to show that t is firable at M'_1. By Definition 9, $M'_1(p) = M_1(p)$ for $p \in P - p_s$ and $M'_1(p_s) = \sum_{p \in P_S} M_1(p)$. Since t is firable at M_1 in N, $M_1(p) \geqslant W(p, t), \forall p \in {}^\bullet t$ in N. Hence, $M'_1(p) \geqslant W'(p, t), \forall p \in {}^\bullet t$ in N'. This implies that t is firable at M'_1 in N'. Hence, $f(\sigma)$ is firable in N'.

Next, consider two cases of t:

a. If $t \in T_S$, then $f(\sigma) = f(\mu)$. $M'(p_s) = M(P_S) = M_1(P_S) = M'_1(p_s)$ and $M'(p) = M(p) = M_1(p) = M'_1(p)$ for $p \in P - P_S$. Hence, $M' = M'_1$ and $M'_0[N', f(\sigma)\rangle M'$.
b. If $t \in T - T_S$, then $f(\sigma) = f(\mu)t$. $M'(p) = M(p) = M_1(p) \pm 1 = M'_1(p) \pm 1$ for $p \in {}^\bullet t \cup t^\bullet - p_s$, $M'(p_s) = \sum_{p \in P_s} M(p) = \sum_{p \in (P_s - {}^\bullet t \cup t^\bullet)} M_1(p) + \sum_{p \in (P_S \cap {}^\bullet t \cup t^\bullet)} (M_1(p) \pm 1) = M'_1(p_s) + W(t, p_s) - W(p_s, t)$, and $M'(p) = M(p) = M_1(p) = M'_1(p)$ for $p \in P' - ({}^\bullet t \cup t^\bullet)$. Hence, $M'_1[N', t\rangle M'$ and $M'_0[N', f(\sigma)\rangle M'$.

Lemma 3. *Suppose N satisfies the Internal Path Condition in Reduce-Subnet. Then, for any sequence σ' and marking M' of N', where $M'_0[N', \sigma'\rangle M'$, there exist sequence σ and marking M of N such that $\sigma' = f(\sigma)$ and $M_0[N, \sigma\rangle M$, where M' is the mapping of M.*

Proof. For any sequence σ' and marking M' of N', where $M'_0[N', \sigma'\rangle M'$, suppose $\sigma' = \sigma'_1 t_1 \sigma'_2 t_2, \ldots, l_i \sigma'_i t_i, \ldots, l_j \ldots t_k \sigma'_d$, where every $\sigma'_i \cap (p_s^\bullet \cup {}^\bullet p_s) = \emptyset$, every $t_i \in {}^\bullet p_s$ and every $l_i \in p_s^\bullet$. Then $t_i, l_i \in T_I \cup T_A$ in N and the Internal Path Condition implies that $\forall x \in (P_S \cap t_i^\bullet) \cup \{p \in P_S \mid M_0(p) > 0\}$ and $\forall y \in {}^\bullet l_i \cap P_S,\ i = 1, 2 \ldots$, there exists a path γ_i from x to y such that γ_i lies entirely within N_S in N. Since these paths lie within a connected SM, they are all firable sequences at M_1 if $M_1(p_r) > 0$, where $p_r \in \gamma_i$ and $M_1 \in R(N, M_0)$, and every firing will preserve the number of tokens within P_S. In particular, some of them are fired so that every place $y \in {}^\bullet l_i,\ i = 1, 2, \ldots$ gets a token eventually in N. Let σ_i be such a firing sequence if a sequence in γ_i is fired and a null

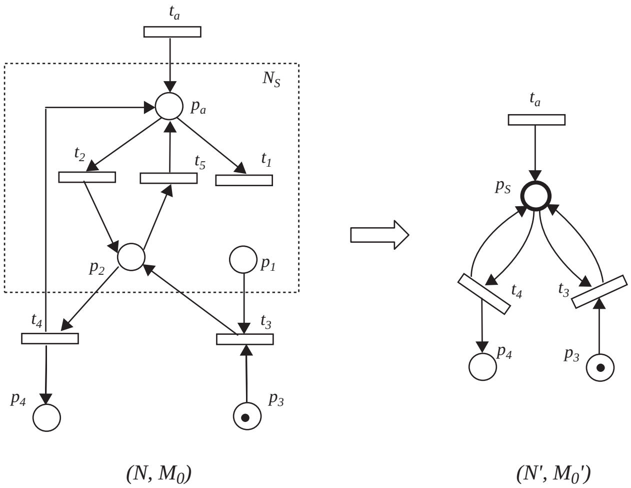


Fig. 10. An example for explaining Lemma 3.

sequence otherwise. Hence, the sequence $\sigma = \sigma'_1 t_1 \sigma'_2 t_2, \ldots, \sigma_i l_i \sigma'_i t_i, \ldots, \sigma_j l_j \ldots t_k \sigma'_d$ is firable and $f(\sigma) = \sigma'$. Suppose $M_0[N, \sigma\rangle M_2$. Since firing σ_i preserves the number of tokens within P_S, $M_2(P_S) = M'(p_s)$ and $M_2(p) = M'(p)$ for $p \in P - P_S$. Hence $M_2 = M$.

Example 3 (*Fig. 10*). For the place-bordered subnet N_S within the square, $P_S = \{p_a, p_1, p_2\}$, $T_S = \{t_1, t_2, t_5\}$, $T_I = \{t_3, t_4\}$, $T_A = \{t_a\}$, $(T_I \cup T_A)^{\bullet} \cap P_S = \{p_a, p_2\}$, ${}^{\bullet}(T_I \cup T_A) \cap P_S = \{p_1, p_2\}$. For N_S, $\{p_a, p_2\}$ is the set of 'entry' places and $\{p_1, p_2\}$ the set of 'exit' places. Since the paths $p_a t_1 p_1$, $p_a t_2 p_2$, $p_2 t_5 p_a t_1 p_1$ and p_2 all lie within N_S, IPC is satisfied. By Lemma 3, for the firing sequences $\sigma'_1 = t_a t_3$, $\sigma'_2 = t_a t_4$, $\sigma'_3 = t_a t_3 t_4$ and $\sigma'_4 = t_a t_4 t_3$ of (N', M'_0), the firing sequences $\sigma_1 = t_a t_1 t_3$, $\sigma_2 = t_a t_2 t_4$, $\sigma_3 = t_a t_1 t_3 t_4$ and $\sigma_4 = t_a t_2 t_4 t_1 t_3$ of (N, M_0) satisfy $\sigma'_i = f(\sigma_i)$, $i = 1, 2, 3, 4$. The other sequences are just subsequences of these ones.

Without IPC, Lemma 3 may be invalid. For example, in Fig. 10, suppose p_1 of Fig. 10 is initially marked. Since there is no path from p_1 to p_2 within N_S, IPC is not satisfied. For the firing sequence $\sigma' = t_4$ in N', there is no firing sequence σ in N, such that $\sigma' = f(\sigma)$. Similarly, for the subnet N_S within the ellipse in Fig. 11, $P_S = \{p_1, p_2, p_3\}$, $T_S = \{t_1, t_2, t_3\}$, $T_I = \{t_4\}$, $T_A = \{t_a, t_b, t_c\}$, $(T_I \cup T_A)^{\bullet} \cap P_S = \{p_1, p_2, p_3\}$, ${}^{\bullet}(T_I \cup T_A) \cap P_S = \{p_1, p_2, p_3\}$. Since there is no directed path from p_1 to p_3 within N_S, IPC is not satisfied. For the firing sequence $\sigma' = t_a t_a t_c$ in N', there is no firing sequence σ in N, such that $\sigma' = f(\sigma)$.

Theorem 3 (*Preservation of boundedness, liveness and reversibility under Reduce-Subnet*). *Let (N, M_0) and (N', M'_0) be the two Petri nets defined in Reduce-Subnet. Then, the following propositions hold*:

(1) *If (N', M'_0) is bounded, then (N, M_0) is bounded.*

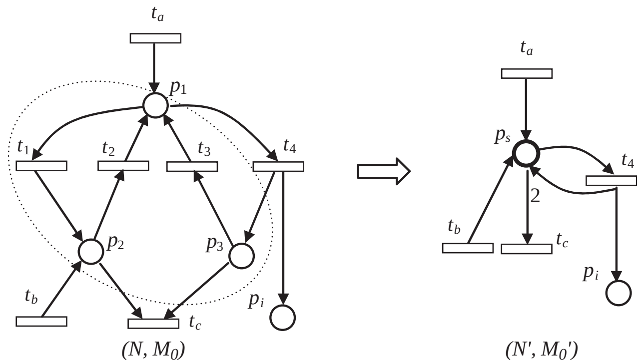


Fig. 11. An example for explaining the Internal Path Condition of Lemma 3.

(2) *If* (N, M_0) *is bounded and either the Internal Path Condition is satisfied or* ${}^\bullet T_I \cap (P - P_S) = \emptyset$ *(i.e.,* T_I *has no input places outside the subnet* N_S*), then* (N', M_0') *is bounded.*

(3) *Suppose the Internal Path Condition is satisfied. Then,* (N, M_0) *is live* (*resp., reversible*) *iff* (N', M_0') *is live* (*resp., reversible*).

(4) *If* (N', M_0') *is live* (*resp., reversible*) *and* ${}^\bullet T_I \cap (P - P_S) = \emptyset$, *then* (N, M_0) *is live* (*resp., reversible*).

Proof. (1) Suppose (N', M_0') is bounded. For every reachable marking M of (N, M_0), by Lemma 2, its mapping M' is a reachable marking of (N', M_0'). By Definition 9, for every place p in N, $M(p)$ is bounded by $M'(p)$ or $M'(p_s)$. Hence,(N, M_0) is bounded.

(2) Suppose (N, M_0) is bounded. For every $M' \in R(N', M_0')$ obtained by firing $\sigma' = \sigma_1' t_1 \sigma_2' t_2, \ldots, l_i \sigma_i' t_i, \ldots, l_j \ldots t_k \sigma_d'$ in N', where every $\sigma_i' \cap (p_s^\bullet \cup {}^\bullet p_s) = \emptyset$, every $t_i \in {}^\bullet p_s$ and every $l_i \in p_s^\bullet$, if the Internal Path Condition is satisfied, by Lemma 3, $M'(p)$ is obviously bounded by $M(p)$ or by $M(P_S)$, where M and M' satisfy the mapping relation in Definition 9. If the Internal Path Condition is not satisfied, $\forall x \in (P_S \cap t_i^\bullet) \cup \{p \in P_S \mid M_0(p) > 0\}$ and $\forall y \in {}^\bullet l_i \cap P_S, i = 1, 2 \ldots$, there exists a path γ_i from x to y such that γ_i lies entirely within the strongly connected SM generated by N_S and T_I in N. By the assumption ${}^\bullet T_I \cap (P - P_S) = \emptyset$, the paths are all firable sequences at M_1 if $M_1(p_r) > 0$, where $p_r \in \gamma_i$ and $M_1 \in R(N, M_0)$. Some of them are fired so that every place $y \in {}^\bullet l_i, i = 1, 2, \ldots$ gets a token eventually in N. Let σ_i be such a firing sequence if a sequence in γ_i is fired and a null sequence otherwise. Hence, the sequence $\sigma = \sigma_1' t_1 \sigma_2' t_2, \ldots \sigma_i l_i \sigma_i' t_i, \ldots \sigma_j l_j \ldots t_k \sigma_d'$ is firable and $f(\sigma) = \sigma_1' t_1 \sigma_2' t_2, \ldots s_i l_i \sigma_i' t_i, \ldots s_j l_j \ldots t_k \sigma_d'$, where every $s_i \in T_I$. Suppose $M_0[N, \sigma\rangle M_2$. Then, $M_2(P_S) = M'(p_s) M_2(p) = M'(p)$ for $p \notin s_i^\bullet$ and $M_2(p) \geqslant M'(p)$ for $p \in s_i^\bullet$. Hence, (N', M_0') is bounded.

(3) ($\Rightarrow$) Suppose (N, M_0) is live. For every $\sigma' \in L(N', M_0')$ and every $t \in T'$, since the Internal Path Condition is satisfied, by Lemma 3, there exists $\sigma \in L(N, M_0)$ such that $\sigma' = f(\sigma)$. Since (N, M_0) is live, there exists $\sigma_1 \in T^*$ such that $\sigma\sigma_1 t \in L(N, M_0)$. By Definition 9 and Lemma 2, $f(\sigma\sigma_1 t) = \sigma'\sigma_1' t \in L(N', M_0')$. Hence, (N', M_0') is live. ($\Leftarrow$) Suppose (N', M_0') is live. For every $\sigma \in L(N, M_0)$ and every $t \in T$, by Lemma 2, there

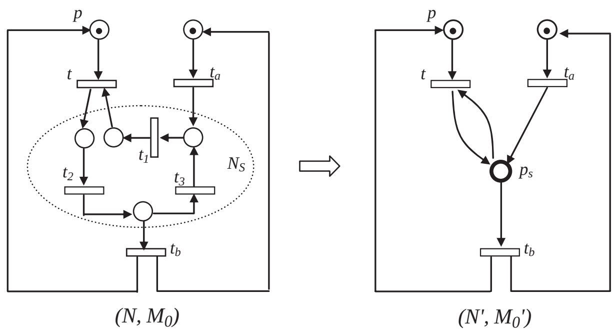


Fig. 12. An example for illustrating the role of ${}^\bullet T_I \cap (P - P_S) = \emptyset$ in Theorem 3.

exists $\sigma' \in L(N', M_0')$ such that $\sigma' = f(\sigma)$. Since (N', M_0') is live, there exists $\sigma_1' \in T'^*$ such that $\sigma'\sigma_1't \in L(N', M_0')$. Since the Internal Path Condition is satisfied, by Lemma 3, there exists $\sigma_1 \in T^*$ such that $\sigma\sigma_1 t \in L(N, M_0)$ and $f(\sigma\sigma_1 t) = \sigma'\sigma_1't$. Hence, (N, M_0) is live.

(4) Suppose (N', M_0') is live. For every $\sigma \in L(N, M_0)$ and every $t \in T$, by Lemma 2, there exists $\sigma' \in L(N', M_0')$ such that $f(\sigma) = \sigma'$.

Case 1: If $t \in T - T_S$ in N, then $t \in T'$ in N'. Since (N', M_0') is live, there exists $\sigma_1' = \mu_1 a_1 a_2 \ldots \mu_2\, b_1 b_2 \ldots \mu_3 a_3 a_4 \ldots \mu_4 b_3 b_4 \ldots \mu_d t \in T'^*$ such that $\sigma'\sigma_1' \in L(N', M_0')$, where every $\mu_i \cap ({}^\bullet p_s \cup p_s^\bullet) = \emptyset$, $a_i \in {}^\bullet p_s = {}^\bullet P_S - T_S$ and $b_j \in p_s^\bullet = P_S^\bullet - T_S$. By the proof of Proposition (2), for such a firable sequence in N', there exists a firable sequence $\sigma_1 = \mu_1 a_1 a_2 \ldots \mu_2 \gamma_1 \gamma_2 \ldots b_1 b_2 \ldots \mu_3 a_3 a_4 \ldots \mu_4 \gamma_3 \gamma_4 \ldots b_3 b_4 \ldots \mu_d t$ in N, where each γ_i is the firable sequence such that firing $\gamma_1\gamma_2\gamma_3\gamma_4 \ldots$. can guarantee that b_j, $j = 1, 2, 3, \ldots$, is still firable in N. Hence, $\sigma\sigma_1 \in L(N, M_0)$.

Case 2: If $t \in T_S$, then $t \notin T'$ in N'. By Case 1, every $t_j \in T - T_S$ is live. Hence, there exists $\sigma_1 \in T^*$ such that $\sigma\sigma_1 t_j \in L(N, M_0)$, where $t_j \in {}^\bullet P_S - T_S$. Let $M_0[N, \sigma\sigma_1 t_j\rangle M_j$. Then, $M_j(P_S) \geqslant 1$. Since P_S and $T_S \cup T_I$ generate a strongly connected SM and T_I has no input places in $P - P_S$, every $t \in T_S$ is obviously a potentially firable transition in (N, M_j). Hence, (N, M_0) is live. The proof for reversibility is similar.

Example 4 (*Fig. 12*). In Theorem 3, without the condition ${}^\bullet T_I \cap (P - P_S) = \emptyset$, (N, M_0) may be nonlive although (N', M_0') is live and (N', M_0') may be unbounded although (N, M_0) is bounded. For example, for the subnet N_S (within the ellipse) of N and $T_I = \{t\}$, ${}^\bullet T_I \cap (P - P_S) = \{p\} \neq \emptyset$, (N', M_0') is live but (N, M_0) is not because N is dead after firing $t_a t_1 t t_2 t_3 t_1$. (N, M_0) is bounded but (N', M_0') is not because p in N' becomes unbounded if the sequence $t_a t_b$ is fired repeatedly.

Corollary 3. *Let (N, M_0) and (N', M_0') be the two Petri nets defined in Reduce-Subnet. Suppose the place-bordered subnet N_S is itself a strongly connected state machine. Then, (N, M_0) is live (resp., bounded, reversible) iff (N', M_0') is live (resp., bounded, reversible).*

Proof. In this case, let $T_I = \emptyset$. It is then always true that ${}^{\bullet}T_I \cap (P - P_S) = \emptyset$ and the Internal Path Condition is satisfied.

The following theorem is about preservation of siphon, trap, conservativeness, structural boundedness, consistence and repetitiveness under Reduce-Subnet. Because the proof for this theorem can refer to [19] and [22], we omitted it here.

Theorem 4 (*Preservation of siphons, traps, conservativeness, structural boundedness, consistence, repetitiveness under Reduce-Subnet*). *Let (N, M_0) and (N', M_0') be the two Petri nets defined in Reduce-Subnet. Then, the following propositions hold*:

(1) *For a set of places $D \subseteq P$ of N, suppose either $D \cap P_S = \emptyset$ or $P_S \subseteq D$. Then, D is a siphon* (*resp., trap*) *of N iff D or $D - P_S \cup \{p_s\}$ is a siphon* (*resp., trap*) *of N'.*

(2) *If there exists a vector $\alpha = (\alpha_1, I_a) > 0$, where $I_a = (a, a, \ldots, a)$ is a $|P_S|$-vector, such that $\alpha V = 0$* (*resp.*, $\alpha V \leqslant 0$), *where V is the incidence matrix of N, then N' is conservative* (*resp., structurally bounded*).

(3) *If N' is conservative* (*resp., structurally bounded*), *then N is conservative* (*resp., structurally bounded*).

(4) *If N is consistent* (*resp., repetitive*), *then N' is consistent* (*resp., repetitive*).

6. Applications to the verification for manufacturing systems

This section illustrates the application of the three transformations to verifying a manufacturing system. Although not shown in this paper, the three transformations can also be applied to system specification since they are two-way preserving transformations. [16] and [17] illustrate the applications to the specification and verification in multi-agent systems and manufacturing systems, respectively.

Example 5 (*Fig. 13 and Table 1*). This manufacturing system consists of three processes: two workstations WS_1 and WS_2 (on the left of Fig. 13) for assembly work and one machining center (on the right of Fig. 13) for machining. WS_1 and WS_2 share robot R_2 between themselves and share Robot R_1 with the machining center. (Note: The left and right components of Fig. 13 are extracted from [34]. Zhou et al. used them just for explaining the concepts of mutual exclusions in resource sharing. They are combined here with some modifications to create an example for illustrating our results.) The system runs as follows:

A. In the machining center, parts are machined first by machine M_1 and then by machine M_2. Each part is automatically fixtured to a pallet and loaded into the machine. After processing, robot R_1 unloads the intermediate part from M_1 into buffer B. At machining station M_2, intermediate parts are automatically loaded into M_2 and processed. When M_2 finishes processing a part, the robot R_1 unloads the final product, defixtures it and returns the fixture to M_1.

B. When either WS_1 or WS_2 is ready to execute the assembly task, it requests both robots R_1 and R_2 and acquires them if they are available. When a workstation starts an assembly task, it cannot be interrupted until it is completed. When WS_1 (WS_2) completes, it releases both robots.

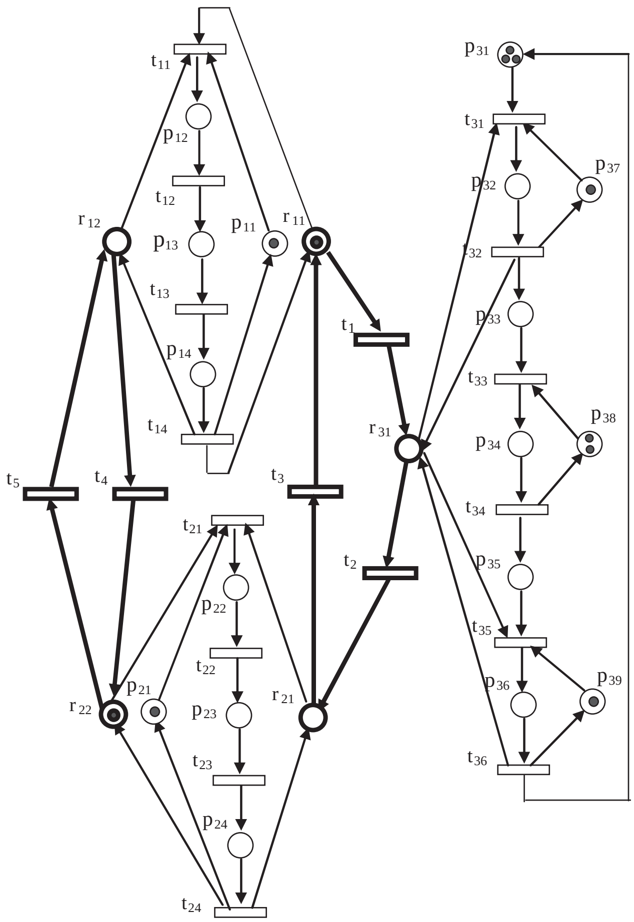


Fig. 13. The original system (N, M_0) with two resource subnets (boldfaced).

C. It is assumed that input parts are always available to be fixtured and that the finished products are removed.

For the specification of the manufacturing system with Petri nets, each operation process is abstracted to a single place and each transition represents the start or/and completion of a process. This is similar to the literature [32,34]. For handling resources sharing problems, this paper has some differences with the literature.

Unlike other systems where the robots are shared among the processes without any modifications, this example considers a more general situation where a robot has to go through some intermediate treatments (e.g., cleaning the oil left from the previous process, adding some parts needed by the next calling process, etc.) when being passed from one process to another. Hence, for the Petri net specification of the system (Fig. 13), each resource is originally represented by a set of places (called *resource-places* hereafter), one in each of the parts it is involved in. The resource-places may form a connected subnet whose

Table 1
The legend for Fig. 13

Places	Transitions
r_{i1} ($i = 1, 2, 3$): Robot R_1 is available	t_{11}: starts acquiring R_1 and R_2
r_{i2} ($i = 1, 2$): Robot R_2 is available	t_{12}: starts first step of assembling at WS_1
p_{11}: WS_1 requests R_1 and R_2	t_{13}: starts final step of assembling at WS_1
p_{12}: WS_1 acquires R_1 and R_2	t_{14}: completes assembling at WS_1
p_{13}: first step of assembling at WS_1	t_{21}: starts acquiring R_1 and R_2
p_{14}: final step of assembling at WS_1	t_{22}: starts first step of assembling at WS_2
p_{21}: WS_2 requests R_1 and R_2	t_{23}: starts final step of assembling at WS_2
p_{22}: WS_2 acquires R_1 and R_2	t_{24}: completes assembling at WS_2
p_{23}: first step of assembling at WS_2	t_{31}: starts activity p_{32}
p_{24}: final step of assembling at WS_2	t_{32}: completes activity p_{32} and start activity p_{33}
p_{31}: pallets are available	t_{33}: completes p_{33} and start the storage activity p_{34}
p_{32}: machine M_1 loads, fixtures and processes a palleted raw part	t_{34}: completes p_{34} and start activity p_{35}
p_{33}: R_1 unloads an intermediate part to the buffer	t_{35}: completes p_{35} and start p_{36}
p_{34}: buffer B stores an intermediate part	t_{36}: completes p_{36}
p_{35}: machine M_2 loads and processes an intermediate part	t_i ($i = 1, 2, 3, 4, 5$): intermediate processing on a robot before passing it from one process to another.
p_{36}: R_1 unloads a final product from M_2, defixtures and returns the pallet	
p_{37}: M_1 is available	
p_{38}: B is available	
p_{39}: M_2 is available	

transitions represent the intermediate processes. For example, Robot R_1 is shared by the three parts (WS_1, WS_2 and the machining center) and need some intermediate treatments when being passed from one part to another. In Fig. 13, places r_{11}, r_{21} and r_{31} are resource-places representing robot R_1. Transitions t_1, t_2 and t_3 represent the intermediate processes. The resource-places and these transitions generate a connected subnet (one of the bold-faced subnets in Fig. 13).

Verification on the final system (Fig. 13) proceeds in three steps:

Step 1: (N, M_0) *(Fig. 13) is transformed to* (N_1, M_1) *(Fig. 14) by using Reduce-Subnet.*

In Fig. 13, the two bold-faced subnets N_{S1} (generated by $\{r_{11}, r_{21}, r_{31}, t_1, t_2, t_3\}$) and N_{S2} (generated by $\{r_{12}, r_{22}, t_4, t_5\}$) are strongly connected SMs. By setting $T_{I1} = T_{I2} = \emptyset$, $T_{A1} = \{t_{11}, t_{14}, t_{21}, t_{24}, t_{32}, t_{33}, t_{35}, t_{36}\}$ and $T_{A2} = \{t_{11}, t_{14}, t_{21}, t_{24}\}$, Reduce-Subnet reduces N_{S1} and N_{S2} to places r_1 and r_2, respectively, resulting in (N_1, M_1) (Fig. 14). By Corollary 3, (N, M_0) is live, bounded and reversible iff (N_1, M_1) is.

Step 2: (N_1, M_1) *(Fig. 14) is transformed to* (N_2, M_2) *(Fig. 15) by using Reduce-T-Path.*

In (N_1, M_1), the transition-bordered paths $s_1 = t_{11}p_{12}t_{12}p_{13}t_{13}p_{14}t_{14}$, $s_2 = t_{21}p_{22}t_{22}p_{23}t_{23}p_{24}t_{24}$, $s_3 = t_{31}p_{32}t_{32}$, $s_4 = t_{33}p_{34}t_{34}$ and $s_5 = t_{35}p_{36}t_{36}$ satisfy the conditions in Corollary 1. e.g., $|t_{11}^{\bullet}| = |^{\bullet}t_{14}| = 1$ for path s_1 and $|t_{33}^{\bullet}| = |^{\bullet}t_{34}| = 1$ for path s_4. Reduce-T-Path reduces the five paths to transitions s_1, s_2, s_3, s_4, and s_5, respectively, resulting in (N_2, M_2) (Fig. 15). By Corollary 1, (N_1, M_1) is live, bounded and reversible iff (N_2, M_2) is live, bounded and reversible.

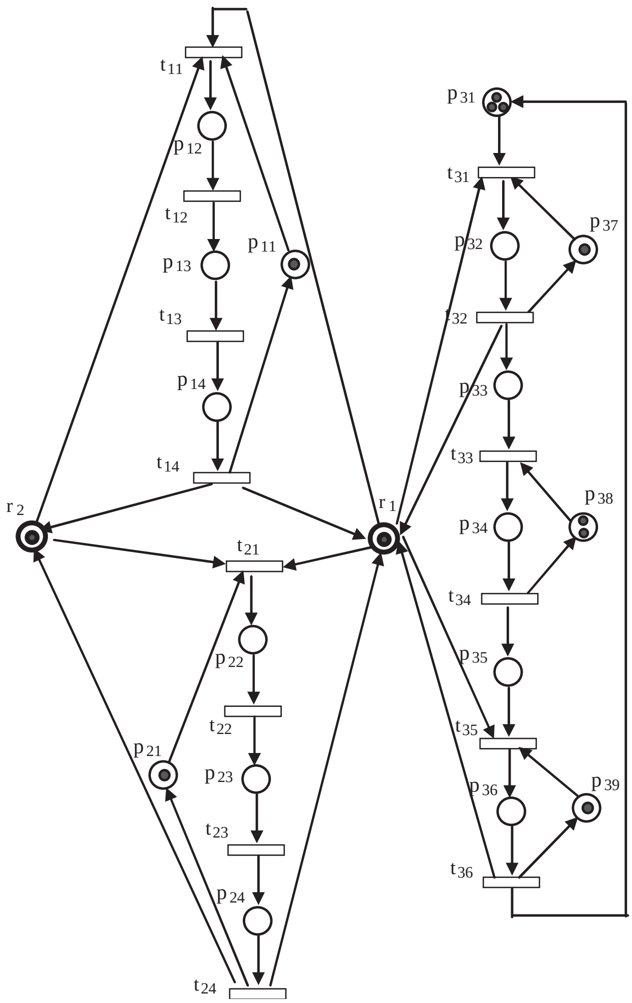


Fig. 14. The system (N_1, M_1), resulting from (N, M_0) by using Reduce-Subnet.

Step 3: *Deleting all the places p in* (N_2, M_2) (*Fig.* 15) *satisfying* $^\bullet p = p^\bullet$ *and* $M_2(p) > 0$ *results in Petri net* (N_3, M_3) (*Fig.* 16).

Since those marked places $p_{11}, p_{21}, p_{37}, p_{38}, p_{39}, r_1, r_2$ consist of self-loops with their associated transitons in (N_2, M_2), deleting them and their associated arcs will not affect the firing sequences and token distribution. Hence, (N_2, M_2) is live, bounded and reversible iff (N_3, M_3) is live, bounded and reversible.

Hence, the complex manufacturing model (N, M_0) (Fig. 13) is live, bounded and reversible if and only if so is (N_3, M_3) (Fig. 16). Since (N_3, M_3) is an initially marked cycle together with two independently transitions, it is obviously live, bounded and reversible [27]. Hence, the manufacturing system (N, M_0) (Fig. 13) is live, bounded and reversible.

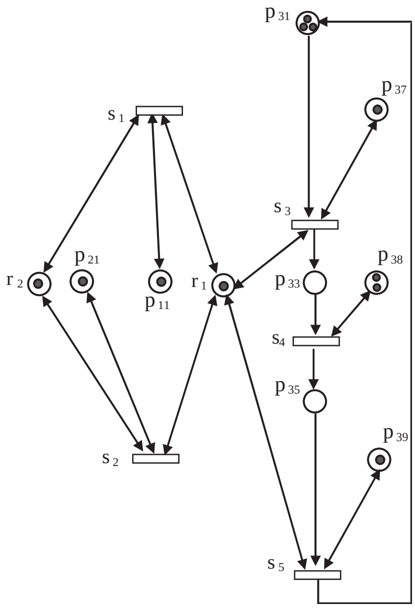


Fig. 15. The system (N_2, M_2), resulting from (N_1, M_1) by using Reduce-T-Path.

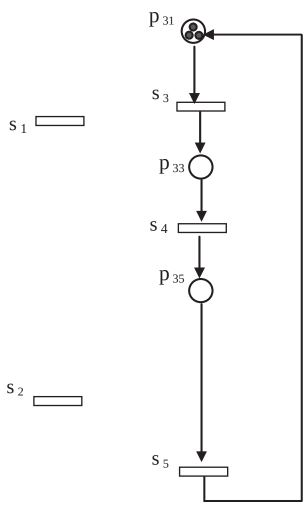


Fig. 16. Deleting self-loops in (N_2, M_2) results in (N_3, M_3).

7. Conclusion

Based on Petri nets, this paper has made the following contributions towards solving the resource-sharing and subsystem abstraction problems in system design:

A. *Enhancing the capability for modeling*—In the literature, these problems are described in quite a straightforward manner as exemplified by Chu's AMGs and Zhou's sequential and parallel mutual exclusions. Also, the systems involved are modeled mostly as an SM or MG. This paper formulates the problems as subnet-reducing transformations. Three transformations are proposed so that a designer has considerable flexibility in selecting an appropriate transformation for specifying the resources, the system, the subsystems and the problems under investigation. In particular, a resource is now allowed to receive intermediate processing when switching from one user to another.
B. *Formalizing the property-preserving approach for verification*—In the literature, very little has been devoted to the development of formal verification techniques specifically for the resource-sharing and system abstraction problems. Usually, just general techniques were suggested. Based on the subnet-reducing transformations, this paper proposes a property-preserving approach for verification. For each of the three transformations, conditions are imposed on the structure of the subnets to be reduced so that various properties of the net will be preserved.
C. Besides their applications to system design, the results obtained in this paper also enrich the theory for property-preserving transformations in Petri nets.

For the three transformations, they touch only the tip of a scarcely explored research area. This area obviously still has several open problems, including reducing transition-bordered subnets, more general subnets, etc. However, even for such simple path-reduction and subnet-reduction problems as considered in this paper, quite restrictive conditions are already imposed. Deeper insights are needed in order to investigate these open problems.

References

[1] Aalst, Wil van der, Verification of workflow nets, Lecture Notes in Comput. Sci. 1248 (1997) 407–426.
[2] T. Agerwala, Y. Choed-Amphai, A synthesis rule for concurrent systems, Proc. 15th Design Automation Conference, June 1978, pp. 305–311.
[3] G. Berthelot, Checking properties of nets using transformations, Lecture Notes in Comput. Sci. 222 (1986) 19–40.
[4] E. Best, R. Devillers, M. Koutny, Petri Net Algebra, Springer, Berlin, 2001.
[5] W. Brauer, R. Gold, W. Vogler, A survey of behavior and equivalence preserving refinement of petri nets, Lecture Notes in Comput. Sci. 483 (1990) 1–46.
[6] K.S. Cheung, Use-case-driven system design: a synthesis methodology based on labelled petri nets, Ph.D. Thesis, Dept. of Computer Science, City University of Hong Kong, 2002.
[7] H.J. Huang, T.Y. Cheung, W.M. Mak, Structure and behavior preservation by Petri-net-based refinements in system design, Theoret. Comput. Sci. 328 (2004) 245–269.
[8] T.Y. Cheung, W. Zeng, Invariant-preserving transformations for the verification of place/transition systems, IEEE Trans. Systems Man Cybernetics—Part A: Systems Humans 28 (1) (1998) 114–121.
[9] F. Chu, X. Xie, Deadlock analysis of petri nets using siphon and mathematical programming, IEEE Trans. Robotics Automation 13 (6) (1997) 793–804.
[10] J. Desel, Reduction and design of well-behaved concurrent systems, Lecture Notes in Comput. Sci. 458 (1990) 166–181.
[11] J. Desel, T. Esparza, Free Choice Petri Nets, Cambridge University Press, Cambridge, 1995.
[12] J. Esparza, Reduction and synthesis of live and bounded free choice petri nets, Inform. Comput. 114 (1) (1994) 50–87.
[13] J. Esparza, M. Silva, On the analysis and synthesis of free choice systems, Lecture Notes in Comput. Sci. 483 (1990) 186–243.

[14] J. Esparza, M. Silva, Compositional synthesis of live and bounded free choice petri nets, Lecture Notes in Comput. Sci. 527 (1991) 172–187.
[15] C. Girault, R. Valk, Petri Nets for System Engineering—A Guide to Modeling, Verification, and Applications, Springer, Berlin, 2003.
[16] H.J. Huang, T.Y. Cheung, A Property-Preserving Component-Based Methodology for Designing Multi-agent Systems, submitted.
[17] H.J. Huang, T.Y. Cheung, Applications of Property-Preserving Algebras to Component-Based Manufacturing System Designs, submitted.
[18] H.J. Huang, T.Y. Cheung, W.M. Mak, Preservation of Nineteen Properties under Place and Transition Reductions, Proc. 9th Bellman Continuum, Vol. 2, 2002, pp. 328–336.
[19] H.J. Huang, L. Jiao, T.Y. Cheung, Property-preserving composition of augmented marked graphs that share common resources, Proc. 2003 IEEE Internat. Conf. on Robotics and Automation, 2003, pp. 1446–1451.
[20] M.D. Jeng, F. DiCesare, Synthesis using resource control nets for modeling shared-resources systems, IEEE Trans. Robotics Automation 11 (3) (1995) 317–327.
[21] L. Jiao, T.Y. Cheung, W.M. Lu, On liveness and boundedness of asymmetric choice nets, Theoret. Comput. Sci. 311 (2004) 165–197.
[22] L. Jiao, H.J. Huang, T.Y. Cheung, Property-preserving composition by place merging, J. Circuits, Systems and Computers, to appear.
[23] I. Koh, F. DiCesare, Modular transformation methods for generalized petri nets and their applications in manufacturing automation, IEEE Trans. Systems Man Cybernetics 21 (1991) 963–973.
[24] B.H. Krogh, C.L. Beck, Synthesis of place/transition nets for simulation and control of manufacturing systems, Proc. IFIP Symp. on Large Scale Systems (1986) 661–666.
[25] H. Lee-Kwang, Generalized petri net reduction method, IEEE Trans. Systems Man Cybernetics 17 (2) (1987) 297–303.
[26] W.M. Mak, Verifying property preservation for component-based software systems (a petri-net based methodology), Ph.D. Thesis, Department of Computer Science, City University of Hong Kong, June 2001.
[27] T. Murata, Petri nets: properties, analysis, and applications, Proc. IEEE 77 (4) (1985) 541–580.
[28] Y. Narahari, N. Viswanadham, A petri net approach to the modeling and analysis of flexible manufacturing systems, Ann. Operat. Res. 3 (1995) 449–472.
[29] Y. Souissi, On liveness preservation by composition of nets via a set of places, Lecture Notes in Comput. Sci. 524 (1990) 277–295.
[30] I. Suzuki, T. Murata, A method for stepwise refinement and abstraction of petri nets, J. Comput. System Sci. 27 (1983) 51–76.
[31] R. Valette, Analysis of petri nets by stepwise refinements, J. Comput. System Sci. 18 (1979) 35–46.
[32] K.S. Valvanis, On the hierarchical analysis and simulation of flexible manufacturing systems with extended petri nets, IEEE Trans. System Man Cybernetics 20 (1) (1990) 94–100.
[33] M. Zhou, Generalizing parallel and sequential mutual exclusions for petri net synthesis of manufacturing systems, IEEE Int. Conf. on Emerging Technologies and Factory Automation, Vol. 1, Hawaii, November, 1996. pp. 49–55.
[34] M. Zhou, F. DiCesare, Petri Net Synthesis for Discrete Event Control of Manufacturing Systems, Kluwer Academic Publishers, Dordrecht, 1993.
[35] M. Zhou, K. Venkatesh, Modeling, Simulation, and Control of Flexible Manufacturing Systems: a Petri Net Approach, World Scientific Publishing Co. Pte. Ltd., 1999.