Equivalence notions for design of concurrent systems using Petri nets *

Igor V. Tarasyuk

Institute of Informatics Systems,
Siberian Division of the Russian Academy of Sciences,
6, Lavrentieva ave.,
630090, Novosibirsk, Russia
e-mail: itar@iis.nsk.su

November 3, 2003

Abstract

The paper is devoted to the investigation of equivalence notions used to abstract from concrete *behavioural* aspects of concurrent systems which are modelled by Petri nets. The basic behavioural equivalences known from the literature are supplemented by new ones to obtain the complete picture and examined for all class of nets as well as for some of their subclasses: sequential nets (nets without concurrency) and strictly labelled nets (which are isomorphic to nonlabelled nets). For top-down design of concurrent systems refinement is used which is the inverse operator to abstraction of concrete *structure* of such systems. An important property of equivalence notions is their preservation by refinements which guarantees equivalence of systems to be modelled on different levels of abstraction. All of considered equivalence notions are checked for preservation by refinements.

Keywords & phrases: Petri nets, sequential nets, strictly labelled nets, behavioural equivalences, bisimulation, refinement.

1 Introduction

By now, concurrency theory is a quickly developing branch of computer science. It is stimulated by practice, where importance of proper understanding of behaviour of concurrent systems became obvious over the last decades.

For specification of concurrent systems and analysis of their behavioural properties a number of formal models were proposed. In [17] the relationship between basic formal models for concurrency was established. The choice of model corresponds to the choice of level of abstraction from concrete *behavioural* aspects of systems to be modelled.

An alternative approach consists in fixing of some powerful formal model and using equivalence notions for such a model to "merge" systems with similar behaviour. So, the choice of equivalence notion will correspond to the choice of abstraction level.

Petri nets [13] are popular formal model for design of concurrent and distributed systems. As mentioned in [22], the main advantage of Petri nets is their ability for structural characterization of three fundamental features, i.e. causality, nondeterminism and concurrency.

In recent years, a wide range of semantic equivalences was proposed in concurrency theory. A lot of them were either directly defined or transferred from other formal models to the framework of Petri nets. The following basic notions of behavioural equivalences for Petri nets are known from the literature.

- *Trace equivalences* (which respect only protocols of nets functioning): interleaving [9], step [15] and pomset [8].
- *(Usual) bisimulation equivalences* (which respect branching structure of nets functioning): interleaving [12], step [11], partial word [23], pomset [6] and process [2].
- *ST-bisimulation equivalences* (which respect the duration of transition occurrences in nets functioning): interleaving [8], partial word [23] and pomset [23].
- *History preserving bisimulation equivalences* (which respect the "past" or "history" of nets functioning): pomset [16] one was proposed.

*The work is supported by Volkswagen Fund, grant N I/70 564 and Russian Foundation for Basic Research, grant N 96-01-01655

- *Conflict respecting equivalences* (which fully respect conflicts in nets): occurrence [8] one was presented.
- *Isomorphism* (i.e. coincidence of nets up to renaming of places and transitions) is a well-known concept.

These equivalence notions may be positioned on coordinate plane with axes ranged over: interleaving – process semantics and trace equivalences – isomorphism. But the resulting picture is uncomplete, and there are some "holes" in it. To complete this picture, we introduce a number of new equivalence notions. These are: partial word and process trace equivalences, process ST-bisimulation equivalence, partial word and process history preserving bisimulation equivalences [18, 19, 20, 21], prime event structure equivalence.

To use these equivalence notions for the choice of abstraction level when modelling concurrent systems, it is important to understand firstly the interrelation of the notions on whole class of Petri nets. In addition, if we intend to model systems with some restrictions, it is worth checking interrelation of the equivalences on subclasses of nets which correspond to such systems. If some equivalence notions merge on some subclass of Petri nets, we will be able to simplify checking of such nets by equivalence. For example, if some strict and weak equivalence notions coincide on some subclass of nets, it is easier to check two nets from the subclass by weak equivalence and conclude that the nets are equivalent in strict sence.

In this paper all equivalelences mentioned above are compared, and their correlation is established on whole class of Petri nets as well as on some of their subclasses: sequential nets (where no transitions can fire concurrently), which model sequential systems, having no concurrent components, and strictly labelled nets (where all transitions have different labelling, and such nets may be considered as nonlabelled), which model systems, all components of which serve different functions (or execute different actions).

For top-down design of concurrent systems *refinement* operator is used. It is inverse to the abstraction of concrete *structure* of systems. When we apply refinement, some elementary components of the systems became having some internal structure, i.e. we consider such systems on lower abstraction level as a result. Very important property of behavioural equivalences is their preservation by refinements, which guarantees that equivalent systems remain equivalent after applying the same refinement operator to them, i.e. that their equivalence is preserved at lower abstraction level. In [5] *SM-refinement* operator for Petri nets was proposed, which replaces transitions of nets by the net of special kind: SM-net. As it was mentioned, in spite of its simplicity, the refinement operator comprises renaminging, simple splitting and simple choice.

In this paper all considered behavioural equivalences are checked for preservation by SM-refinements.

The paper is organized as follows. In Section 2 the basic definitions are presented. In Section 3 we introduce behavioural equivalence notions. Section 4 is devoted to the investigation of the equivalences on whole class of Petri nets, and Section 5 — on two of their subclasses: sequential and strictly labelled nets. Preservation of the equivalence notions by refinements is investigated in Section 6. Concluding Section 7 contains a short overview of the results obtained and some directions of further research.

Let us note that long proofs are omitted in this paper.

2 Basic definitions

2.1 Multisets

Let X be some set. A *multiset* M over X is a mapping $M : X \to \mathbf{N}$, where $\mathbf{N}$ is a set of natural numbers. For $x \in X$, $M(x)$ is a *multiplicity* x in M. We write $x \in M$ if $M(x) > 0$.

When $\forall x \in X\ M(x) \leq 1$, M is a proper set. M is *finite* if $M(x) = 0$ for all $x \in X$, except maybe a finite number of them. *Cardinality* of multiset M is defined in such a way: $|M| = \sum_{x \in X} M(x)$. $\mathcal{M}(X)$ denotes the *set of all finite multisets* over X.

Set-theotetic notions are extended to finite multisets in the standard way. If $M, M' \in \mathcal{M}(X)$, we define $M + M'$ by $(M + M')(x) = M(x) + M'(x)$. We write $M \subseteq M'$, if $\forall x \in X\ M(x) \leq M'(x)$. When $M' \subseteq M$, we define $M - M'$ by $(M - M')(x) = M(x) - M'(x)$. Notation $M + x - y$ is used instead of $M + \{x\} - \{y\}$. We write symbol $\emptyset$ for empty multiset.

2.2 Labelled nets

Let $Act = \{a, b, \ldots\}$ be a set of *action names* or *labels*. A *labelled net* is a quadruple $N = \langle P_N, T_N, F_N, l_N \rangle$, where:

- $P_N = \{p, q, \ldots\}$ is a set of *places*;
- $T_N = \{u, v, \ldots\}$ is a set of *transitions*;
- $F_N : (P_N \times T_N) \cup (T_N \times P_N) \to \mathbf{N}$ is the *flow relation* with weights ($\mathbf{N}$ denotes a set of natural numbers);

- $l_N : T_N \to Act$ is a *labelling* of transitions with action names.

Let $N = \langle P_N, T_N, F_N, l_N \rangle$ be labelled net and $X \subseteq P_N \cup T_N$. A *restriction* of N on set X is a labelled net $N\lceil_X = \langle P_N \cap X, T_N \cap X, F_N\lceil_{(X\times X)}, l_N\lceil_{(T_N \cap X)} \rangle$. N' is a *subset* of N, denoted by $N' \subseteq N$, if $\exists X \subseteq P_N \cup T_N \ N' = N\lceil_X$.

Given a labelled net N and some transition $t \in T_N$, the *precondition* and *postcondition* t, notation respectively ${}^\bullet t$ and $t^\bullet$, are the multisets defined in such a way: $({}^\bullet t)(p) = F_N(p,t)$ and $(t^\bullet)(p) = F_N(t,p)$. Analogous definitions are introduced for places: $({}^\bullet p)(t) = F_N(t,p)$ and $(p^\bullet)(t) = F_N(p,t)$. A transition t is *unstable* if ${}^\bullet t = \emptyset$. A labelled net is *stable* if it has no unstable transitions. Further we will deal only with stable labelled nets. A labelled net N is *acyclic*, if there exists no sequence $x_1, \ldots, x_n,\ x_i \in P_N \cup T_N\ (1 \le i \le n)$ s.t. $F_N(x_{i-1}, x_i) > 0\ (1 \le i \le n)$ and $x_0 = x_n$. A labelled net N is *ordinary* if $\forall p \in P_N\ {}^\bullet p$ and $p^\bullet$ are proper sets (not multisets). A labelled net N is *finite* if $P_N \cup T_N$ is. Let ${}^\circ N = \{p \in P_N \mid {}^\bullet p = \emptyset\}$ is a set of *initial (input)* places of N and $N^\circ = \{p \in P_N \mid p^\bullet = \emptyset\}$ is a set of *final (output)* places of N.

Given labelled nets $N = \langle P_N, T_N, F_N, l_N \rangle$ and $N' = \langle P_{N'}, T_{N'}, F_{N'}, l_{N'} \rangle$.

A mapping $\beta : N \to N'$ is an *isomorphism* between N and N', notation $\beta : N \simeq N'$, if:

1. β is a bijection s.t. $\beta(P_N) = P_{N'}$ and $\beta(T_N) = T_{N'}$;
2. $\forall t \in T_N\ l_N(t) = l_{N'}(\beta(t))$;
3. $\forall t \in T_N\ {}^\bullet\beta(t) = \beta({}^\bullet t)$ and $\beta(t)^\bullet = \beta(t^\bullet)$.

Labelled nets N and N' are *isomorphic*, notation $N \simeq N'$, if there exists an isomorphism $\beta : N \simeq N'$.

Let $N = \langle P_N, T_N, F_N, l_N \rangle$ be acyclic ordinary labelled net and $x, y \in P_N \cup T_N$. Let us introduce the following notions.

- $x \prec_N y \ \Leftrightarrow\ x F_N^\star y$, where $F_N^\star$ is a transitive closure of F_N (*strict causal dependence* relation);
- $x \preceq_N y \ \Leftrightarrow\ (x \prec_N y) \vee (x = y)$ (*causal dependence* relation);
- $x \diamondsuit_N y \ \Leftrightarrow\ (x \prec_N y) \vee (y \prec_N x)$ (*linear ordering* relation);
- $x \#_N y \ \Leftrightarrow\ \exists t, u \in T_N\ (t \ne u,\ {}^\bullet t \cap {}^\bullet u \ne \emptyset,\ t \preceq_N x,\ u \preceq_N y)$ (*conflict* relation);
- $\downarrow_N x = \{y \in P_N \cup T_N \mid y \prec_N x\}$ (a set of *strict predecessors* of x).

Let $X \subseteq P_N \cup T_N$. A set X is *left-closed* in N, if $\forall x \in X\ \ \downarrow_N x \subseteq X$. A set X is *conflict-free* in N, if $\forall x, y \in X\ \neg(x \#_N y)$. A *configuration* of N is a finite left-closed conflict-free set $X \subseteq P_N \cup T_N$ s.t. ${}^\circ N \subseteq X$ and $\forall t \in T_N \cap X\ t^\bullet \subseteq X$. A *computation* of N is its subnet $N\lceil_X$, where X is a configuration of N.

2.3 Marked nets

Let N be a labelled net. A *marking* of N is a multiset $M \in \mathcal{M}(P_N)$. A *marked net (net)* is a tuple $N = \langle P_N, T_N, F_N, l_N, M_N \rangle$ where $\langle P_N, T_N, F_N, l_N \rangle$ is a labelled net and $M_N \in \mathcal{M}(P_N)$ is an *initial* marking. Let $M \in \mathcal{M}(P_N)$ be a marking of a net N. A transition $t \in T_N$ is *firable* in M, if ${}^\bullet t \subseteq M$. If t is firable in M, firing it yields a new marking $M' = M - {}^\bullet t + t^\bullet$, notation $M \xrightarrow{t} M'$. We write $M \to M'$, if $M \xrightarrow{t} M'$ for some t. A marking M' of a net N is *reachable from marking* M of the net, if:

1. $M' = M$ or
2. there exists a reachable from M marking M'' of a net N s.t. $M'' \to M'$.

A marking M of a net N is *reachable*, if it is reachable from M_N. $Mark(N, M)$ denotes a *set of all reachable from* M markings of a net N, and $Mark(N)$ — a *set of all reachable* markings of a net N.

2.4 Partially ordered sets

A *labelled partially ordered set (lposet)* is a triple $\rho = \langle X, \prec, l \rangle$, where:

- $X = \{x, y, \ldots\}$ is some set;
- $\prec \subseteq X \times X$ is a strict partial order (irreflexive transitive relation) over X;
- $l : X \to Act$ is a *labelling* function.

Let $x \in X$. Then $\downarrow x = \{y \in X \mid y \prec x\}$ is a set of *strict predecessors* of x.

Let $\rho = \langle X, \prec, l \rangle$ and $\rho' = \langle X', \prec', l' \rangle$ be lposets.

A mapping $\beta : X \to X'$ is a *label-preserving bijection* between ρ and ρ', notation $\beta : \rho \approx \rho'$, if:

1. β is a bijection;
2. $\forall x \in X\ l(x) = l'(\beta(x))$.

We write $\rho \approx \rho'$, if there exists a label-preserving bijection $\beta : \rho \approx \rho'$.

A mapping $\beta : X \to X'$ is a *homomorphism* between ρ and ρ', notation $\beta : \rho \sqsubseteq \rho'$, if:

1. $\beta : \rho \approx \rho'$;
2. $\forall x, y \in X\ x \prec y\ \Rightarrow\ \beta(x) \prec' \beta(y)$.

We write $\rho \sqsubseteq \rho'$, if there exists a homomorphism $\beta : \rho \sqsubseteq \rho'$.

A mapping $\beta : X \to X'$ is an *isomorphism* between ρ and ρ', notation $\beta : \rho \simeq \rho'$, if $\beta : \rho \sqsubseteq \rho'$ and $\beta^{-1} : \rho' \sqsubseteq \rho$. Lposets ρ and ρ' are *isomorphic*, notation $\rho \simeq \rho'$, if there exists an isomorphism $\beta : \rho \simeq \rho'$.

Partially ordered multiset (pomset) is an isomorphism class of lposets.

2.5 Event structures

A *labelled event structure (LES)* is a quadruple $\xi = \langle X, \prec, \#, l\rangle$, where:

- $X = \{x, y, \ldots\}$ is a set of *events*;
- $\prec \subseteq X \times X$ is a strict partial order, a *causal dependence* relation, which satisfies to the principle of *finite causes*: $\forall x \in X\ |\downarrow x| < \infty$;
- $\# \subseteq X \times X$ is an irreflexive symmetrical *conflict relation*, which satisfies to the principle of *conflict heredity*: $\forall x, y, z \in X\ x\#y \prec z\ \Rightarrow\ x\#z$;
- $l : X \to Act$ is a *labelling* function.

Let $Y \subseteq X$. A set Y is *left-closed* in ξ, if $\forall x \in Y\ \downarrow x \subseteq Y$. A set Y is *conflict-free* in ξ, if $\forall x, y \in Y\ \neg(x\#y)$. A *configuration* of LES ξ is a finite left-closed conflict-free set $Y \subseteq X$. A *computation* of LES ξ is lposet $\rho = \langle Y, \prec \cap (Y \times Y), l\lceil_Y\rangle$, where Y is a configuration of ξ.

Let $\xi = \langle X, \prec, \#, l\rangle$ and $\xi' = \langle X', \prec', \#', l'\rangle$ be LES.

A mapping $\beta : X \to X'$ is an *isomorphism* between ξ and ξ', notation $\beta : \xi \simeq \xi'$, if:

1. β is a bijection;
2. $\forall x \in X\ l(x) = l'(\beta(x))$;
3. $\forall x, y \in X\ x \prec y\ \Leftrightarrow\ \beta(x) \prec' \beta(y)$;
4. $\forall x, y \in X\ x\#y\ \Leftrightarrow\ \beta(x)\#'\beta(y)$.

LES ξ and ξ' are *isomorphic*, notation $\xi \simeq \xi'$, if there exists an isomorphism $\beta : \xi \simeq \xi'$.

A *prime event structure (PES)* is an isomorphism class of LES.

3 Equivalence notions

3.1 Equivalences based on C-processes

In this subsection we introduce definitions of equivalences based on C-processes, i.e. processes with causal nets [4].

3.1.1 C-processes

A *causal net* is acyclic ordinary labelled net $C = \langle P_C, T_C, F_C, l_C\rangle$, s.t:

1. $\forall r \in P_C\ |{}^\bullet r| \leq 1$ and $|r^\bullet| \leq 1$, i.e. places are unbranched;
2. $|\downarrow_C x| < \infty$, i.e. a set of causes is finite.

The fundamental property of causal nets is known [2, 3]: if C is a causal net, then there exists a transition sequence ${}^\circ C = L_0 \stackrel{v_1}{\to} \cdots \stackrel{v_n}{\to} L_n = C^\circ$ s.t. $L_i \subseteq P_C\ (0 \leq i \leq n)$, $P_C = \cup_{i=0}^n L_i$ and $T_C = \{v_1, \ldots, v_n\}$. Such a sequence is called a *full execution* of C.

Given a net N and a causal net C. A mapping $\varphi : P_C \cup T_C \to P_N \cup T_N$ is an *embedding* C into N, notation $\varphi : C \to N$, if:

1. $\varphi(P_C) \in \mathcal{M}(P_N)$ and $\varphi(T_C) \in \mathcal{M}(T_N)$, i.e. sorts are preserved;

2. $\forall v \in T_C\ l_C(v) = l_N(\varphi(v))$, i.e. labelling is preserved;

3. $\forall v \in T_C\ {}^\bullet\varphi(v) = \varphi({}^\bullet v)$ and $\varphi(v)^\bullet = \varphi(v^\bullet)$, i.e. flow relation is respected.

Since embeddings respect the flow relation, if ${}^\circ C \stackrel{v_1}{\to} \cdots \stackrel{v_n}{\to} C^\circ$ is a full execution of C, then $M = \varphi({}^\circ C) \stackrel{\varphi(v_1)}{\longrightarrow} \cdots \stackrel{\varphi(v_n)}{\longrightarrow} \varphi(C^\circ) = M'$ is a transition sequence in N, *corresponding* to this full execution, notation $M \stackrel{C,\varphi}{\to} M'$. Conversely, for any transition sequence $M \stackrel{t_1}{\to} \cdots \stackrel{t_n}{\to} M'$ of a net N there exists a causal net C and an embedding $\varphi : C \to N$ s.t. $M = \varphi({}^\circ C),\ M' = \varphi(C^\circ),\ t_i = \varphi(v_i)\ (0 \le i \le n)$ and ${}^\circ C \stackrel{v_1}{\to} \cdots \stackrel{v_n}{\to} C^\circ$ is a full execution of C.

A *firable in marking M C-process (process)* of a net N is a pair $\pi = (C, \varphi)$, where C is a causal net and $\varphi : C \to N$ is an embedding s.t. $M = \varphi({}^\circ C)$. A firable in M_N process is a *process* of N. We write $\Pi(N, M)$ for a *set of all firable* in marking M processes of a net N and $\Pi(N)$ for a *set of all processes* of a net N. Further we will deal only with *finite* processes, i.e. processes having finite causal nets. An *initial* process of a net N is $\pi_N = (C_N, \varphi_N) \in \Pi(N)$, s.t. $T_{C_N} = \emptyset$. If $\pi \in \Pi(N, M)$, then firing of this process transforms a marking M into $M' = M - \varphi({}^\circ C) + \varphi(C^\circ) = \varphi(C^\circ)$, notation $M \stackrel{\pi}{\to} M'$. So, processes and reachable markings of a net N are connected in the following way: $Mark(N, M) = \{\varphi(C^\circ) \mid \pi = (C, \varphi) \in \Pi(N, M)\}$.

Let $\pi_1 = (C_1, \varphi_1),\ \pi_2 = (C_2, \varphi_2) \in \Pi(N)$. A mapping $\beta : P_{C_1} \cup T_{C_1} \to P_{C_2} \cup T_{C_2}$ is an *isomorphism* between π_1 and π_2, notation $\beta : \pi_1 \simeq \pi_2$, if:

1. $\beta : C_1 \simeq C_2$;

2. $\forall x \in P_{C_1} \cup T_{C_1}\ \varphi_1(x) = \varphi_2(\beta(x))$.

Processes π_1 and π_2 are *isomorphic*, notation $\pi_1 \simeq \pi_2$, if there exists an isomorphism $\beta : \pi_1 \simeq \pi_2$.

Let $\pi = (C, \varphi),\ \tilde\pi = (\tilde C, \tilde\varphi) \in \Pi(N),\ \hat\pi = (\hat C, \hat\varphi) \in \Pi(N, \varphi(C^\circ)),\ C = \langle P_C, T_C, F_C, l_C\rangle,\ \tilde C = \langle P_{\tilde C}, T_{\tilde C}, F_{\tilde C}, l_{\tilde C}\rangle,\ \hat C = \langle P_{\hat C}, T_{\hat C}, F_{\hat C}, l_{\hat C}\rangle$.

We write $\pi \stackrel{\hat\pi}{\to} \tilde\pi$, if:

1. $P_C \cup P_{\hat C} = P_{\tilde C},\ T_C \cup T_{\hat C} = T_{\tilde C},\ F_C \cup F_{\hat C} = F_{\tilde C},\ l_C \cup l_{\hat C} = l_{\tilde C}$;

2. $\varphi \cup \hat\varphi = \tilde\varphi$.

In such a case $\tilde\pi$ is an *extension* of π *by process* $\hat\pi$, and $\hat\pi$ is an *extending* process for π. We write $\pi \to \tilde\pi$, if $\pi \stackrel{\hat\pi}{\to} \tilde\pi$ for some extending process $\hat\pi$.

A process π of a net N is *maximal*, if it can be extended by no process $\pi = (C, \varphi)$ s.t. $T_C \ne \emptyset$. Let us denote a set of *all maximal processes* of a net N by $\Pi_{max}(N)$.

$\tilde\pi$ is an extension of π *by one action*, if $\pi \stackrel{\hat\pi}{\to} \tilde\pi$ and $|T_{\hat C}| = 1$. In such a case we write $\pi \stackrel{v}{\to} \tilde\pi$ or $\pi \stackrel{a}{\to} \tilde\pi$, if $T_{\hat C} = \{v\}$ and $l_{\hat C}(v) = a$.

$\tilde\pi$ is an extension of π *by multiset of actions* or *step*, if $\pi \stackrel{\hat\pi}{\to} \tilde\pi$ and $\prec_{\hat C} = \emptyset$. In such a case we write $\pi \stackrel{V}{\to} \tilde\pi$ or $\pi \stackrel{A}{\to} \tilde\pi$, if $T_{\hat C} = V$ and $l_{\hat C}(T_{\hat C}) = A,\ A \in \mathcal{M}(Act)$.

Let us note that on the basis of any causal net C one can define lposet $\rho_C = \langle T_C, \prec_N \cap (T_C \times T_C), l_C\rangle$.

3.1.2 Trace equivalences

An *interleaving trace* of a net N is a sequence $a_1 \cdots a_n \in Act^*$ s.t. $\pi_N \stackrel{a_1}{\to} \pi_1 \stackrel{a_2}{\to} \ldots \stackrel{a_n}{\to} \pi_n$, where $\pi_i \in \Pi(N)\ (1 \le i \le n)$ and π_N is an initial process of N. $SeqTraces(N)$ denotes a *set of all interleaving traces* of N. Nets N and N' are *interleaving trace equivalent*, notation $N \equiv_i N'$, if $SeqTraces(N) = SeqTraces(N')$.

A *step trace* of a net N is a sequence $A_1 \cdots A_n \in (\mathcal{M}(Act))^*$ s.t. $\pi_N \stackrel{A_1}{\to} \pi_1 \stackrel{A_2}{\to} \ldots \stackrel{A_n}{\to} \pi_n$, where $\pi_i \in \Pi(N)\ (0 \le i \le n)$, and π_N is an initial process of N. $StepTraces(N)$ denotes a *set of all step traces* of N. Nets N and N' are *step trace equivalent*, notation $N \equiv_s N'$, if $StepTraces(N) = StepTraces(N')$.

A *pomset trace* of a net N is a pomset ρ, an isomorphism class of lposet ρ_C for $\pi = (C, f) \in \Pi(N)$. We write $\rho \sqsubseteq \rho'$, if $\rho_C \sqsubseteq \rho_{C'}$ for $\rho_C \in \rho$ and $\rho_{C'} \in \rho'$. In such a case we say that pomset ρ is *less sequential* or *more parallel* than ρ'. $Pomsets(N)$ denotes a *set of all pomset traces* of N. Nets N and N' are *partial word trace equivalent*, notation $N \equiv_{pw} N'$, if $Pomsets(N) \sqsubseteq Pomsets(N')$ and $Pomsets(N') \sqsubseteq Pomsets(N)$, i.e. for any $\rho' \in Pomsets(N')$ there exists $\rho \in Pomsets(N)$ s.t. $\rho \sqsubseteq \rho'$ and vice versa. Nets N and N' are *pomset trace equivalent*, notation $N \equiv_{pom} N'$, if $Pomsets(N) = Pomsets(N')$.

A *process trace* of a net N is an isomorphism class of causal net C for $\pi = (C, f) \in \Pi(N)$. $ProcessNets(N)$ denotes a *set of all process traces* of N. Nets N and N' are *process trace equivalent*, notation $N \equiv_{pr} N'$, if $ProcessNets(N) = ProcessNets(N')$.

3.1.3 (Usual) bisimilation equivalences

A notation $\mathcal{R} : N \underline{\leftrightarrow}_{\star} N'$ means that $\mathcal{R}$ is a bisimulation of type $\star$ ($\star$-bisimulation) between nets N and N'. Nets N and N' are called $\star$-bisimulation equivalent, notation $N \underline{\leftrightarrow}_{\star} N'$, if $\mathcal{R} : N \underline{\leftrightarrow}_{\star} N'$ for some $\star$-bisimulation $\mathcal{R}$.

Let $\mathcal{R} \subseteq \Pi(N) \times \Pi(N')$. In the following definition $\hat{\pi} = (\hat{C}, \hat{f}),\ \hat{\pi}' = (\hat{C}', \hat{f}')$.

$\mathcal{R}$ is a *$\star$-bisimulation* between N and N', $\star \in \{$*interleaving, step, partial word, pomset, process*$\}$, notation $\mathcal{R} : N \underline{\leftrightarrow}_{\star} N',\ \star \in \{i, s, pw, pom, pr\}$, if:

1. $(\pi_N, \pi_{N'}) \in \mathcal{R}$;
2. $(\pi, \pi') \in \mathcal{R},\ \pi \xrightarrow{\hat{\pi}} \tilde{\pi}$,
   (a) $|T_{\hat{C}}| = 1$, if $\star = i$;
   (b) $\prec_{\hat{C}} = \emptyset$, if $\star = s$;

   then $\exists \tilde{\pi}' :\ \pi' \xrightarrow{\hat{\pi}'} \tilde{\pi}',\ (\tilde{\pi}, \tilde{\pi}') \in \mathcal{R}$ and
   (a) $\rho_{\hat{C}'} \sqsubseteq \rho_{\hat{C}}$, if $\star = pw$;
   (b) $\rho_{\hat{C}} \simeq \rho_{\hat{C}'}$, if $\star \in \{i, s, pom\}$;
   (c) $\hat{C} \simeq \hat{C}'$, if $\star = pr$;
3. As previous item but the roles of N and N' are reversed.

3.1.4 ST-processes

ST-processes are introduced for the representation of states of nets supposing that transitions of the nets may have some internal structure or their occurrences have a duration.

A *firable in marking M ST-process* of a net N is a pair (π_E, π_P) s.t. $\pi_E, \pi_P \in \Pi(N, M)$, $\pi_P \xrightarrow{\pi_W} \pi_E$ and $\forall v, w \in T_{C_E}\ v \prec_{C_E} w \ \Rightarrow\ v \in T_{C_P}$ In such a case π_E is a process which has started, i.e. all actions of π_E have started. A process π_P corresponds to the finished part of π_E, and π_W corresponds to the still working part. Clearly, $\prec_{C_W} = \emptyset$. A firable in M_N ST-process is an *ST-process* of a net N. $ST - \Pi(N, M)$ denotes a *set of all firable in M* ST-processes of N, and $ST - \Pi(N)$ denotes a *set of all ST-processes* of N. (π_N, π_N) is an *initial ST-process* of N. Let $(\pi_E, \pi_P),\ (\tilde{\pi}_E, \tilde{\pi}_P) \in ST - \Pi(N)$. We write $(\pi_E, \pi_P) \rightarrow (\tilde{\pi}_E, \tilde{\pi}_P)$, if $\pi_E \rightarrow \tilde{\pi}_E$ and $\pi_P \rightarrow \tilde{\pi}_P$.

3.1.5 ST-bisimulation equivalences

Let $\mathcal{R} \subseteq ST - \Pi(N) \times ST - \Pi(N') \times \mathcal{B}$, where $\mathcal{B} = \{\beta \mid \beta : T_C \rightarrow T_{C'},\ \pi = (C, f) \in \Pi(N),\ \pi' = (C', f') \in \Pi(N')\}$. In the following definition $\pi_E = (C_E, f_E),\ \pi_P = (C_P, f_P),\ \pi'_E = (C'_E, f'_E),\ \pi'_P = (C'_P, f'_P),\ \pi = (C, f),\ \pi' = (C', f')$.

$\mathcal{R}$ is a *$\star$-ST-bisimulation* between N and N' $\star \in \{$*interleaving, partial word, pomset, process*$\}$, notation $\mathcal{R} : N \underline{\leftrightarrow}_{\star ST} N',\ \star \in \{i, pw, pom, pr\}$, if:

1. $((\pi_N, \pi_N), (\pi_{N'}, \pi_{N'}), \emptyset) \in \mathcal{R}$;
2. $((\pi_E, \pi_P), (\pi'_E, \pi'_P), \beta) \in \mathcal{R} \ \Rightarrow\ \beta : \rho_{C_E} \approx \rho_{C'_E}$ and $\beta(T_{C_P}) = T_{C'_P}$;
3. $((\pi_E, \pi_P), (\pi'_E, \pi'_P), \beta) \in \mathcal{R},\ (\pi_E, \pi_P) \rightarrow (\tilde{\pi}_E, \tilde{\pi}_P) \ \Rightarrow\ \exists \tilde{\beta},\ (\tilde{\pi}'_E, \tilde{\pi}'_P) :\ (\pi'_E, \pi'_P) \rightarrow (\tilde{\pi}'_E, \tilde{\pi}'_P),\ \tilde{\beta}\lceil_{T_{C_E}} = \beta,$
   $((\tilde{\pi}_E, \tilde{\pi}_P), (\tilde{\pi}'_E, \tilde{\pi}'_P), \tilde{\beta}) \in \mathcal{R}$, and if $\pi_P \xrightarrow{\pi} \tilde{\pi}_E,\ \pi'_P \xrightarrow{\pi'} \tilde{\pi}'_E$ then:
   (a) $(\tilde{\beta}\lceil_{T_C})^{-1} : \rho_{C'} \sqsubseteq \rho_C$, if $\star = pw$;
   (b) $\tilde{\beta}\lceil_{T_C} : \rho_C \simeq \rho_{C'}$, if $\star \in \{pom, pr\}$;
   (c) $C \simeq C'$, if $\star = pr$;
4. As previous item but the roles of N and N' are reversed.

3.1.6 History preserving bisimulation equivalences

Let $\mathcal{R} \subseteq \Pi(N) \times \Pi(N') \times \mathcal{B}$, where $\mathcal{B} = \{\beta \mid \beta : T_C \to T_{C'},\ \pi = (C, f) \in \Pi(N),\ \pi' = (C', f') \in \Pi(N')\}$. In the following definition $\pi = (C, f), \tilde{\pi} = (\tilde{C}, \tilde{f}),\ \pi' = (C', f'), \tilde{\pi}' = (\tilde{C}', \tilde{f}')$.

$\mathcal{R}$ is a $\star$-*history preserving bisimulation* between N and $N', \star \in$ {*partial word, pomset, process*}, notation $N \underline{\leftrightarrow}_{\star h} N'$, $\star \in \{pw, pom, pr\}$, if:

1. $(\pi_N, \pi_{N'}, \emptyset) \in \mathcal{R}$;
2. $(\pi, \pi', \beta) \in \mathcal{R} \ \Rightarrow\ \beta : \rho_C \approx \rho_{C'}$;
3. $(\pi, \pi', \beta) \in \mathcal{R},\ \pi \to \tilde{\pi} \ \Rightarrow\ \exists \tilde{\beta},\ \tilde{\pi}' : \pi' \to \tilde{\pi}',\ \tilde{\beta}\lceil_{T_C} = \beta,\ (\tilde{\pi}, \tilde{\pi}', \tilde{\beta}) \in \mathcal{R}$ and
   (a) $\tilde{\beta}^{-1} : \rho_{\tilde{C}'} \sqsubseteq \rho_{\tilde{C}}$, if $\star = pw$;
   (b) $\tilde{\beta} : \rho_{\tilde{C}} \simeq \rho_{\tilde{C}'}$, if $\star \in \{pom, pr\}$;
   (c) $\tilde{C} \simeq \tilde{C}'$, if $\star = pr$;
4. As previous item but the roles of N and N' are reversed.

3.2 Equivalences based on O-processes

In this subsection we introduce definitions of equivalences based on O-processes, i.e. processes with occurrence nets (branching processes, in terminology of [7]).

3.2.1 O-processes

An *occurrence net* is an acyclic ordinary labelled net $O = \langle P_O, T_O, F_O, l_O \rangle$, s.t.:

1. $\forall r \in P_O\ |{}^\bullet r| \leq 1$, i.e. there is no forward conflict;
2. $\forall x \in P_O \cup T_O\ \neg(x \#_O x)$, i.e. conflict relation is irreflexive;
3. $\forall x \in P_O \cup T_O \mid \downarrow_O x| < \infty$, i.e. set of causes is finite.

Let us note that computations of occurrence net are its initial causal subnets.

Let $O = \langle P_O, T_O, F_O, l_O \rangle$ be occurrence net and $N = \langle P_N, T_N, F_N, l_N, M_N \rangle$ be some net. A mapping $\psi : P_O \cup T_O \to P_N \cup T_N$ is an *embedding* O into N, notation $\psi : O \to N$, if:

1. $\psi(P_O) \in \mathcal{M}(P_N)$ and $\psi(T_O) \in \mathcal{M}(T_N)$. i.e. sorts are preserved;
2. $\forall v \in T_O\ l_O(v) = l_N(\psi(v))$, i.e. labelling is preserved;
3. $\forall v \in T_O\ {}^\bullet\psi(v) = \psi({}^\bullet v)$ and $\psi(v)^\bullet = \psi(v^\bullet)$, i.e. flow relation is respected;
4. $\forall v, w \in T_O\ ({}^\bullet v = {}^\bullet w) \wedge (\psi(v) = \psi(w)) \Rightarrow\ v = w$, i.e. there are no "superfluous" conflicts.

A *firable in marking M O-process* of a net N is a pair $\varpi = (O, \psi)$, where O is an occurrence net and $\psi : O \to N$ is an embedding s.t. $M = \psi(^\circ O)$. Let us note that marking M may be not reachable in general case. A firable in M_N O-process is *O-process* of a net N. We write $\wp(N, M)$ for a set of *all firable in marking M O-processes* of a net N and $\wp(N)$ for a set of *all O-processes* of a net N. Further we will deal only with *finite* O-processes, i.e. O-processes having finite occurrence nets. An *initial* O-process of a net N coincides with its initial C-process, i.e. $\varpi_N = \pi_N$.

Let $\varpi = (O_1, \psi_1),\ \varpi_2 = (O_2, \psi_2) \in \wp(N)$. A mapping $\beta : P_{O_1} \cup T_{O_1} \to P_{O_2} \cup T_{O_2}$ is an *isomorphism* between ϖ_1 and ϖ_2, notation $\beta : \varpi_1 \simeq \varpi_2$, if:

1. $\beta : O_1 \simeq O_2$;
2. $\forall x \in P_{O_1} \cup T_{O_1}\ \psi_1(x) = \psi_2(\beta(x))$.

O-processes ϖ_1 and ϖ_2 are *isomorphic*, notation $\varpi_1 \simeq \varpi_2$, if there exists an isomorphism $\beta : \varpi_1 \simeq \varpi_2$.

Let $\varpi = (O, \psi),\ \tilde{\varpi} = (\tilde{O}, \tilde{\psi}) \in \wp(N),\ \hat{\varpi} = (\hat{O}, \hat{\psi}) \in \wp(N, \psi(O^\circ)),\ O = \langle P_O, T_O, F_O, l_O \rangle,\ \tilde{O} = \langle P_{\tilde{O}}, T_{\tilde{O}}, F_{\tilde{O}}, l_{\tilde{O}} \rangle$, $\hat{O} = \langle P_{\hat{O}}, T_{\hat{O}}, F_{\hat{O}}, l_{\hat{O}} \rangle$.

We write $\varpi \overset{\hat{\varpi}}{\to} \tilde{\varpi}$, if:

1. $P_O \cup P_{\hat{O}} = P_{\tilde{O}};\ T_O \cup T_{\hat{O}} = T_{\tilde{O}};\ F_O \cup F_{\hat{O}} = F_{\tilde{O}};\ l_O \cup l_{\hat{O}} = l_{\tilde{O}}$;

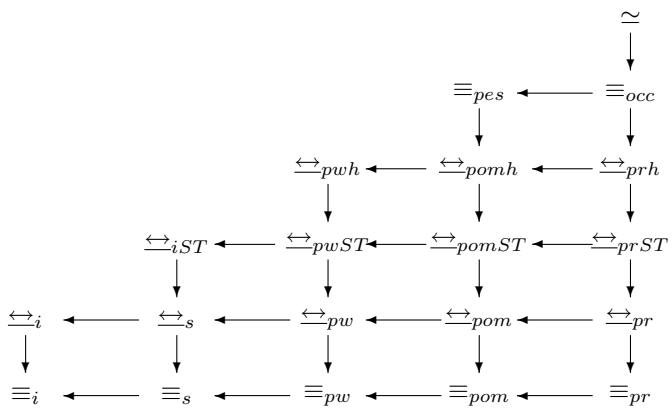

Figure 1: Correlation of equivalence notions on whole class of Petri nets

2. $\psi \cup \hat{\psi} = \tilde{\psi}$.

In such a case O-process $\tilde{\varpi}$ is an *extension* of ϖ *by O-process* $\hat{\varpi}$, and $\hat{\varpi}$ is an *extending* O-process for ϖ. We write $\varpi \rightarrow \tilde{\varpi}$, if $\varpi \stackrel{\hat{\varpi}}{\rightarrow} \tilde{\varpi}$ for some extending O-process $\hat{\varpi}$.

An O-process ϖ of a net N is *maximal*, if it can be extended by no O-process $\hat{\varpi} = (\hat{O}, \hat{\psi})$ s.t. $T_{\hat{O}} \neq \emptyset$. A set of *all maximal O-processes* of a net N, notation $\wp_{max}(N)$, consists ot the unique (up to isomorphism) O-process $\varpi_{max} = (O_{max}, \psi_{max})$. In such a case an isomorphism class of occurrence net O_{max} is an *unfolding* of a net N, notation $\mathcal{U}(N)$.

Let us note that on the basis of any occurrence net O one can define LES $\xi_O = \langle T_O, \prec_O \cap (T_O \times T_O), \#_O \cap (T_O \times T_O), l_O \rangle$. Then on the basis of unfolding $\mathcal{U}(N)$ of a net N one can define PES $\mathcal{E}(N) = \xi_{\mathcal{U}(N)}$ which is an isomorphism class of LES ξ_O for $O \in \mathcal{U}(N)$.

3.2.2 Conflict respecting equivalences

A *PES-trace* of a net N is PES ξ, an isomorphism class of LES ξ_O for $\varpi = (O, \psi) \in \wp(N)$. $PEStructs(N)$ denotes a set of *all PES-traces* of a net N. Nets N and N' are *PES-equivalent*, notation $N \equiv_{pes} N'$, if $PEStructs(N) = PEStructs(N')$. Let us note that we can change this requirement by $\mathcal{E}(N) = \mathcal{E}(N')$ due to the uniqueness of maximal O-process.

An *occurrence trace* of a net N is an isomorphism class of occurrence net O for $\varpi = (O, \psi) \in \wp(N)$. $OccNets(N)$ denotes a set of *all occurrence traces* of a net N. Nets N and N' are *occurrence equivalent* notation $N \equiv_{occ} N'$, if $OccNets(N) = OccNets(N')$. Let us note that we can change this requirement by $\mathcal{U}(N) = \mathcal{U}(N')$ due to the uniqueness of maximal O-process.

4 Comparing equivalence notions on whole class of Petri nets

Theorem 1 *Let* $\leftrightarrow \in \{\equiv, \underline{\leftrightarrow}, \simeq\}$ *and* $\star, \star\star \in \{i, s, pw, pom, pr, iST, pwST, pomST, prST, pwh, pomh, prh, pes, occ\}$. *For nets* N *and* N' $N \leftrightarrow_\star N' \Rightarrow N \leftrightarrow_{\star\star} N'$ *iff there exists a directed path from* $\leftrightarrow_\star$ *to* $\leftrightarrow_{\star\star}$ *in the graph in Figure 1.*

Proof. $\Leftarrow$ See [18] and the following substantiations.

- Implication $\equiv_{occ} \rightarrow \equiv_{pes}$ is valid since PES of isomorphic occurrence nets coincide.
- Implication $\equiv_{pes} \rightarrow \underline{\leftrightarrow}_{pomh}$ is proved as follows. Let $\varpi = (O, \psi) \in \wp_{max}(N)$, $\varpi' = (O', \psi') \in \wp_{max}(N')$, $\gamma : \xi_O \simeq \xi_{O'}$. We have $\mathcal{R} : N \underline{\leftrightarrow}_{pomh} N'$, where relation $\mathcal{R}$ is defined in the following way. Let $\pi = (C, \varphi)$, $\pi' = (C', \varphi')$, then $(\pi, \pi', \beta) \in \mathcal{R} \Leftrightarrow$ [C is a computation of O, $\varphi = \psi\lceil_{(P_C \cup T_C)}$, C' is a computation of O', $\varphi' = \psi'\lceil_{(P_{C'} \cup T_{C'})}$ s.t. $\gamma\lceil_{T_C} : \rho_C \simeq \rho_{C'}$, $\beta = \gamma\lceil_{T_C}$].
- Implication $\equiv_{occ} \rightarrow \underline{\leftrightarrow}_{prh}$ is proved as follows. Let $\varpi = (O, \psi) \in \wp_{max}(N)$, $\varpi' = (O', \psi') \in \wp_{max}(N')$, $\gamma : O \simeq O'$. We have $\mathcal{R} : N \underline{\leftrightarrow}_{prh} N'$, where relation $\mathcal{R}$ is defined in the following way. Let $\pi = (C, \varphi)$, $\pi' = (C', \varphi')$, then $(\pi, \pi', \beta) \in \mathcal{R} \Leftrightarrow$ [C is a computation of O, $\varphi = \psi\lceil_{(P_C \cup T_C)}$, C' is a computation of O', $\varphi' = \psi'\lceil_{(P_{C'} \cup T_{C'})}$ s.t. $\gamma\lceil_{(P_C \cup T_C)} : C \simeq C'$, $\beta = \gamma\lceil_{T_C}$].

- Implication $\simeq \rightarrow \equiv_{occ}$ is valid since unfoldings of isomorphic nets coincide.

$\Rightarrow$ Impossibility to draw any additional arrow in Figure 1 is proved by the following examples on nets.

- In Figure 2(a) $N \underline{\leftrightarrow}_i N'$, but $N \not\equiv_s N'$, since only in N actions a and b can happen concurrently.
- In Figure 2(e) $N \underline{\leftrightarrow}_{iST} N'$, but $N \not\equiv_{pw} N'$, since the pomset corresponds to the net N s.t. even less sequential pomset is not in N'.
- In Figure 2(c) $N \underline{\leftrightarrow}_{pwh} N'$, but $N \not\equiv_{pom} N'$, since only in net N action b can depend on action a.
- In Figure 2(d) $N \equiv_{pes} N'$, but $N \not\equiv_{pr} N'$, since N is causal net which is not isomorphic to N' (because of additional output place).
- In Figure 2(b) $N \equiv_{pr} N'$, but $N \underline{\not\leftrightarrow}_i N'$, since only in net N action a can happen so that action b can not happen afterwards.
- In Figure 3(a) $N \underline{\leftrightarrow}_{pr} N'$, but $N \underline{\not\leftrightarrow}_{iST} N'$, since only in net N' action a can start so that no action b can begin working until a finishes.
- In Figure 3(b) $N \underline{\leftrightarrow}_{prST} N'$, but $N \underline{\not\leftrightarrow}_{pwh} N'$, since only in net N' after action a action b can happen so that action c must depend on a.
- In Figure 3(c) $N \underline{\leftrightarrow}_{prh} N'$, but $N \not\equiv_{pes} N'$, since only net N' is corresponded by PES with two conflict actions a.
- In Figure 3(d) $N \equiv_{occ} N'$, but $N \not\simeq N'$, since unfirable transitions of nets N and N' are labelled by different actions (a and b). □

5 Comparing equivalence notions on subclasses of Petri nets

5.1 Sequential nets

A *sequential net* is a net $N = \langle P_N, T_N, F_N, l_N, M_N \rangle$ s.t. $\forall \pi = (C, \varphi) \in \Pi(N)\ \forall v, w \in T_C\ (v \diamondsuit_C w)$ (i.e. $\prec_C$ is a total ordering on transitions of causal net C).

Proposition 1 *For sequential nets N and N':*

1. $N \equiv_i N' \ \Leftrightarrow\ N \equiv_{pom} N'$;
2. $N \underline{\leftrightarrow}_i N' \ \Leftrightarrow\ N \underline{\leftrightarrow}_{pomh} N'$.

Proof.

1. $\Leftarrow$ By Theorem 1.

   $\Rightarrow$ Let $N \equiv_i N'$, then $SeqTraces(N) = SeqTraces(N')$. To prove $N \equiv_{pom} N'$, it is sufficient to establish the equality $Pomsets(N) = Pomsets(N')$. It follows immediately, since $Pomsets(N)$ and $Pomsets(N')$ are totally ordered multisets (chains), and there is on-to-one correspondence between $SeqTraces(N)$ and $Pomsets(N)$ ($SeqTraces(N')$ and $Pomsets(N')$ respectively).
2. See [5]. □

Theorem 2 *Let $\leftrightarrow \in \{\equiv, \underline{\leftrightarrow}, \simeq\}$ and $\star, \star\star \in \{i, pr, prST, prh, pes, occ\}$. For sequential nets N and N' $N \leftrightarrow_\star N' \ \Rightarrow\ N \leftrightarrow_{\star\star} N'$ iff there exists a directed path from $\leftrightarrow_\star$ to $\leftrightarrow_{\star\star}$ in the graph in Figure 4.*

Proof. $\Leftarrow$ By Theorem 1.

$\Rightarrow$ Impossibility to draw any additional arrow in Figure 4 is proved by the following examples on sequential nets.

- In Figure 2(d) $N \equiv_{pes} N'$, but $N \not\equiv_{pr} N'$.
- In Figure 2(b) $N \equiv_{pr} N'$, but $N \underline{\not\leftrightarrow}_i N'$.
- In Figure 5(a) $N \underline{\leftrightarrow}_{pr} N'$, but $N \underline{\not\leftrightarrow}_{prST} N'$, since only in net N' process with action a can start so that it can be extended by process with action b in the only way (i.e. so that extended process be unique).

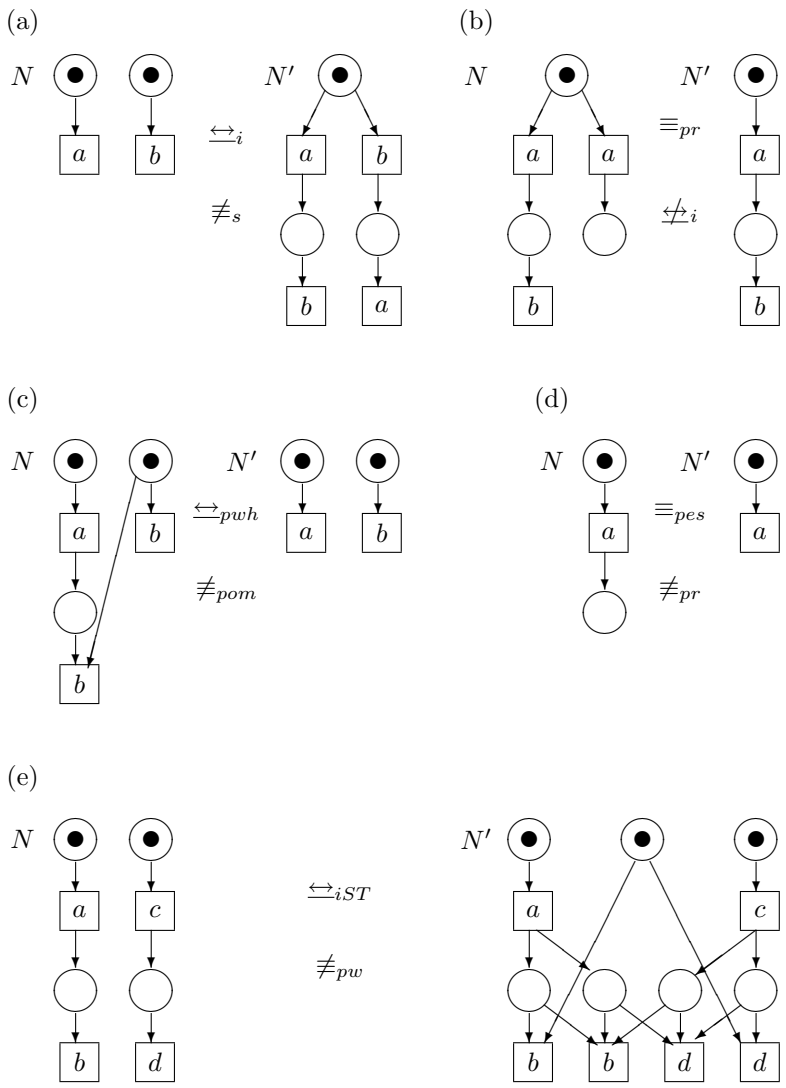

Figure 2: Examples on Petri nets

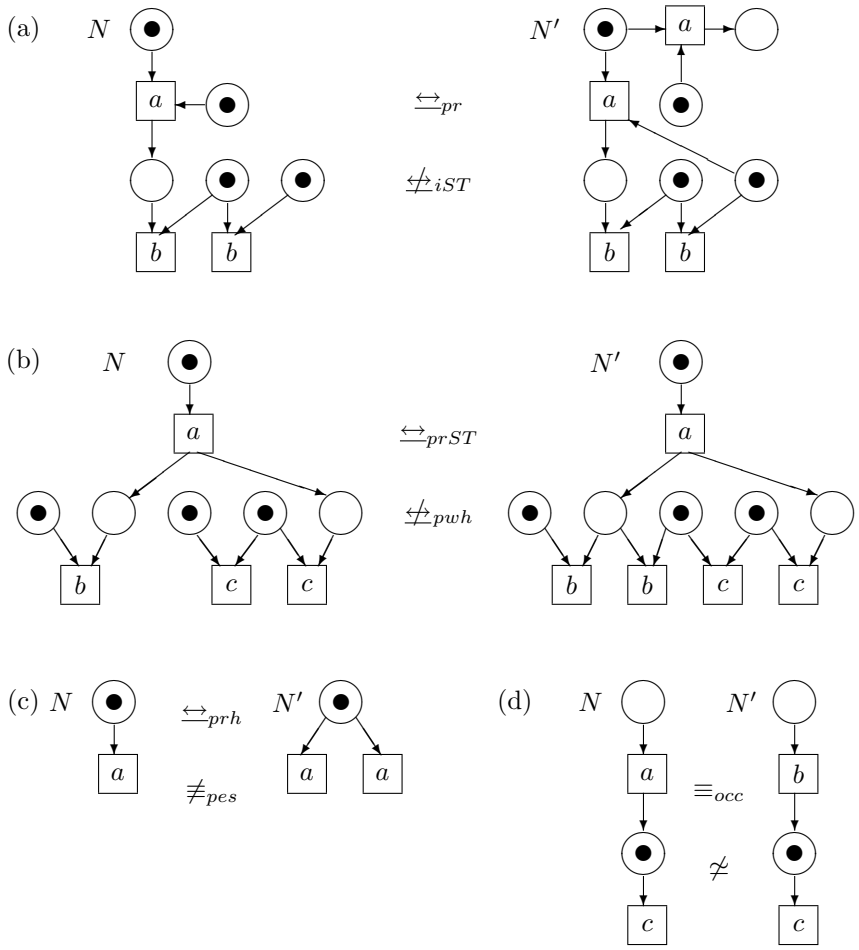

Figure 3: Examples on Petri nets (continued)

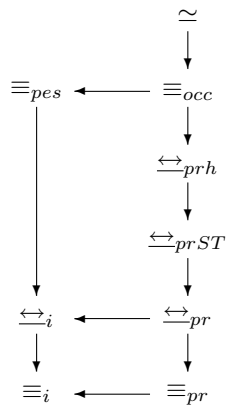

Figure 4: Correlation of equivalence notions on sequential nets

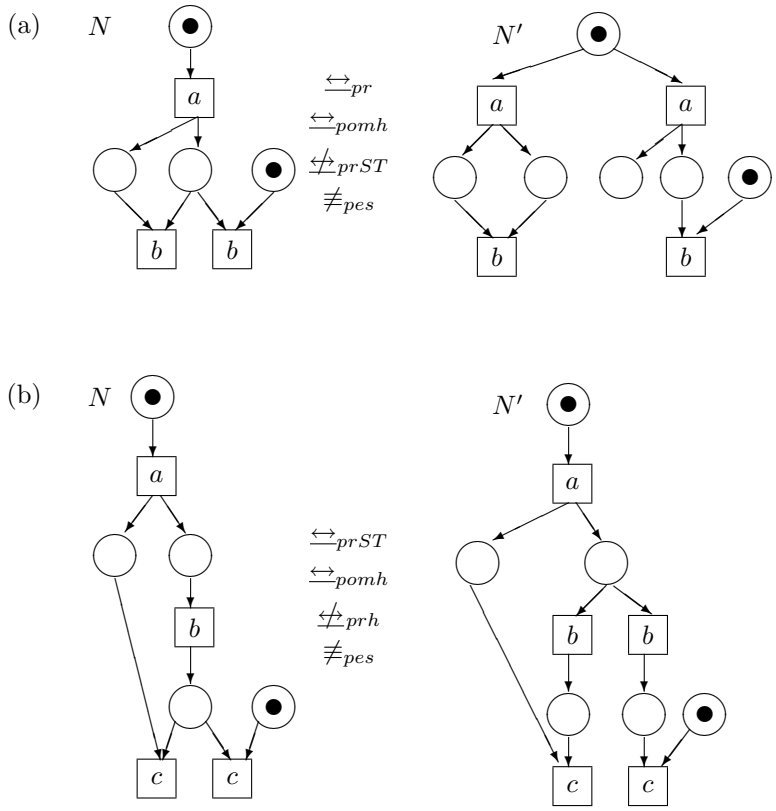

Figure 5: Examples on sequential nets

- In Figure 5(b) $N \underline{\leftrightarrow}_{prST} N'$, but $N \not\underline{\leftrightarrow}_{prh} N'$, since only in net N' there is process with actions a and b s.t. it can be extended by process with action c in the only way. (i.e. so that connection of causal net with action c and a-containing subnet of causal net with actions a and b be unique).
- In Figure 3(c) $N \underline{\leftrightarrow}_{prh} N'$, but $N \not\equiv_{pes} N'$.
- In Figure 3(d) $N \equiv_{occ} N'$, but $N \not\simeq N'$. □

5.2 Strictly labelled nets

A *strictly labelled net* is a net $N = \langle P_N, T_N, F_N, l_N, M_N \rangle$ s.t. $\forall t, u \in T_N \; t \neq u \;\Rightarrow\; l_N(t) \neq l_N(u)$ (i.e. its labelling function is injective).

Proposition 2 *For strictly labelled nets N and N':*

1. $N \equiv_\star N' \Leftrightarrow \; N \underline{\leftrightarrow}_\star N', \; \star \in \{i, pw, pom, pr\}$;
2. $N \equiv_s N' \Leftrightarrow \; N \underline{\leftrightarrow}_{iST} N'$.

Proof. Omitted. □

The following example demonstrates which equivalence notions do not merge on strictly labelled nets.

Example 1
- *In Figure 6(a) $N \underline{\leftrightarrow}_i N'$, but $N \not\equiv_s N'$, since only in net N actions a and b can happen concurrently.*
- *In Figure 6(b) $N \underline{\leftrightarrow}_{pwh} N'$, but $N \not\equiv_{pom} N'$, since only in net N' action b can depend on action a.*
- *In Figure 2(d) $N \equiv_{pes} N'$, but $N \not\equiv_{pr} N'$.*
- *In Figure 6(c) $N \underline{\leftrightarrow}_{pomST} N'$, but $N \not\underline{\leftrightarrow}_{pwh} N'$, since only in net N' after action a action b can happen so that action c must depend on a.*

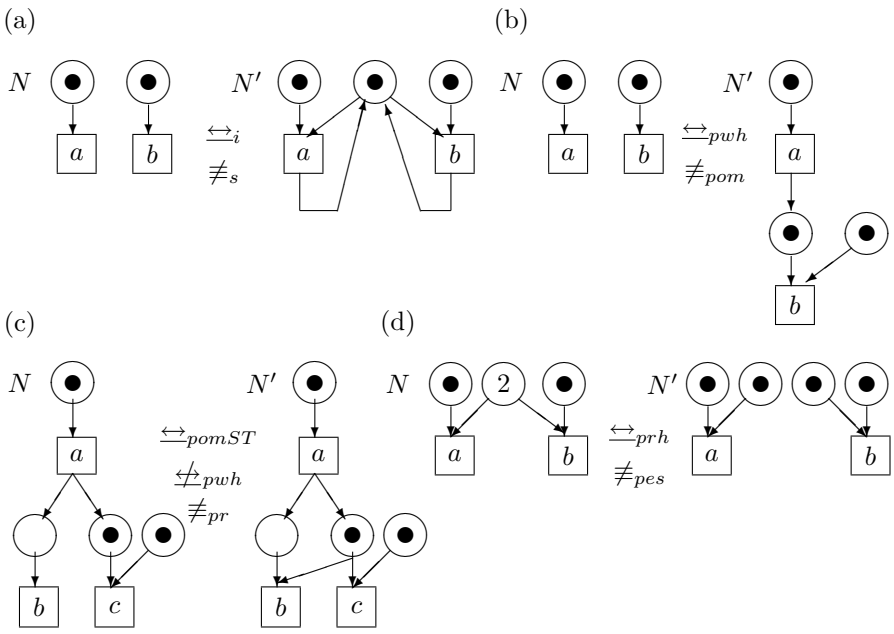

Figure 6: Examples on strictly labelled nets

- *In Figure 6(d)* $N \underline{\leftrightarrow}_{prh} N'$*, but* $N \not\equiv_{pes} N'$*, since only in unfolding of a net* N' *transitions with labels* a *and* b *have common input place. PES with conflict actions* a *and* b *corresponds to this unfolding.*
- *In Figure 3(d)* $N \equiv_{occ} N'$*, but* $N \not\simeq N'$*.*

6 Preservation of equivalence notions by refinements

An *empty in/out net* is a net $D = \langle P_D, T_D, F_D, l_D, M_D \rangle$ s.t.:

1. $\exists p_{in},\ p_{out} \in P_D$ s.t. $p_{in} \neq p_{out}$ and $^\circ D = \{p_{in}\},\ D^\circ = \{p_{out}\}$, i.e. net D has unique input and unique output place.
2. $M_D = \{p_{in}\}$ and $\forall M \in Mark(D)\ (p_{out} \in M \ \Rightarrow\ M = \{p_{out}\})$, i.e. at the beginning there is unique token in p_{in}, and at the end there is unique token in p_{out};
3. $p_{in}^\bullet$ and $^\bullet p_{out}$ are proper sets (not multisets), i.e. p_{in} (respectively p_{out}) represents a set of all tokens consumed (respectively produced) for any refined transition.

Let $N = \langle P_N, T_N, F_N, l_N, M_N \rangle$ be some net, $a \in l_N(T_N)$ and $D = \langle P_D, T_D, F_D, l_D, M_D \rangle$ be empty in/out system. An *empty in/out refinement*, notation $ref(N, a, D)$, is (up to isomorphism) a net $\overline{N} = \langle P_{\overline{N}}, T_{\overline{N}}, F_{\overline{N}}, l_{\overline{N}}, M_{\overline{N}} \rangle$, s.t.:

1. $P_{\overline{N}} = P_N \cup \{\langle p, u \rangle \mid p \in P_D \setminus \{p_{in}, p_{out}\},\ u \in l_N^{-1}(a)\}$;
2. $T_{\overline{N}} = (T_N \setminus l_N^{-1}(a)) \cup \{\langle t, u \rangle \mid t \in T_D,\ u \in l_N^{-1}(a)\}$;
3. $F_{\overline{N}}(\bar{x}, \bar{y}) = \begin{cases} F_N(\bar{x}, \bar{y}), & \bar{x}, \bar{y} \in P_N \cup (T_N \setminus l_N^{-1}(a)); \\ F_D(x, y), & \bar{x} = \langle x, u \rangle,\ \bar{y} = \langle y, u \rangle,\ u \in l_N^{-1}(a); \\ F_N(\bar{x}, u), & \bar{y} = \langle y, u \rangle,\ x \in {}^\bullet u,\ u \in l_N^{-1}(a),\ y \in p_{in}^\bullet; \\ F_N(u, \bar{y}), & \bar{x} = \langle x, u \rangle,\ y \in {}^\bullet u,\ u \in l_N^{-1}(a),\ x \in {}^\bullet p_{out}; \\ 0, & \text{otherwise;} \end{cases}$
4. $l_{\overline{N}}(\bar{u}) = \begin{cases} l_N(\bar{u}), & \bar{u} \in T_N \setminus l_N^{-1}(a); \\ l_D(t), & \bar{u} = \langle t, u \rangle,\ t \in T_D,\ u \in l_N^{-1}(a); \end{cases}$
5. $M_{\overline{N}}(p) = \begin{cases} M_N(p), & p \in P_N; \\ 0, & \text{otherwise.} \end{cases}$

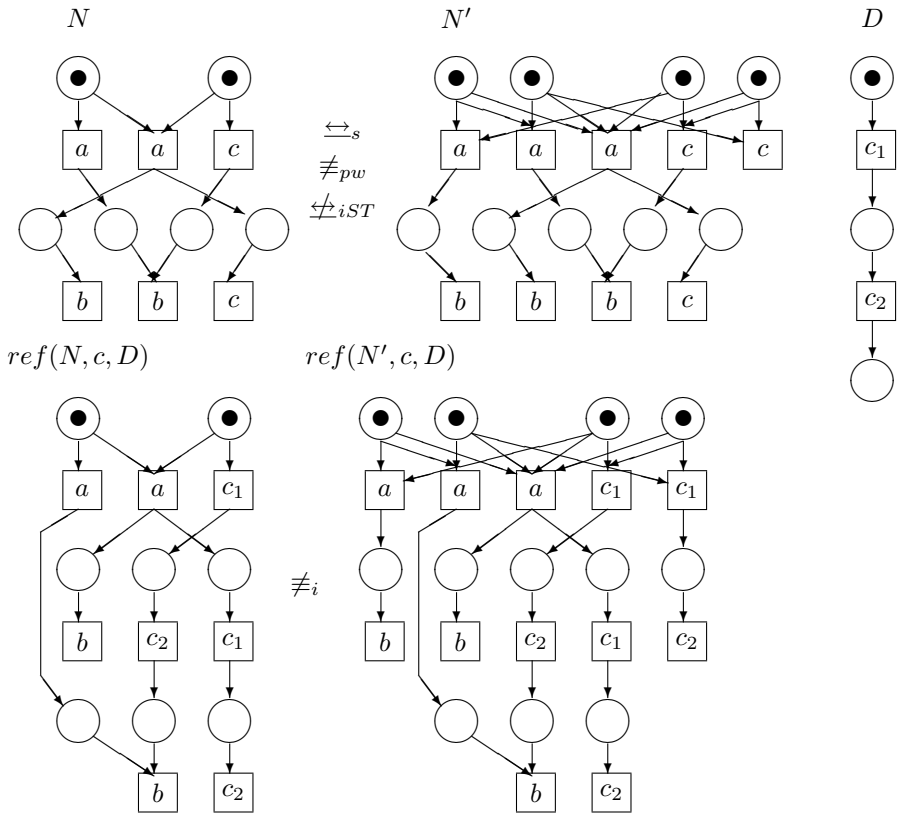

Figure 7: Equivalences from $\equiv_i$ to $\underline{\leftrightarrow}_s$ are not preserved by SM-refinements

An *SM-net* is an empty in/out net $D = \langle P_D, T_D, F_D, l_D, M_D\rangle$ s.t. $\forall t \in T_D\ |{}^\bullet t| \leq 1$ and $|t^\bullet| \leq 1$. An *SM-refinement* is an empty in/out refinement $ref(N, a, D)$ s.t. D is SM-net.

We say that some equivalence on nets is preserved by refinements, if equivalent nets remain equivalent after applying any refinement operator to them accordingly. Let us consider some examples which demonstrate that some considered in the paper equivalence notions are not preserved by SM-refinements.

Example 2

- *In Figure 7* $N \underline{\leftrightarrow}_s N'$*, but* $ref(N, c, D) \not\equiv_i ref(N', c, D)$*, since only in* $ref(N', c, D)$ *the sequence of actions* c_1abc_2 *can happen. Consequently, no equivalence from* $\equiv_i$ *to* $\underline{\leftrightarrow}_s$ *is preserved by SM-refinements.*
- *In Figure 8* $N \underline{\leftrightarrow}_{pr} N'$*, but* $ref(N, a, D) \not\underline{\leftrightarrow}_i ref(N', a, D)$*, since only in* $ref(N', a, D)$ *after occurrence of action* a_1 *action* b *can not happen. Consequently, no equivalence from* $\underline{\leftrightarrow}_i$ *to* $\underline{\leftrightarrow}_{pr}$ *is preserved by SM-refinements.*
- *In Figure 9* $N \underline{\leftrightarrow}_{pwh} N'$*, but* $ref(N, b, D) \not\underline{\leftrightarrow}_{pwh} ref(N', b, D)$*, since only in* $ref(N, b, D)$ *after action* a *action* b_1 *can happen so that action* b_2 *must depend on* a*. Consequently, equivalence* $\underline{\leftrightarrow}_{pwh}$ *is not preserved by SM-refinements.*

Theorem 3 *Let* $\leftrightarrow \in \{\equiv, \underline{\leftrightarrow}, \simeq\}$ *and* $\star \in \{i, s, pw, pom, pr, iST, pwST, pomST, prST, pwh, pomh, prh, pes, occ\}$*. For nets* $N = \langle P_N, T_N, F_N, l_N, M_N\rangle$*,* $N' = \langle P_{N'}, T_{N'}, F_{N'}, l_{N'}, M_{N'}\rangle$ *s.t.* $a \in l_N(T_N) \cap l_{N'}(T_{N'})$ *and SM-net* $D = \langle P_D, T_D, F_D, l_D, M_D\rangle$ *the following is valid:* $N \leftrightarrow_\star N' \Rightarrow ref(N, a, D) \leftrightarrow_\star ref(N', a, D)$ *iff* $\leftrightarrow_\star$ *is in oval in Figure 10.*

Proof. Omitted. □

Let us note that preservation of $\underline{\leftrightarrow}_{pomh}$ by SM-refinements was proved in [5]. Preservation by refinements of $\underline{\leftrightarrow}_{\star ST}$, $\star \in \{i, pw, pom\}$ was established in [23], but it was done in the framework of event structures, and different refinement operator was used (for example, conflicts are not allowed in substituted event structure). For other equivalences our results seem to be new.

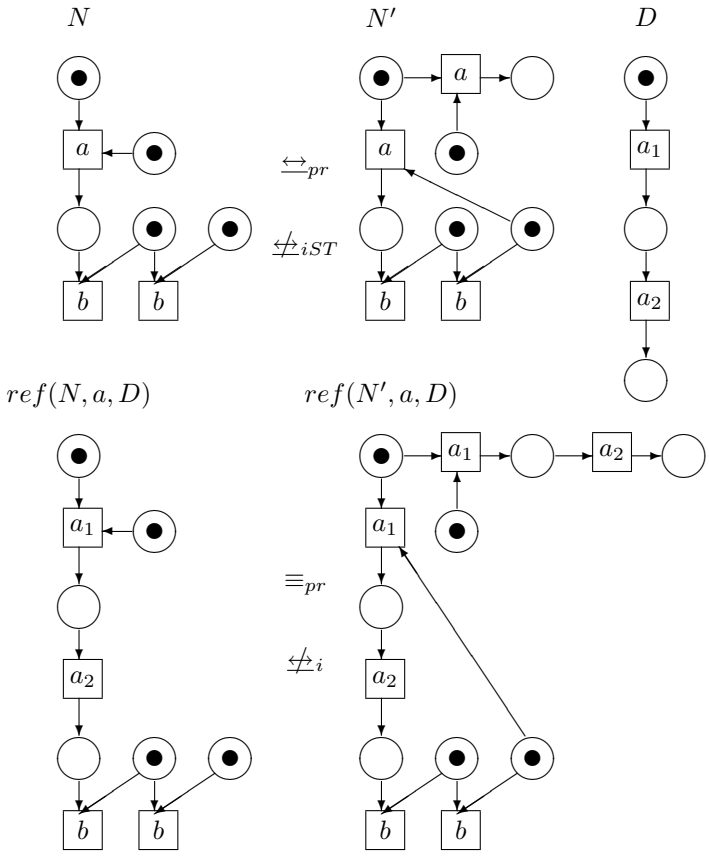

Figure 8: Equivalences from $\leftrightarrow_i$ to $\leftrightarrow_{pr}$ are not preserved by SM-refinements

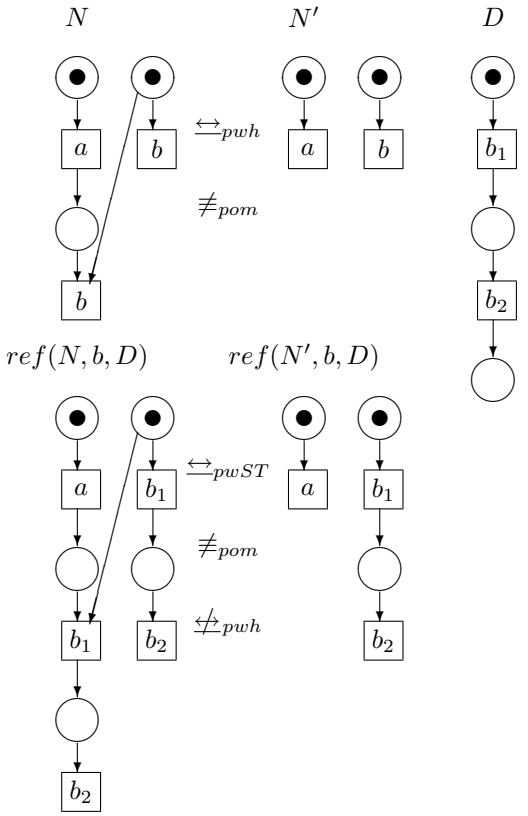

Figure 9: Equivalence $\leftrightarrow_{pwh}$ is not preserved by SM-refinements

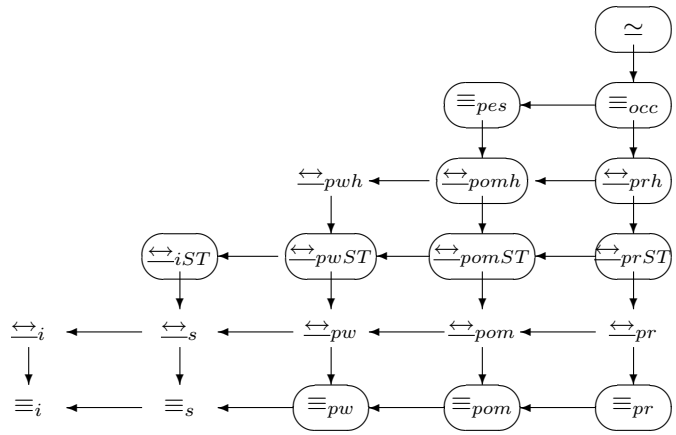

Figure 10: Preservation of equivalences by SM-refinements

7 Conclusion

In this paper we examined and supplemented by new ones a group of basic behavioural equivalences which can be used to consider systems that are modelled by Petri nets, at different abstraction levels. A correlation between all equivalence notions was investigated on whole class of Petri nets as well as on some of their subclasses: sequential nets and strictly labelled nets. All equivalences were checked for preservation by SM-refinements. So, we can use equivalence notions that are preserved by SM-refinements, for top-down design of concurrent systems.

Further research may consist in obtaining the complete picture of correlation of the equivalence notions on strictly labelled nets and investigation of the notions on T-nets (conflict-free nets). The author proved the merging of interleaving trace and ST-bisimulation equivalences on autoconcurrency-free T-nets.

We want to extend our results on nets with τ-actions also. Since it is wider class of nets, some equivalences would not be connected on such nets.

Another direction of further research consists in the investigation of place bisimulation equivalences from [1, 2, 3]. We intend to compare these equivalences with the ones we examined (for example, the relationship is unknown between place bisimulation equivalences and ST-, history preserving ones). We would like to introduce ST- and history preserving versions of place bisimulation equivalences. In addition, it is interesting to check all place-based bisimulation equivalences for preservation by refinements.

We can extend our research to back-forth bisimulation equivalences [10, 14] also, which are more strict than ones we considered. In accordance to these notions, simulation should not only be in forward direction, but in backward direction also.

Acknowledgements I would like to thank my scientific supervisor Irina B. Virbitskaite for advice to compare equivalence notions on different subclasses of Petri nets and check them for preservation by refinements and for many useful discussions which played an important role in the research.

This work was done while the author's visit to Institute of Informatics of Hildesheim. I thank the head of the institute Prof. Dr. Eike Best for providing a good working atmosphere and for useful discussions, remarks and questions.

I am indebted to researcher of the institute, Tom Thielke, for advices that helped to improve an initial variant of the paper.

References

[1] Autant C., Belmesk Z., Schnoebelen Ph. *Strong bisimularity on nets revisited. Extended abstract.* LNCS 506, p.295–312, June 1991.

[2] Autant C., Schnoebelen Ph. *Place bisimulations in Petri nets.* LNCS 616, p.45–61, June 1992.

[3] Autant C. *Petri nets for the semantics and the implementation of parallel processes.* Ph.D. thesis, Institut National Polytechnique de Grenoble, May 1993 (in French).

[4] Best E., Devillers R. *Sequential and concurrent behaviour in Petri net theory.* TCS 55, p.87–136, 1987.

[5] Best E., Devillers R., Kiehn A., Pomello L. *Concurrent bisimulations in Petri nets.* Acta Informatica 28, p.231–264, 1991.

[6] Boudol G., Castellani I. *On the semantics of concurrency: partial orders and transition systems.* LNCS 249, p.123–137, 1987.

[7] Engelfriet J. *Branching processes of Petri nets.* Acta Informatica 28(6), p.575–591, 1991.

[8] van Glabbeek R.J., Vaandrager F.W. *Petri net models for algebraic theories of concurrency.* LNCS 259, p.224–242, 1987.

[9] Hoare C.A.R. *Communicating sequential processes, on the construction of programs.* (McKeag R.M., Macnaghten A.M., eds.) Cambridge University Press, p.229–254,1980.

[10] De Nicola R., Montanari U., Vaandrager F.W. *Back and forth bisimulations.* LNCS 458, p.152–165, 1990.

[11] Nielsen M., Thiagarajan P.S. *Degrees of non-determinizm and concurrency: A Petri net view.* LNCS 181, p.89–117, December 1984.

[12] Park D.M.R. *Concurrency and automata on infinite sequences.* LNCS 104, p.167–183, March 1981.

[13] Petri C.A. *Kommunikation mit Automaten.* Ph.D. thesis, Universität Bonn, Schriften des Instituts für Instrumentelle Mathematik, 1962 (in Deutsch).

[14] Pinchinat S. *Bisimulations for the semantics of reactive systems.* Ph.D. thesis, Institut National Politechnique de Grenoble, January 1993 (in French).

[15] Pomello L. *Some equivalence notions for concurrent systems. An overview.* LNCS 222, p.381–400, 1986.

[16] Rabinovitch A., Trakhtenbrot B.A. *Behaviour structures and nets.* Fundamenta Informaticae XI, p.357–404, 1988.

[17] Sassone V., Nielsen M., Winskel G. *A classification of models for concurrency.* LNCS 715, p.82–96, 1993.

[18] Tarasyuk I.V. *Equivalences on Petri nets.* Specification, Verification and Net Models of Concurrent Systems, p.33–54, Institute of Informatics Systems, Novosibirsk, 1994.

[19] Tarasyuk I.V. *An investigation of equivalence notions on some subclasses of Petri nets.* Bulletin of the Novosibirsk Computing Center 3 (Series Computer Science), p.89–101, Computing Center, Novosibirsk, 1995.

[20] Tarasyuk I.V. *An investigation of net equivalences.* Proceedings of 4^{th} International Conference on Applied Logics, p.74–75, Irkutsk, Russia, June 1995 (in Russian).

[21] Tarasyuk I.V. *Equivalence notions for design of concurrent systems using Petri nets.* Hildesheimer Informatik Berichte 4/96, part 1, 19 p., Institut für Informatik, Universität Hildesheim, Hildesheim, Germany, 1996.

[22] Vogler W. *Failures semantics based on interval semiwords is a congruence for refinement.* LNCS 415, p.285–297, 1990.

[23] Vogler W. *Bisimulation and action refinement.* LNCS 480, p.309–321, 1991.