Extended Structure Morphisms inducing a Petri net semantics.

J. Delcour
J. Heirbaut
D. Janssens

Department of Computer Science
Free University of Brussels (V.U.B.)

November 16, 1993

Abstract

ESM systems have been developed by D. Janssens [Jan93] as a model of concurrent systems based on graph rewriting. It will be shown that ESM systems can be used to model Petri nets or, more precisely, that Petri nets can be seen as ESM systems without any edges between the places. This leads to a Petri net semantics based upon the external effect of computations (or processes), which is compositional with respect to the composition of Petri nets. This semantics describes how tokens have been rewritten by a computation rather than describing the firing of transitions.

Contents

1 Introduction

Extended Structure Morphisms [Jan93] are a generalization of Actor Grammars [Jan90, Jan91] which are graph grammar [EKL90, EL91, Löw91] based descriptions of Actors.

The semantics of ESM systems are based on computation structures that describe how its initial graph gets transformed by a rewriting process of the ESM system. Every step in this rewriting process in represented in the computation structure. The semantics of an ESM system P can then be described by the set of computation structures associated with P. A more abstract semantics can be obtained by considering the external effect of such computation structures, that is by describing how the initial graph is transformed to its resulting graph by the complete rewriting process without showing the single rewriting steps of this process. This is more abstract in the sense that two different computation structures can have the same effect if they transform the initial graph in the same way to its result graph using different rewritings.

The aim of this article is to show that ESM systems are capable of modelling Petri nets [Rei85] by translating P/T-nets in ESM systems that operate on discrete graphs. P/T-nets are then a special case of ESM systems, called Petri systems. It will be shown that for every P/T-net, an ESM system can be constructed such that there is both a structural and semantical equivalence between them. Computation structures of these Petri systems will then correspond to processes, and hence will be called process structures. We will then investigate the Petri net semantics resulting from the external effect of ESM systems, by considering the external effect of the constructed ESM systems. It should then be clear from this approach that the essential difference between P/T-nets and ESM systems is the absence of relations between the tokens in the former.

Section 3 defines the notion of similarity between P/T-nets and Petri systems, a special kind of ESM systems suited to model Petri nets, and between processes and process structures (computation structures of Petri systems). Furthermore, the relation between similarity and isomorphism is shown.

In section 4, it is first shown that process structures yield the same kind of semantics for Petri systems as processes do for Petri nets, i.e. different P/T-nets have the same sets of processes if and only if similar Petri systems have the same sets of process structures. Then the external effect semantics of ESM systems is applicated onto P/T-nets and the differences with traditional Petri net semantics are investigated.

2 Preliminaries

In this section we recall some basic terminology about graphs, relations and Petri nets [Rei85] to be used in the paper and we summarize the results of ESM systems as described in [Jan93].

2.1 Graphs and Relations

1. For a set A, Id_A denotes the identity relation on A. For a relation $R \subseteq A \times A$, R^* denotes the reflexive and transitive closure of R. We will often write R^*_A to stress that the closure is taken with respect to the set A. The relation R is *antisymmetric* if $(x, y) \in R$, $(y, x) \in R$ implies that $x = y$. For sets A and B, the difference of A and B is denoted by $A - B$. The union of disjoint sets A and B is often denoted by $A \oplus B$. For disjoint sets A, B and functions $f_1 : A \rightarrow C$ and $f_2 : B \rightarrow C$, the common extension of f_1 and f_2 to a function from $A \oplus B$ into C is denoted by $f_1 \oplus f_2$.

2. Let Σ and Δ be sets. A (Σ, Δ)*–graph* is a system $g = (V, E, \varphi)$ where V is a finite set (called the *set of nodes of* g), $E \subseteq V \times \Delta \times V$ (called the *set of edges* of g), and φ is a function from V into Σ (called the *node–labeling function of* g). For a (Σ, Δ)–graph g, its components are denoted by V_g, E_g and φ_g, respectively. A *discrete* Σ*–graph* is a (Σ, Δ)–graph g for which $E_g = \emptyset$.

3. Let g and h be (Σ, Δ)–graphs, let $f : V_g \rightarrow V_h$ be an injective function and let $R \subseteq V_g \times \Delta \times V_g$. Then the set $\{(f(x), \delta, f(y)) \mid (x, y) \in R\}$ is denoted by $f(R)$. We use a similar notation in the case where $R \subseteq V_g \times V_g$ or $R \subseteq (V_g \times \Delta) \times (V_g \times \Delta)$, and for the inverse relation f^{-1}. $f(g)$ denotes the graph $(f(V_g), f(E_g), \varphi_h \circ f)$.

4. A *graph morphism from g into h* is an injective function $f : V_g \to V_h$ such that $\varphi_g = \varphi_h \circ f$ and $f(E_g) \subseteq E_h$. f is a *graph isomorphism* if its inverse is a graph morphism from h into g. Id_g^{gr} denotes the identical graph morphism on g.

5. Let g and h be (Σ, Δ)–graphs. Then h is a *subgraph of g* if $V_h \subseteq V_g$, $E_h \subseteq E_g$ and φ_h is the restriction of φ_g to V_h. For a subset A of V_g, the *subgraph of g induced by A* is the graph $(A, E_g \cap (A \times \Delta \times A), \varphi')$, where φ' is the restriction of φ_g to A. The graphs g and h are *disjoint* if $V_g \cap V_h = \emptyset$. For disjoint graphs g and h, $g \oplus h$ denotes the graph $(V_g \oplus V_h, E_g \oplus E_h, \varphi_g \oplus \varphi_h)$.

Throughout this paper we assume that Σ and Δ denote arbitrary but fixed alphabets.

2.2 Petri nets

1. A *net* with arc weights is a triple $N = (S, T, W)$ where S and T are disjoint sets ($S \cap T = \emptyset$) and $W : ((S \times T) \cup (T \times S)) \to \mathbb{N}$. We sometimes denote the three components S,T and W of a net N by S_N,T_N and W_N, respectively.

   For $x \in S_N \cup T_N$, ${}^\bullet x = \{\, y \mid W_N(y, x) \geq 1 \,\}$ is called the *preset* of x, $x^\bullet = \{\, y \mid W_N(x, y) \geq 1 \,\}$ is called the *postset* of x. An element $x \in N$ is called *isolated* if ${}^\bullet x \cup x^\bullet = \emptyset$. N is called simple if distinct elements do not have the same pre- and postsets: $\forall x, y \in S_N \cup T_N : {}^\bullet x = {}^\bullet y) \wedge (x^\bullet = y^\bullet) \Rightarrow x = y$.

   Let N, N' be two nets and let $\beta : S_N \cup T_N \to S_{N'} \cup T_{N'}$ be a bijection. N and N' are called β–*isomorphic* if $s \in S_N \Leftrightarrow \beta(s) \in S_{N'}$, $t \in T_N \Leftrightarrow \beta(t) \in T_{N'}$ and $W_N(x, y) = W_{N'}(\beta(x), \beta(y))$.

2. Let N be a net. A mapping $M : S_N \to \mathbb{N}$ is called a *marking* of N.

3. A P/T-net (or Petri net) is a four-tuple $N = (S, T, W, M_0)$ such that (S, T, W) is a net with arc weights and $M_0 : S \to \mathbb{N}$ is an initial marking.

4. An occurence net $N = (B, E, F)$ is an acyclic ordinary net without branched places, i.e. $\forall x, y \in N : (x, y) \in F^* \wedge x \neq y \Rightarrow (y, x) \notin F^*$ and $\forall b \in B : |{}^\bullet b| \leq 1 \wedge |b^\bullet| \leq 1$. Elements of E are called events and elements of B are called conditions.

   A B*–cut* $c \subseteq B$ of an occurence net (B, E, F) is a maximal unordered set of B–elements with respect to F^*, that is, $\forall x, y \in c : (x \neq y) \Rightarrow (x, y), (y, x) \notin F^*$. $Min(N)$ is defined as $\{\, x \in N \mid {}^\bullet x = \emptyset \,\}$, and $Max(N)$ is defined as $\{\, x \in N \mid x^\bullet = \emptyset \,\}$.

5. A *process* $\pi = (B, E, F, \nu)$ of a Petri net $N = (S, T, W, M_0)$ is an occurence net (B, E, F) together with a labelling $\nu \subseteq (B \times S) \cup (E \times T)$ such that ν is a function and $Min(B, E, F)$ is a B-cut which corresponds to M_0, that is, $\forall s \in S : M_0(s) = |p^{-1}(s) \cap Min(N)|$. Let $\Pi(N)$ be the set of all processes of a Petri net N.

For technical simplicity, we will only consider nets without isolated elements.

2.3 ESM systems

1. Let g and h be (Σ, Δ)-graphs. An *ESM morphism* from g into h is a 3-tuple $R = (R^c, R^s, R^t)$ of relations such that

   (a) $R^c, R^t \subseteq V_g \times V_h$ and $R^s \subseteq (V_g \times \Delta) \times (V_h \times \Delta)$.

   (b) $R^t \subseteq R^c$ and, for each $\delta, \mu \in \Delta$, $((x, \delta), (y, \mu)) \in R^s$ implies $(x, y) \in R^c$.

   (c) $R(E_g) \subseteq E_h$.

   where $R(E) = \{(u, \mu, w) \in E_h \mid \exists (x, \delta, y) \in E : ((x, \delta), (u, \mu)) \in R^s \textit{ and } (y, w) \in R^t\}$ with $E \subseteq E_g$.

$Id_g^{ESM} = (Id_{V_g}, Id_{V_g \times \Delta}, Id_{V_g})$ is an ESM morphism from g into g, called the *identical ESM-morphism on g*. A *primitive ESM morphism* is an ESM morphism $R : g \rightarrow h$ such that g, h are nonempty and $R^c = V_g \times V_h$. For an ESM morphism $R : g \rightarrow h$, let $Min(R)$ denote V_g and $Max(R)$ denote V_h.

An ESM system P is a set of primitive ESM morphisms. The elements of an ESM system are often called *productions*. Two ESM systems P_1 and P_2 will be called *γ–isomorphic* if there exists a bijection $\gamma : P_1 \rightarrow P_2$ that maps a production of P_1 on an isomorphic production of P_2.

2. A computation structure is a 4-tuple (V, E, φ, R) such that (V, E, φ) is a (Σ, Δ)-graph, $R : (V, E, \varphi) \rightarrow (V, E, \varphi)$ is an ESM morphism, R^c, R^s and R^t are reflexive, transitive and antisymmetric, and for each $((x, \delta), (y, \mu)) \in R^s$, $x = y$ implies $\delta = \mu$.

   The 4-tuple $(V_g, E_g, \varphi_g, Id_g^{ESM})$ is a computation structure, called the *identical computation* structure on a (Σ, Δ)-graph g. For each primitive ESM morphism $R : g \rightarrow h$, R may be represented by the computation structure $C_R = (V_g \oplus V_h, E_h \cup R(E_g), \phi_g \oplus \phi_h, R^*)$. The 4-tuple $(\emptyset, \emptyset, \emptyset, R_\emptyset)$ is called the *trivial computation structure.*

   The set of minimal and maximal nodes of a computation structure C with respect to R_C^c are denoted by $Min(C)$ and $Max(C)$, respectively. Note that for a primitive ESM morphism $p : g \rightarrow h$, $Min(p) = Min(C_p)$ and $Max(p) = Max(C_p)$.

   For computation structures $C_1 = (V_1, E_1, \varphi_1, R_1)$, $C_2 = (V_2, E_2, \varphi_2, R_2)$, a *CS morphism* from C_1 into C_2 is an injective function $f : V_1 \rightarrow V_2$ such that $\varphi_2 \circ f = \varphi_1$, $f(E_1) \subseteq E_2$, $f(R_1^c) \subseteq R_2^c$, $f(R_1^s) \subseteq R_2^s$ and $f(R_1^t) \subseteq R_2^t$. The computation structures together with CS morphisms form a category CS. By consequence, f is a *CS isomorphism* if its inverse is a CS morphism

3. Let P be an ESM system and let C be a computation structure. A P-covering of C is a collection $(f_i, \pi_i)_{i \in I}$ of pairs, with $C_{\pi_i} = (V_i, E_i, R_i, \varphi_i)$, such that

   (a) For each $i \in I$, $\pi_i \in P$ and $f_i : C_{\pi_i} \rightarrow C$ is a CS morphism.

   (b) For each $x \in V_C$, there are at most two indices $i \in I$ such that $x \in f_i(V_i)$, and if $x \in f_i(V_i) \cap f_j(V_j)$ where $i \neq j$, then either $x \in f_i(Max(C_{\pi_i})) \cap f_j(Min(C_{\pi_j}))$ or $x \in f_i(Min(C_{\pi_i})) \cap f_j(Max(C_{\pi_j}))$.

   A P–covering of C is valid if $R_C = (\cup_{i \in I} f_i(R_i))^*_{V_C}$ and $E_C = R_C(E_{min} \cup \cup_{i \in I} f_i(E_{i,max}))$ where $E_{min} = E_C \cap (Min(C) \times \Delta \times Min(C))$ and $E_{i,max} = E_i \cap (Max(C_{\pi_i}) \times \Delta \times Max(C_{\pi_i}))$.

   C is P–valid if there exists a valid P–covering of C. The set of all P–valid computation structures of an ESM system P is called $Comp(P)$.

4. Let $C_1 = (V_1, E_1, \phi_1, R_1)$, $C_2 = (V_2, E_2, \phi_2, R_2)$ be computation structures. A (C_1, C_2)-interaction is a 3-tuple $int = (C_{int}, d_1, d_2)$ where $C_{int} = (V_{int}, E_{int}, \phi_{int}, R_{int})$ is a computation structure and $d_1 : C_{int} \rightarrow C_1$, $d_2 : C_{int} \rightarrow C_2$ are CS-morphisms such that

   (a) for each $x \in V_{int}$, either $d_1(x) \in Min(C_1)$ and $d_2(x) \in Max(C_2)$, or $d_1(x) \in Min(C_2)$ and $d_2(x) \in Max(C_1)$, and

   (b) The relation $(d_1^{-1}(R_1^c) \cup d_2^{-1}(R_2^c))^*_{V_{int}}$ is antisymmetric.

   The composition of C_1 and C_2 over int is the set

$$C_1 \Box_{int} C_2 = \{(C_{12}, c_1, c_2) \mid \text{the diagram } \begin{array}{ccc} C_{int} & \xrightarrow{d_2} & C_2 \\ {\scriptstyle d_1}\downarrow & & \downarrow{\scriptstyle c_2} \\ C_1 & \xrightarrow{c_1} & C \end{array} \text{ is a pushout in CS } \}$$

   We will often write $C_{12} \in C_1 \Box_{int} C_2$ instead of $(C_{12}, c_1, c_2) \in C_1 \Box_{int} C_2$.

   $Comp(P)$ an now be derived inductively as follows:

(a) The trivial computation structure is in $Comp(P)$.

(b) For every (Σ, Δ)-graph g, the identical computation structure on g is in $Comp(P)$.

(c) For every $p \in P$, every computation structure isomorphic with C_p is in $Comp(P)$.

(d) For every $C_1, C_2 \in Comp(P)$ and int a (C_1, C_2)–interface, $C_1 \Box_{int} C_2$ is a subset of $Comp(P)$.

Note that $Comp(P)$ is closed under isomorphism.

5. Let $C = (V, E, \varphi, R)$, then the *external effect* of C, denoted $\mathit{Eff}(C)$, is the ESM morphism $R_{eff} : g \rightarrow h$ where

$$V_g = Min(C), \quad E_g = E \cap V_g \times \Delta \times V_g, \quad \varphi_g = \varphi \cap V_g \times \Sigma,$$
$$V_h = Max(C), \quad E_h = E \cap V_h \times \Delta \times V_h, \quad \varphi_h = \varphi \cap V_h \times \Sigma, \text{ and}$$
$$R_eff = (R^c \cap V_g \times V_h, R^s \cap (V_g \times \Delta) \times (V_h \times \Delta), R^t \cap V_g \times V_h)$$

The external effect of an ESM system P is the set of all external effects of P–valid computation structures, i.e. $\mathit{Eff}(P) = \{\, \mathit{Eff}(C) \mid C \in Comp(P) \,\}$. The external effect of a system P can also be derived inductively since $\mathit{Eff}(C_1 \Box_{int} C_2) = \mathit{Eff}(C_1) \Box_{int} \mathit{Eff}(C_2)$.

3 Petri systems

Since P/T-nets and ESM systems are both transition based formalisms, it is a rather straightforward approach to map markings (state descriptions of P/T-nets) onto graphs (state descriptions of ESM systems) and transitions onto productions. As markings formally associate with each place a quantity, we can model them as discrete graphs in which every place appears as many times as the label of a node as given by the marking. Figure 1 shows a marking (left) with a corresponding graph (right). The names in this figure represent node identities in the marking, and represent node labels in the graph. The identities of the nodes are not shown in the graphical representation of a graph.

This implies that the productions of this specific ESM systems will only operate on discrete graphs. This, by consequence, eliminates the need for source and target relations in the ESM morphisms, since they describe how source and target parts of edges are transferred between graphs.

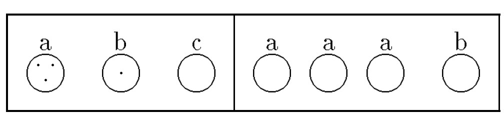


Figure 1: A marking (left) and its graph representation (right)

In the subsequent paragraphs, the restricted versions of ESM systems we will use (called Petri systems) are introduced. Thereafter, we will define the constructions between ESM systems and Petri nets. In the following section, we'll do the same between their computations and processes. At last, it is proved that computations play the same role for ESM systems as processes do for P/T-nets.

3.1 Basic definitions

We will define a discrete ESM morphism, a process structure and a Petri system as restrictions of an ESM morphism, a computation structure and an ESM system respectively, by omitting all references to edges as described above. Process structures and Petri systems will be the ESM equivalents of respectively processes and P/T-nets.

Definition 3.1 *Let g, h be discrete Σ–graphs, then $R : g \rightarrow h$ is a discrete ESM morphism if $(R, \emptyset, \emptyset)$ is an ESM morphism from g into h.*

Definition 3.2 *A Σ–process structure is a triple (V, R, φ) where $R \subseteq V \times V$ and $\varphi : V \rightarrow \Sigma$ such that $(V, \emptyset, \varphi, (R, \emptyset, \emptyset))$ is a (Σ, Δ)–computation structure.*

These definitions stress the fact that discrete ESM morphisms and (Σ, Δ)–process structures are restrictions of ESM morphisms and computation structures. We can also say that a discrete ESM morphism is a relation from V_g to V_h and that a (Σ, Δ)–process structure is a triple (V, R, φ) where $R \subseteq V \times V$ is a reflexive, transitive and acyclic relation on V.

Graphically, we will represent a discrete ESM morphism $R : g \rightarrow h$ by placing g above h and drawing the ESM morphism as vertical lines from g to h (see also figure 2). A process structure (V, R, φ) will be represented graphically by drawing R vertically as a partial order between the nodes, i.e. the reflexive and transitive relations are not shown (figure 4).

Since a Petri net contains the notion of an initial marking, we will have to add the corresponding notion of an initial graph to ESM systems. A Petri system then consists of productions, i.e. primitive discrete ESM morphisms, and an initial graph $Init$.

Definition 3.3 *A primitive discrete ESM morphism is a discrete ESM morphism $R : g \rightarrow h$ such that g, h are nonemepty discrete Σ–graphs and $R = V_g \times V_h$.*

Definition 3.4 *A Petri system is a tuple $ES = (Init, P)$ where*

1. *P is a set of primitive discrete ESM morphism.*
2. *$Init$ is a discrete Σ–graph.*

Definition 3.5 *Let $ES = (Init, P)$ be a Petri system. The set of computation structures of ES is defined by $IComp(ES) = \{ C \in Comp(P) \mid \mathrm{Min}(C)$ is isomorphic with $Init \}$.*

For ES a Petri system and C a process structure, if C is an element of $IComp(ES)$ then we say that C is a process structure of ES.

3.2 Nets and Petri systems

Having defined Petri systems, we can now describe which Petri systems correspond to a given Petri net and vice versa. This will be done by defining a construction and a similarity property. The constructions *Esm* and *Petri* construct one Petri system resp. net that fullfills the similarity property. We then show that every other Petri system (resp. net) meeting this similarity property, is isomorphic to the one constructed by *Esm* resp. *Petri*.

Nets and systems will be called "similar" if there exists a bijection between their transitions and productions that maps productions to similar transitions. As is shown in Figure 2, a production is similar to a transition if every place occurs as many times as the label of a node in the left respectively right side graphs, as described by the weights to respectively from the transition. We will therefore consider the alfabets Σ and S to be equal.

In the graphical representation of a net, circles respresent places (or occurences) and boxes represent transitions. The flow relation is drawn by directed edges, labelled with their weights (if different from one). The names represent node identities. The tokens are represented by dots. A production is represented graphically by drawing circles for the nodes labelled with the node labels. The ESM morphism $R : g \rightarrow h$ then is drawn by vertical lines with g above h.

Figure 3 shows a Petri net and a similar Petri system. As is shown in this figure, we will often give productions a name to reference them. This name is only for the ease of use and is not a part of the Petri system.

Definition 3.6 (similarity of nets and systems)
Let N be a Petri net and ES a Petri system where $N = (S, T, W, M_0)$ and $ES = (Init, P)$.

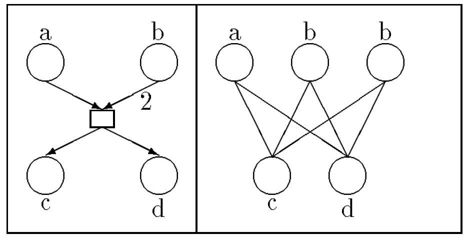


Figure 2: A transition (left) and a similar production (right).

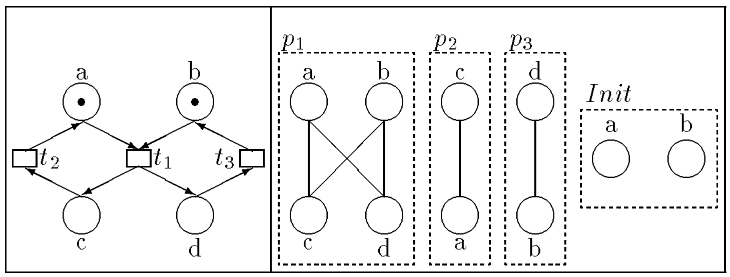


Figure 3: A Petri net and a similar Petri system

1. *Let p be a primitive discrete ESM morphism and $t \in T$, then p and t are similar, denoted $p \sim t$, if*

$$\forall s \in S : \begin{cases} W(s,t) = |\mathrm{Min}(p) \cap \varphi^{-1}(s)| \\ W(t,s) = |\mathrm{Max}(p) \cap \varphi^{-1}(s)| \end{cases}$$

2. *For $g = (V, \varphi)$ a discrete S–graph and M an S–marking, we say that g and M are similar and we write $g \sim M$ if*

$$\forall s \in S : M(s) = |\varphi^{-1}(s)|$$

3. *A similarity between ES and N is a bijection $\eta : P \to T$ such that*
   - *For each $p \in P$ and for each $t \in T$,* *$\eta(p) = t$ if and only if $p \sim t$*
   - *$Init \sim M_0$*

We say that $N = (S, T, W, M_0)$ and $ES = (Init, P)$ are similar (for η) and we write $ES \sim N$ (for η), if there exists a similarity η from ES to N.

It is obvious from the figures 1, 2, 3 and definition 3.6, that the place names of a P/T-net are used as node labels in an ESM system. In this way, the nodes in Init and P get the same meaning as the nodes in processes: they represent tokens. As a result, productions describe how tokens are replaced (rewritten) when they're applied to a configuration.

In the following, it is argued that similarity is strongly connected to the notion of isomorphism by proving that nets, similar to the same system, are isomorphic and vice versa. We start with stating that for P/T-nets (without isolated elements) and a similar Petri system, every place name appears as a label in a production.

Lemma 3.1 *For a Petri net $N = (S, T, W, M_0)$ and a Petri system $ES = (Init, P)$ such that η is a similarity from ES to N,*

$$S = \bigcup_{p \in P} (\varphi(\mathrm{Min}(p)) \cup \varphi(\mathrm{Max}(p)))$$

Proof. Let $t \in T$, $p \in P$ such that $t = \eta(p)$. For any $s \in S$,

$$\begin{array}{lcl} s \in {}^{\bullet}t & \iff & W(s,t) \geq 1 \\ & \iff & |Min(p) \cap \varphi^{-1}(s)| \geq 1 \\ & \iff & s \in \varphi(Min(p)) \end{array}$$

It can be proven in the same way that $s \in t^{\bullet}$ if and only if $s \in \varphi(Max(p))$. The result then follows from $S = \bigcup_{t \in T} ({}^{\bullet}t \cup t^{\bullet})$. □

The previous theorem is valid only in the context of Petri nets without isolated elements. This is because an isolated place in a net, does not occur as node label in one of the productions of a similar system. If we admitted the presence of isolated places in a Petri net, then Petri nets similar to the same system would be isomorphic up to those isolated places (i.e. they could differ in those isolated places only).

Theorem 3.1 *Let $ES = (Init, P)$ be a Petri system and let $N = (S, T, W, M_0)$, $N' = (S', T', W', M_0')$ be Petri nets. Then $ES \sim N$ and $ES \sim N'$ implies N' is isomorphic with N.*

Proof. Suppose ES is similar to N with η' a similarity from ES to N', then

$$\begin{array}{lcl} S & = & \bigcup_{p \in P} (\varphi(Min(p)) \cup \varphi(Max(p))) \\ S' & = & \bigcup_{p \in P} (\varphi(Min(p)) \cup \varphi(Max(p))) \end{array}$$

thus $S = S'$. Now let

$$\begin{array}{lcl} \xi_1 & : & S \to S' \text{ be the identity, and} \\ \xi_2 & = & \eta' \circ \eta^{-1} : T \to T' \end{array}$$

then ξ_1 and ξ_2 obviously are bijections. We now prove that $\xi = (\xi_1, \xi_2)$ is an isomorphism. Let $s \in S$, $t \in T$ and let $s' \in S'$, $t' \in T'$ such that $s' = \xi_1(s) = s$ and $t' = \xi_2(s)$, then

$$\begin{array}{lcl} W(s', t') & = & |Min(p) \cap \varphi^{-1}(s')| \\ & = & |Min(p) \cap \varphi^{-1}(s)| \\ & = & W(s,t) \text{ and one may show that} \\ W(t', s') & = & W(t, s) \text{ in a similar way} \end{array}$$

□

Theorem 3.2 *Let $N = (S, T, W, M_0)$ be a Petri net, and let $ES = (Init, P)$, $ES' = (Init', P')$ be Petri systems. Then $ES \sim N$ and $ES' \sim N$ implies ES' is isomorphic with ES.*

Proof. Assume ES and ES' are similar to N with η and η' similarities from resp. ES and ES' to N. Then $\xi = \eta'^{-1} \circ \eta : P \to P'$ is a bijection. We now prove that for each $p \in P$ and each $p' \in P'$, $\xi(p) = p'$ if and only if $C_p$is isomorphic with C'_p.

Let $C_p = (V, R, \varphi) \in P$ and $C'_p = (V', R', \varphi') \in P'$ such that $\eta(p) = t = \eta'(p')$, then for each $s \in S$, $|Min(p) \cap \varphi^{-1}(s)| = W(s,t) = |Min(p') \cap \varphi'^{-1}(s)|$, and thus $|Min(p)| = |Min(p')|$. By consequence, we can construct a label preserving bijection from $Min(p)$ onto $Min(p')$. For the same reason we can construct another label preserving bijection from $Max(p)$ onto $Max(p')$.

Define $\mu : V \to V'$ the union of these two bijections, then, by the disjunction of $Min(p)$ and $Max(p)$ (resp. $Min(p')$ and $Max(p')$), μ also is a bijection. Now, if $(v_1, v_2) \in R$, then v_1, v_2 are elements of resp. $Min(p)$ and $Max(p)$. As a consequence, $\mu(v_1)$ and $\mu(v_2)$ are elements of resp. $Min(p')$ and $Max(p')$, and thus is $(\mu(v_1), \mu(v_2)) \in R'$.

$Init$ is isomorphic with $Init'$ then follows from the fact that $|\varphi^{-1}(s)| = M_0(s) = |\varphi'^{-1}(s)|$, for each $s \in S$. □

For a given net, we now know that all similar systems are, up to an isomorphism, the same. The *Esm* construction constructs one such system for a given net N that will be similar to N. We then know from theorems 3.2 and 3.3 that every other Petri system ES, similar to N, is isomorphic to $Esm(N)$.

The *Esm* construction essentially creates for every transition t a similar production p, as shown in figure 2. The main problem is to create, for every place, the requested number of *different* nodes. To this aim, nodes will be tuples (s, i) where s is a place and i is a serial number. This structure of the created nodes is however not essential to the ESM construction, it is just an easy way to assure that different nodes have different identities. For example, the production constructed for the transition shown in figure 2 will be the primitive discrete ESM morphism $p : g \rightarrow h$ where

$$
\begin{array}{lll}
V_g = \{ (a,1), (b,1), (b,2) \} & E_g = \emptyset & \varphi_g = \{ ((a,1),a), ((b,1),b), ((b,2),b) \} \\
V_h = \{ (c,1), (d,1) \} & E_h = \emptyset & \varphi_h = \{ ((c,1),c), ((d,1),1) \} \\
p = V_g \times V_h & &
\end{array}
$$

Definition 3.7 (Esm Construction) *Let $N = (S, T, W, M_0)$ be a Petri net. Consider a numbering of the transitions for this net: $T = \{ t_j \mid j \in J \}$ for some $J \subseteq \mathbb{N}$.*

1. *For a transition $t \in T$, define $Esm(t)$ as follows*
   Let $V_g = \bigcup_{s \in {}^\bullet t} \{ (s,i) \mid 1 \le i \le W(s,t) \}$ and $V_h = \bigcup_{s \in t^\bullet} \{ (s,i) \mid 1 \le i \le W(t,s) \}$, then $Esm(t)$ is the discrete ESM morphism $p : g \rightarrow h$ defined by

$$
\begin{array}{ll}
g = (V_g, \emptyset, \varphi_g) & \textit{with} \quad \varphi_g : V_g \rightarrow S : (s,i) \mapsto s \\
h = (V_h, \emptyset, \varphi_h) & \textit{with} \quad \varphi_h : V_h \rightarrow S : (s,i) \mapsto s \\
p = V_g \times V_h &
\end{array}
$$

2. *For an S-marking M, $Esm(M)$ is defined by $Esm(M) = (V, \varphi)$ where*

$$
V = \bigcup_{s \in S} \{ (s,i) \mid 1 \le i \le M(s) \}
$$
$$
\varphi : V \rightarrow S : (s,i) \mapsto s
$$

3. *$Esm(N)$ is defined by $Esm(N) = (Init, P)$ where*

$$
\begin{array}{l}
Init = Esm(M_0) \\
P = \{ p_j \mid j \in J \} \textit{ where } p_j = Esm(t_j),\ j \in J
\end{array}
$$

Another remark is needed here. If we consider a net N that is not simple, i.e. there exist $t, t' \in T_N$ such that ${}^\bullet t = {}^\bullet t'$ and $t^\bullet = t'^\bullet$, then the *Esm* construction would create for t and t' identical productions p and p', because $Esm(t)$ is only determined by ${}^\bullet t$ and $t^\bullet$. This would result in $Esm(N) = (Init, P)$ not being similar to N because P contains only one production for the transitions t and t', which implies that there doesn't exist a bijection between P and T_N. This situation can however be solved easily by assuring that all constructed productions use different nodes, which can be done by including the transitions' identity into the nodes' identity. This way, a node would be a triple $(s, t, i) \in S \times T \times \mathbb{N}$. Such an "improvement" however, would render the *Esm* construction more complicated and would give the impression that the transitions' identities somehow have to be coded into the process structures.

Lemma 3.2 *For a net $N = (S, T, W, M_0)$, $Esm(N)$ is a Petri system.*

Proof. Let $ES = (Init, P) = Esm(N)$. Each $p \in P$ clearly is a primitive discrete ESM morphism and $Init$ is a discrete Σ—graph. □

Theorem 3.3 *Let $N = (S, T, W, M_0)$ be a Petri net, then $Esm(N) \sim N$.*

Proof. Suppose $T = \{ t_j \mid j \in J \}$ and let $ES = Esm(N)$ with $ES = (Init, P)$, then $P = \{ Esm(t_j) \mid j \in J \}$. Let $p_j = Esm(t_j), j \in J$. Now, let $\eta : P \to T : p_j \mapsto t_j$ (this clearly is a bijection). We prove that $Init \sim M_0$ and $\eta(p_j) = t_j$ if and only if $p_j \sim t_j$.

1. Let $Init = (V, \varphi)$, $s \in S$, then

$$\varphi^{-1}(s) = \{ (s', i) \in V \mid s' = s \} = \{ (s, i) \mid 1 \leq i \leq M_0(s) \}$$

   thus $|\varphi^{-1}(s)| = M_0(s)$.

2. $\eta(p) = t$ if and only if $p = p_i$ and $t = t_j$ for some $j \in J$. Then for each $s \in S$,

$$\begin{array}{lclcl} |Min(p) \cap \varphi^{-1}(s)| & = & |\{ (s, i) \mid 1 \leq i \leq W(s, t) \}| & = & W(s, t) \\ |Max(p) \cap \varphi^{-1}(s)| & = & |\{ (s, i) \mid 1 \leq i \leq W(t, s) \}| & = & W(t, s) \end{array}$$

   but then $p \sim t$. □

The Petri construction is the inverse of the Esm construction in that it constructs a Petri net out of an ESM system. This is done by creating for every production p a transition t. The thus constructed Petri net will be similar to the given Petri system (Theorem 3.4). Again, it then follows from theorem 3.1 that every Petri net similar to the original system, is isomorphic to the constructed one.

Definition 3.8 (Petri construction) *Let ES = (Init,P) be a Petri system. Consider a numbering of the productions of ES: $P = \{ p_j \mid j \in J \}$ for some $J \subseteq I\!N$, and let, for each $j \in J$, $p_j = (V_j, R_j, \varphi_j)$.*

1. *For a configuration $g = (V, \varphi) \in Conf(ES)$, define $M = Petri(g)$ as*

$$M(s) = |\varphi^{-1}(s)|$$

2. *$N = (S, T, W, M_0) = Petri(ES)$ is constructed as follows: choose, for every $j \in J$, a transition t_j,*

$$\begin{array}{l} T = \{ t_j \mid j \in J \} \\ S = \bigcup_{j \in J} \varphi_j(V_j) \\ M_0 = Petri(Init) \\ W : (S \times T) \cup (T \times S) \to I\!N : \left\{ \begin{array}{lcl} (s, t_j) & \mapsto & |\mathrm{Min}(p_j) \cap \varphi_j^{-1}(s)| \\ (t_j, s) & \mapsto & |\mathrm{Max}(p_j) \cap \varphi_j^{-1}(s)| \end{array} \right. \end{array}$$

It is obvious from the construction that for every ESM system ES, $Petri(ES)$ is a Petri net.

Theorem 3.4 *Let $ES = (Init, P)$ be a Petri system, then $ES \sim Petri(ES)$.*

Proof. Suppose $P = \{ p_j \mid j \in J \}$ and let $N = (S, T, W, M_0) = Petri(ES)$. Then $T = \{ t_j \mid j \in J \}$. Now, let $\eta : P \to T : p_j \mapsto t_j$ (this clearly is a bijection). We prove that $Init \sim M_0$ and $\eta(p_j) = t_j \Leftrightarrow p_j \sim t_j$.

1. $M_0 = Petri(Init)$ implies $M(s) = |\varphi^{-1}(s)|, \forall s \in S$. Thus $Init \sim M_0$.

2. $\eta(p) = t$ if and only if $p = p_j$ and $t = t_j$ for some $j \in J$. Consider $p_j \in P$ and $t_j \in T$ for some $j \in J$, then by construction we have, for each $s \in S$

$$\begin{aligned} W(s, t_j) &= |Min(p_j) \cap \varphi_j^{-1}(s)| \\ W(t_j, s) &= |Max(p_j) \cap \varphi_j^{-1}(s)| \end{aligned}$$

which shows that $p_j \sim t_j$. □

We have now shown that it is possible to define a structural similarity between P/T-nets and Petri systems that is closely related to isomorphism. We will now do the same for the processes of a net and the process structures of a system.

3.3 Processes and process structures

Analogously to the way that Petri nets and Petri systems are related to each other, we can relate processes with process structures using their (structural) similarity and by *Esm* and *Petri* constructions. Consider the process shown in Figure 4, which is a process of the net in Figure 3. A process of a Petri net describes how the tokens are replaced from one place to another and which transitions have been used to accomplish this. In ESM systems, the first is done by computation structures, who describe a rewriting history of an ESM system. The second one, describing the productions that have been used during this rewriting process, is not contained within a computation structure but is described by a covering which maps productions into the computation structure.

A similarity will thus be defined as a tuple of bijections between a process π and a process structure C together with its covering, one bijection between the conditions of π and the nodes of C and another between the events of π and the elements of the covering, as is shown in Figure 4.

A process is represented graphically by circles and boxes connected by directed edges representing respectively conditions, events and the flow relation. The names are now labels. A process structures is drawn as circles, representing nodes, labelled with their node labels connected with vertical lines directed from top to bottom, who represent the ESM morphism. The covering is represented graphically by drawing rectangles labelled with the productions used at that place in the computation or process structure.

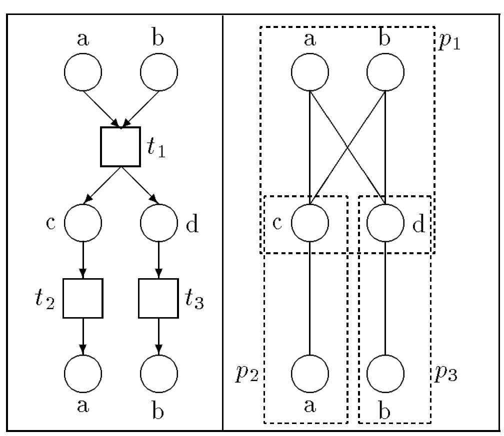


Figure 4: A process and a similar process structure

Definition 3.9 (similarity of processes and process structures)
Let $N = (S, T, W, M_0)$ be an Petri net, $ES = (Init, P)$ a Petri system, and let $\pi = (B, E, F, \nu) \in \Pi(N)$, $C = (V, R, \varphi) \in IComp(ES)$ such that $cov = (p_i, f_i)_I$ be a P–covering of C, Then μ is a similarity from (C, cov) to π if and only if $\mu = (\mu^c, \mu^i)$ where

$$\begin{array}{llcll} \mu^c & : & V & \longrightarrow & B \quad \text{a label preserving bijection} \\ \mu^i & : & I & \longrightarrow & E \quad \text{a bijection} \end{array}$$

such that

$$\forall i \in I : \left\{ \begin{array}{lcl} {}^{\bullet}\mu^i(i) & = & \mu^c(f_i(\mathrm{Min}(p_i))) \\ \mu^c(i)^{\bullet} & = & \mu^c(f_i(\mathrm{Max}(p_i))) \end{array} \right.$$

Whenever there exists a similarity μ between a process structure with a covering (C, cov) and a process π, we say that (C, cov) is similar to π and write $(C, cov) \sim \pi$. For a set of processes Π and a set of process structures $\mathcal{C}$, we say that $\mathcal{C}$ is similar to Π if

- for each $C \in \mathcal{C}$ there exists a $\pi \in \Pi$ such that C is similar to π, and
- for each $\pi \in \Pi$ there exists a $C \in \mathcal{C}$ such that C is similar to π.

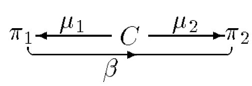
For the remaining paragraphs within this section, $N = (S, T, W, M_0)$ will be a Petri net and $ES = (Init, P)$ a Petri system such that η is a similarity from ES to N.

Just as we did in the previous section, we can prove the connection between isomorphism and similarity of processes and process structures (with coverings), i.e. processes similar to the same process structure (with covering) are isomorphic and process structures (with coverings) similar to the same process also are isomorphic. Both statements are proved by composing the similarities μ_1 and μ_2 to an isomorphism.

Theorem 3.5 *Let π_1, π_2 be processes of N and let C be a process structure of ES with a valid P–covering $cov = (f_i, p_i)_I$ of C.*

If $(C, cov) \sim \pi_1$ and $(C, cov) \sim \pi_2$ then π_1 is isomorphic to π_2.

Proof. Let $\mu_1 = (\mu_1^c, \mu_1^i)$ and $\mu_2 = (\mu_2^c, \mu_2^i)$ be similarities from (C, cov) to π_1 and (C, cov) to π_2, respectively, where $\pi_1 = (B_1, E_1, F_1, \nu_1)$, $\pi_2 = (B_2, E_2, F_2, \nu_2)$, $C = (V, R, \varphi)$ and $cov = (f_i, p_i)_I$, as usual.



Let $\beta^b = \mu_2^c \circ {\mu_1^c}^{-1}$ be a mapping from B_1 to B_2, and let $\beta^e = \mu_2^i \circ {\mu_1^i}^{-1}$ be a mapping from E_1 to E_2. Then B^b and B^e are obtained by composing bijections, and thus are bijections themselves. By consequence, $\beta = \beta^b \cup \beta^e$ is a bijection because β^b and β^e have disjoint domains and codomains. This way, β automatically fullfils the properties $x \in B_1 \Leftrightarrow \beta(x) \in B_2$ and $x \in E_1 \Leftrightarrow \beta(x) \in E_2$.

Thus it only remains to be verified that $(x, y) \in F_1$ if and only if $(\beta(x), \beta(y)) \in F_2$. Now $(x, y) \in F_1$ implies either $x \in B_1$ and $y \in E_1$ or $x \in E_1$ and $y \in B_1$. We'll consider the first case (the second is analoguous).

Take $v \in C$ and $i \in I$ such that $\mu_1^c(v) = x$ and $\mu_1^i(i) = y$, then

$$\begin{array}{lcl} (\beta(x), \beta(y)) \in F_2 & \Leftrightarrow & \beta(x) \in {}^{\bullet}\beta(y) \\ & \Leftrightarrow & \mu_2^c(v) \in {}^{\bullet}\mu_2^i(i) \\ & \Leftrightarrow & \mu_2^c(v) \in \mu_2^c(f_i(Min(p_i))) \\ & \Leftrightarrow & v \in f_i(Min(p_i)) \\ (x, y) \in F_1 & \Leftrightarrow & x \in {}^{\bullet}y \\ & \Leftrightarrow & \mu_1^c(v) \in {}^{\bullet}\mu_1^i(i) \\ & \Leftrightarrow & \mu_1^c(v) \in \mu_1^c(fi(Min(p_i))) \\ & \Leftrightarrow & v \in f_i(Min(p_i)) \end{array}$$

Thus, $(x, y) \in F_1$ if and only if $(\beta(x), \beta(y)) \in F_2$. □

Theorem 3.6 *Let π be a process of N and let C, C' be process structures of ES with valid P–coverings cov and cov' of respectively C and C'.*

If $(C, cov) \sim \pi$ and $(C', cov') \sim \pi$ then C is isomorphic to C'.

Proof. Let $\mu_1 = (\mu_1^c, \mu_1^i)$ and $\mu_2 = (\mu_2^c, \mu_2^i)$ be similarities from (C, cov) to π and (C', cov') to π, respectively, where $\pi = (B, E, F, \nu)$, $C = (V, R, \varphi)$, $C' = (V', E', \varphi')$ and $cov = (f_i, p_i)_I$, $cov' = (f_j', p_j')_J$, as usual.

$$C \xrightarrow{\mu_1} \pi \xleftarrow{\mu_2} C' \qquad C \xrightarrow{\gamma} C'$$

Let $\gamma^c = \mu_2^{c^{-1}} \circ \mu_1^c$ be a mapping from V to V', and let $\gamma^i = \mu_2^{i^{-1}} \circ \mu_1^i$ be a mapping from I to J. then γ^c is a label preserving bijection from C to C' and γ^i is a bijection from I to J since they are obtained by composing such bijections. We now show that C is γ^c–isomorphic to C', for which it is sufficient to show, under the restrictions of Petri systems, that $\gamma^c(R) = R'$.

Now, take $(x, y) \in f_i(R_i)$, then for each $i \in I$ and $j \in J$ such that $\gamma^i(i) = j$,

$$\begin{aligned}
(x, y) \in f_i(R_i) \quad &\Leftrightarrow \quad \begin{cases} x \in f_i(Min(p_i)) \\ y \in f_i(Max(p_i)) \end{cases} && R_i = Min(p_i) \times Max(p_i) \\
&\Leftrightarrow \quad \begin{cases} \mu_1^c(x) \in \mu_1^c(f_i(Min(p_i))) = {}^\bullet\mu_1^i(i) \\ \mu_1^c(y) \in \mu_1^c(f_i(Max(p_i))) = \mu_1^i(i)^\bullet \end{cases} \\
&\Leftrightarrow \quad \begin{cases} \mu_1^c(x) \in {}^\bullet\mu_2^i(j) = \mu_2^c(f_j'(Min(p_j'))) \\ \mu_1^c(y) \in \mu_2^i(j)^\bullet = \mu_2^c(f_j'(Max(p_j'))) \end{cases} && i = (\mu_1^{i^{-1}} \circ \mu_2^i)(j) \\
&\Leftrightarrow \quad \begin{cases} \gamma^c(x) \in f_j'(Min(p_j')) \\ \gamma^c(y) \in f_j'(Max(p_j')) \end{cases} \\
&\Leftrightarrow \quad (\gamma^c(x), \gamma^c(y)) \in f_j'(R_j')
\end{aligned}$$

By consequence, $\gamma^c(f_i(R_i)) = f_j'(R_j')$ and thus $\gamma^c(R) = R'$ since $R = (\cup_{i \in I} f_i(R_i))^*$ and $R' = (\cup_{j \in J} f_j'(R_j'))^*$. □

We now turn to the construction of a similar process structure (with covering) for a given process, the *Esm* construction. Since there exists a label preserving bijection between the conditions of a process π and the nodes of a similar process structure C (this also is obvious from figure 4), we can just take the condition identities as nodes of $Esm(\pi)$ together with their labels, and use the identity relation as a label preserving bijection. The *Esm* construction then creates, for every event e, an element (f, p) where p is the production similar to the transition corresponding with the event, formally $p = \eta^{-1}(\nu(e))$, and where f maps C_p on the nodes in C rewritten by p.

Definition 3.10 (Esm construction) *Let $\pi = (B, E, F, \nu) \in \Pi(N)$ and consider a numbering of its event nodes: $E = \{ e_1, e_2, \ldots, e_m \}$. Then $Esm(\pi)$ is defined by $Esm(\pi) = (C, cov)$ where $C = (V, R, \varphi)$ and $cov = (f_i, p_i)_I$ with $C_{p_i} = (V_i, R_i, \varphi_i)$ such that*

1. *V=B. For the purpose of clarity, we will often use $id_V : B \to V$ the identity.*
2. $\varphi = \nu|_B$
3. *Let $I = \{ 1, 2, \ldots, m \}$, then for each $i \in I$, $p_i = \eta^{-1}(\nu(e_i))$ and $f_i : C_{p_i} \to C$ such that*

$$\begin{aligned}
f_i(\mathrm{Min}(p_i)) \quad &= \quad id_V({}^\bullet e_i) \\
f_i(\mathrm{Max}(p_i)) \quad &= \quad id_V(e_i{}^\bullet) \\
f_i(v_1) = v_2 \quad &\Leftrightarrow \quad \varphi_i(v_1) = \varphi_i(v_2)
\end{aligned}$$

4. $R = (\cup_{i \in I} f_i(R_i))^*$

The first thing to be verified is that the *Esm*-construction of a process (of N), yields a computation structure (of ES). To this, we prove that the constructed covering is a valid P-covering of C. Then, we verify that the constructed process structure and covering are indeed similar to the process. It then follows from theorem 3.6 that every other similar process structure is isomorphic to the constructed one.

Lemma 3.3 *For π a process of N, $Esm(\pi) \sim \pi$.*

Proof. Let $\pi = (B, E, F, \nu)$ and let $Esm(\pi) = (C, cov)$ where $C = (V, R, \varphi)$ and $cov = (f_i, p_i)_I$. Reconsider the numbering of the events used in the Esm construction.

Define $\mu = (\mu^c, \mu^i)$ as $\mu^c = id_V$ and $\mu^i : I \to E : i \mapsto e_i$, then $\mu^c(f_i(Min(p_i))) = id_V^{-1}(id_V({}^\bullet e_i)) = {}^\bullet e_i = {}^\bullet\mu^i(i)$ and $\mu^c(f_i(Max(p_i))) = id_V^{-1}(id_V(e_i{}^\bullet)) = e_i{}^\bullet = \mu^i(i)^\bullet$ □

Lemma 3.4 *For π a process of N, $Esm(\pi)$ is an element of $IComp(ES)$*

Proof. Let $Esm(\pi) = (C, cov)$ where $\pi = (B, E, F, \nu)$, $C = V, R, \varphi)$ and $cov = (f_i, p_i)_I$ such that $C_{p_i} = (V_i, R_i, \varphi_i)$. We subsequently prove that cov is a valid P–covering of C and that $Min(C)$ is isomorphic with $Init$. cov is a P–covering of C because

1. All p_i clearly are elements of P and all f_i are CS morphisms from C_{p_i} into C.
2. This follows from the non-branched places property of occurence nets, i.e. for any $b \in B$, $|{}^\bullet b| \leq 1$ and $|b^\bullet| \leq 1$.

The covering then is valid by the Esm construction since $R = (\cup_{i\in I} f_i(R_i))^*$.

$Min(C) = Min(\pi)$, thus $M_0(s) = |Min(C) \cap \varphi^{-1}(s)|$ for each $s \in S$. $Init \sim M_0$, thus $M_0(s) = |Init \cap \varphi^{-1}(s)|$ for each $s \in S$. By consequence, there exists a label preserving bijection between $Min(C)$ and $Init$, thus they are isomorphic (since they are discrete graphs). □

The inverse operation is performed by the $Petri$ construction. As is done in the Esm construction, we will take the set of conditions to be the set of nodes of C. The $Petri$ construction then creates for every element (f, p) in the covering, an event node e such that ${}^\bullet e = f_i(Min(p_i))$ and $e^\bullet = f_i(Max(p_i))$.

Definition 3.11 (Petri construction) *Let $C \in IComp(ES)$ and cov be a P-covering of C, where $C = (V, R, \varphi)$ and $cov = (p_i, f_i)_I$. Define $Petri(C, cov)$ to be (B, E, F, ν) such that*

1. *$B = V$. For the purpose of clarity, we will often use $id_B : V \to B$ the identity.*
2. *Let E be a set containing $|I|$ event nodes. Consider a numbering of this event nodes such that E can be written as: $E = \{ e_i \mid i \in I \}$.*
3. *Define* $pre : I \to 2^B : i \mapsto id_B(f_i(\mathrm{Min}(p_i)))$
   $post : I \to 2^B : i \mapsto id_B(f_i(\mathrm{Max}(p_i)))$

   $$F = \bigcup_{i\in I} ((pre(i) \times \{ e_i \}) \cup (\{ e_i \} \times post(i)))$$
4. $\nu|_B : B \to S : b \mapsto \varphi(b)$
   $\nu|_E : E \to T : e_i \mapsto \mu(p_i)$

Again, $Petri(C, cov)$ constructs a process of N (the verification is left to the reader) that is similar to (C, cov), as is shown in the following lemma.

Lemma 3.5 *For C a process structure of ES with a valid P–covering cov, $(C, cov) \sim Petri(C, cov)$.*

Proof. Let $Petri(C, cov) = (B, E, F, \nu)$ where $C = (V, R, \varphi)$ and $cov = (f_i, p_i)_I$ such that $C_{p_i} = (V_i, R_i, \varphi_i)$. Reconsider the numbering of the event nodes in the $Petri$ construction.

Define $\mu = (\mu^c, \mu^i)$ as $\mu^c = id_B$ and $\mu^i : I \to E : i \mapsto e_i$, then ${}^\bullet\mu^i(i) = {}^\bullet e_i = pre(i) = id_B(f_i(Min(p_i))) = \mu^c(f_i(Min(p_i)))$. The equality $\mu^i(i)^\bullet = \mu^c(f_i(Max(p_i)))$ is shown analogously. □

It then follows again from this lemma and theorem 3.5 that every process similar to (C, cov) is isomorphic with $Petri(C, cov)$.

4 Petri net and Petri system semantics

In this section, it is shown that the process structures yield the same semantics for Petri systems as processes do for petri nets. For this, it is proved that two nets have the same sets of processes if ans only if similar systems have the same process structures, and that similar nets and systems have similar sets of processes and process structures. This last situation is shown in Figure 5.

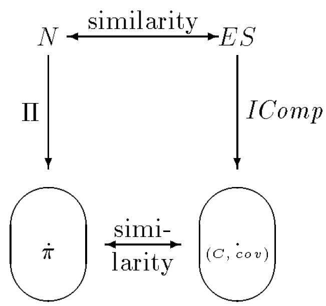


Figure 5: Similar nets and systems have similar processes and process structures

It are these observations that justify the use of process structures as a Petri net semantics and using processes as a Petri system semantics. Moreover, it is possible to use processes based semantics, such as bisimilarities, for Petri systems and, conversely, to use process structures based semantics, such as the external effect semantics, for Petri nets. The first one is studied in more detail by J.Delcour [Del93], the second one will be shown by examples in the second part of this section.

4.1 Processes and process structures

Theorem 4.1 *Let N_1, N_2 be Petri nets and let ES_1, ES_2 be Petri systems such that N_1 is similar to ES_1 and N_2 is similar to ES_2.*

Then, $\Pi(N_1) = \Pi(N_2)$ if and only if $IComp(ES_1) = ICompES_2$.

Proof.

$\boxed{\Rightarrow}$ Let $C_1 \in IComp(ES_1)$, then $Petri(C_1) \in \Pi(N_1) = \Pi(N_2)$ thus $Esm(Petri(C_1)) \in IComp(ES_2)$ and thus $C_1 \in IComp(ES_2)$ because $Esm(Petri(C_1))$ is isomorphic to C_1.

$\boxed{\Leftarrow}$ analoguous. □

We now prove that similar nets and systems have similar sets of processes and process structures (every process is similar to a process structure and vice versa), as is shown in Figure 5.

Theorem 4.2 *Let N be a Petri net and let ES be a Petri system. If ES is similar to N, then $\Pi(N)$ is similar to $IComp(ES)$.*

Proof. We have to prove that there exists, for every $C \in IComp(ES)$, a process $\pi \in \Pi(N)$ such that C is similar to π and vice versa.

For the first, $Petri(C)$ is a process in $\Pi(N)$ and for the second, $Esm(\pi)$ is a process structure in $IComp(ES)$. □

4.2 External effect for Petri nets

We will first prove some properties about the external effect in general, to simplify the study of equivalence of both Petri and ESM systems. The main property to show is that two systems are equivalent if the productions of one system can be obtained as the effect of the other system and vice versa.

First, since $\mathit{Eff}(P)$ (for an ESM system P) can be derived inductively, we have following properties:

Lemma 4.1 *Let P, P' be two ESM systems, then*

1. $P \subseteq \mathit{Eff}(P') \Rightarrow \mathit{Eff}(P) \subseteq \mathit{Eff}(P')$
2. $P \subseteq \mathit{Eff}(P') \Rightarrow \mathit{Eff}(P \cup P') \subseteq \mathit{Eff}(P')$

Theorem 4.3 *Let P_1, P_2 be two ESm systems, then $\mathit{Eff}(P_1) = \mathit{Eff}(P_2)$ if and only if $P_1 \subseteq \mathit{Eff}(P_2)$ and $P_2 \subseteq \mathit{Eff}(P_1)$.*

Proof. Suppose $\mathit{Eff}(P_1) = \mathit{Eff}(P_2)$, then $P_1 \subseteq \mathit{Eff}(P_1) = \mathit{Eff}(P_2)$ and $P_2 \subseteq \mathit{Eff}(P_2) = \mathit{Eff}(P_1)$.

Conversely, suppose $P_1 \subseteq \mathit{Eff}(P_2)$ and $P_2 \subseteq \mathit{Eff}(P_1)$. Then $P_1 \subseteq \mathit{Eff}(P_2)$ implies $\mathit{Eff}(P_1) \subseteq \mathit{Eff}(P_2)$ and $P_2 \subseteq \mathit{Eff}(P_1)$ implies $\mathit{Eff}(P_2) \subseteq \mathit{Eff}(P_1)$, which completes the proof. □

So in order to prove that two systems have the same effects, it suffices to show that the productions of one system can be generated by the other and vice versa. For general ESM systems, these properties are not very important since $\mathit{Comp}(P)$ is not compositional with respect to the composition of computations, and thus $\mathit{Eff}(P)$ isn't too. For Petri systems however, $\mathit{Comp}(P)$ actually is a compositional semantics with respect to the union of Petri systems (if we consider nets without an initial marking), since conditional computation structures degenerate to just computation structures for this kind of ESM systems. This is because the conditions, in a conditional computation structure, describe the way edges have to be rewritten by the context to "enable" a computation structure. Since we do not have edges in a Petri system, the role of these conditions disappear. Thus, we can introduce a composition operator for Petri nets such that $\mathit{Comp}(N)$ and $\mathit{Eff}(N)$ are compositional for this operator. The composition of two Petri nets N_1 and N_2 then is the Petri net that is similar to $\mathit{Esm}(N_1)$ and $\mathit{Esm}(N_1)$, or $\mathit{Petri}(\mathit{Esm}(N_1) \cup \mathit{Esm}(N_2))$.

We will now consider two Petri nets N_1 and N_2 and question wether they have the same external effect or not, and how this differs from transition oriented Petri net semantics (e.g. bisimulation). We will do this by studying an example and counterexample.

Figure 6 shows two Petri nets N_1 and N_2 together with similar Petri systems P_1 and P_2. Figure 7 then shows how the productions of P_1 can be obtained as the effect of computations in P_2 and vice versa. This proves that N_1 and N_2 have the same external effect. Intuitively, this can be explained by observing that b,c and d form equivalent places because a token, in one of these places, can always evolve to a token in another place. By consequence, it does not matter where the transitions t_1 and t'_1 place their output tokens. Using bisimilarity, this effect could only be obtained by making the transitions t_2, t_3, t_4 and t'_1, t'_2, t'_3 silent.

Conversely, consider the Petri nets N'_1, N'_2 with their similar Petri systems P'_1, P'_2 in Figure 8. Both nets have the same firing properties, and will thus be equivalent for bisimulations if the corresponding transitions have the same labels. For the external effect, however, it is impossible for N'_1 to produce a token in a place labelled e, thus N'_1 and N'_2 differ for the external effect semantics (productions p'_1 and p'_3 cannot be obtained as the effect of system P'_1).

If we would like to consider N'_1 and N'_2 (or P'_1 and P'_2) equivalent, we would have to define a more abstract semantics than the external effect (e.g. one that does not consider nodes labelled e). Some research still has to be done in this direction.

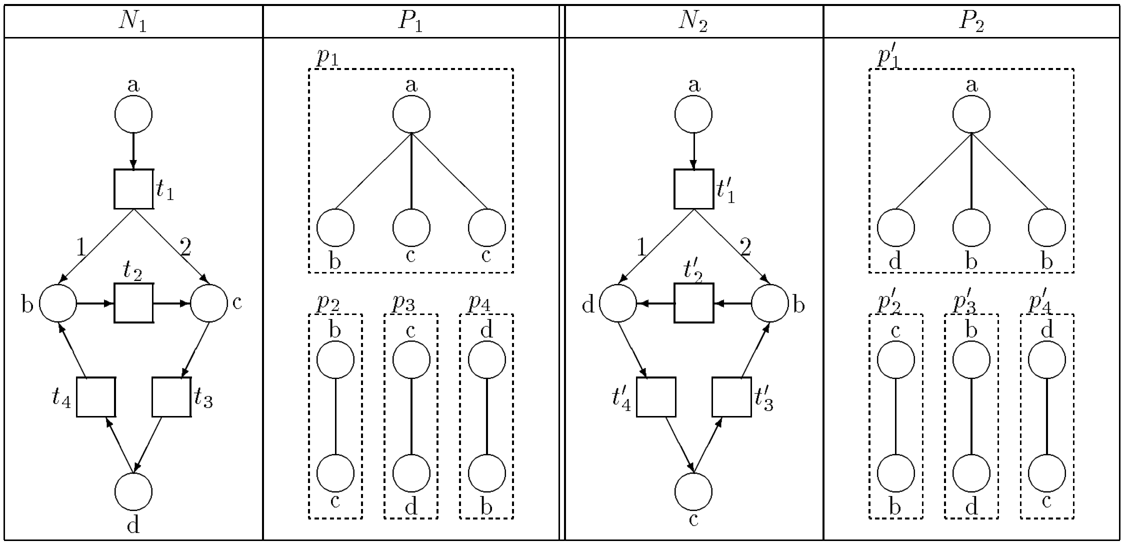


Figure 6: example nets and systems

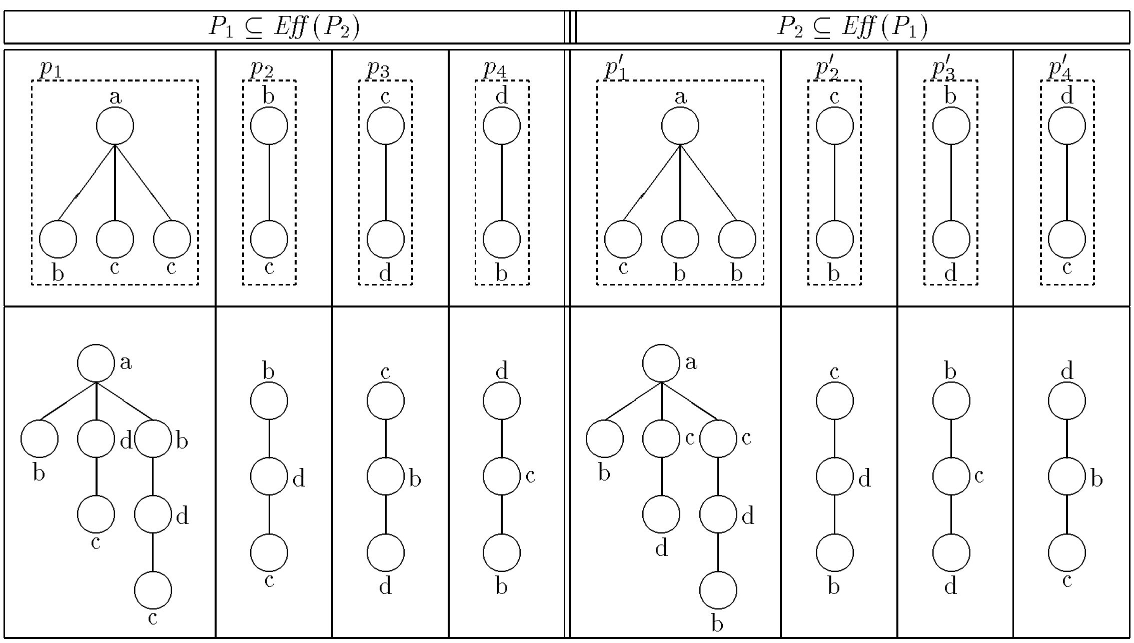


Figure 7: Productions (above) with their corresponding computations (below).

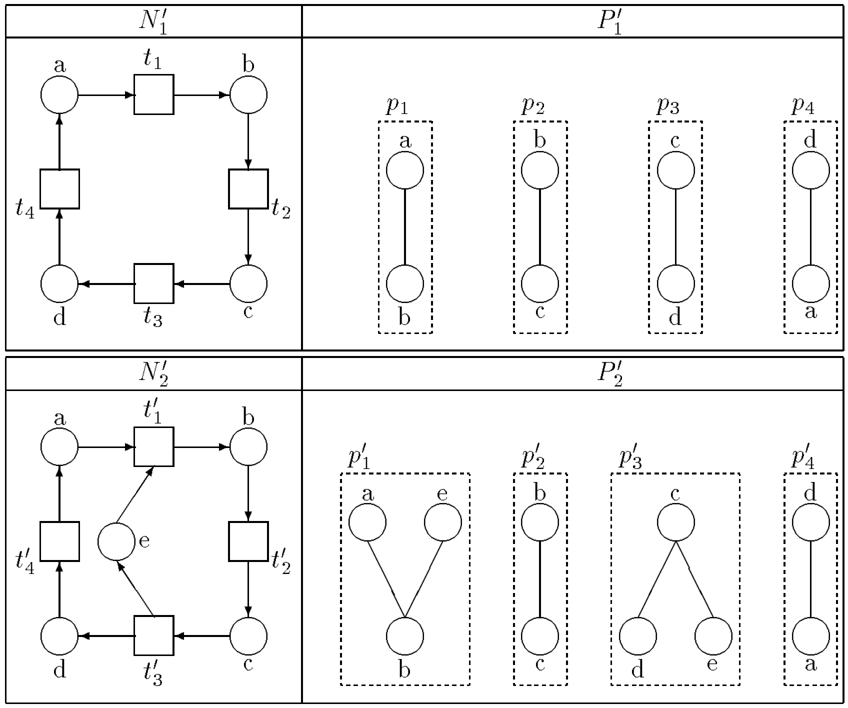


Figure 8: counterexample nets and systems

5 Conclusion

As is claimed in the introduction, we have shown that P/T-nets correspond to ESM systems that operate on discrete graphs and that processes then correspond to process structures. We can now argue that, on the one hand, computation structures play the same role for ESM systems as processes do for Petri nets, i.e. they give a detailed description of the computations of a system. On the other hand, we can use the external effect as a more abstract semantics of Petri nets, which is a compositional semantics for Petri systems and thus for Petri nets. Also, this approach shows that ESM systems are essentially Petri nets with relations between the "tokens" (nodes in ESM terminology).

Furthermore, it has been shown in [Del93] that similar constructions can be defined for labelled P/T-nets and that their concurrent bisimilarity [BDKP91] can be easily expressed using labelled Petri systems and their external effect. It should still be investigated how other bisimilarities (such as FC-bisimilarity) can be expressed by labelled Petri systems, and how bisimilarity can be used for general ESM systems.

References

[BDKP91] E. Best, R. Devillers, A. Kiehn, and L. Pomello. Concurrent bisimulations in petri nets. *Acta Informatica*, 28:231–264, 1991.

[Del93] J. Delcour. Simulatie en bisimulatie voor ESM systemen. Master's thesis, Vrije universiteit Brussel, 1993.

[EKL90] H. Ehrig, M. Korff, and M. Löwe. Tutorial introduction to the algebraic approach of graph grammars based on double and single pushouts. *Lecture Notes in Computer Science*, 532:24–37, 1990.

[EL91] H. Ehrig and M. Löwe. Parallel and distributed derivations in the single-pushout approach. *Theoretical Computer Science*, 109:123–144, March 1991.

[Jan90] D. Janssens. Structured transformations and computation graphs for actor grammars. *Lecture Notes in Computer Science*, 532:446–460, 1990.

[Jan91] D. Janssens. Equivalence of computations in actor grammars. *Theoretical Computer Science*, 109:145–180, March 1991.

[Jan93] D. Janssens. ESM systems and the composition of their computations. In *Graph transformations in Computer Science, Proceedings of Dagstuhl workshop 1993*, September 1993.

[Löw91] M. Löwe. Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109:181–224, March 1991.

[Rei85] W. Reisig. *Petri Nets: An introduction.* Springer-Verlag, 1985.