Fundamenta Informaticae XXI (2001) 1001–1023
IOS Press

Working with the Past: Integrating History in Petri Nets

Kees van Hee, Alexander Serebrenik [C], Natalia Sidorova, Wil van der Aalst
Department of Mathematics and Computer Science
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
k.m.v.hee, a.serebrenik, n.sidorova, w.m.p.v.d.aalst@tue.nl

Abstract. Most information systems that are driven by process models (e.g., workflow management systems) record events in event logs, also known as transaction logs or audit trails. We consider processes that not only keep track of their history in a log, but also make decisions based on this log. To model such processes we extend the basic Petri net framework with the notion of history and add guards to transitions evaluated on the process history. We show that some classes of history-dependent nets can be automatically converted to classical Petri nets for analysis purposes. These classes are characterized by the form of the guards (e.g., LTL+Past guards) and sometimes the additional requirement that the underlying classical Petri net is either bounded or has finite synchronization distances.

1. Introduction

Numerous state-of-the-art enterprize information systems contain a workflow engine, that keeps track of all events as a part of its basic functionality. In this paper, which is a revised version of [17], we consider processes that not only record the events but also make choices based on the previous events, i.e. based on its *history*. The ability of a system to change its behavior depending on its observed behavior is known as *adaptivity* and in this sense this paper is about a special class of adaptive systems.

In classical Petri nets the enabling of a transition depends only on the availability of tokens in the input places of the transition. We extend the model by recording the history of the process and introducing transition guards evaluated on the history. To illustrate the use of history, we consider a simple example of two traffic lights on crossing roads.

[C]Corresponding author

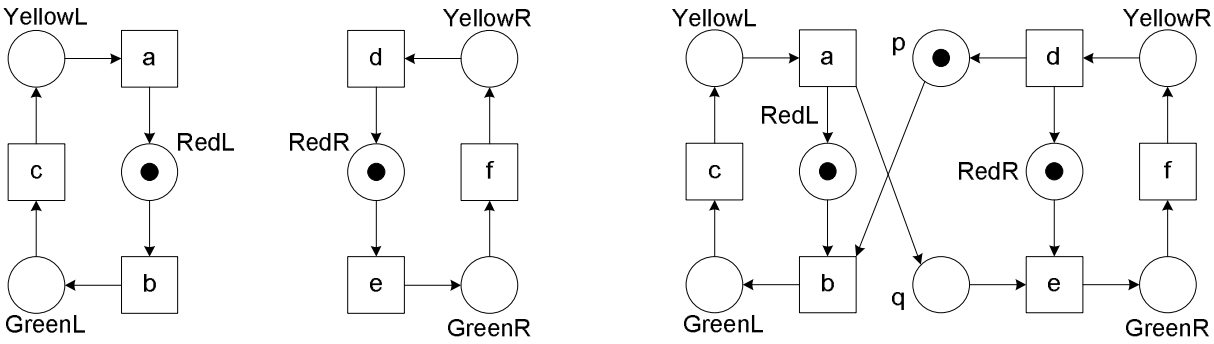

Figure 1. Traffic lights: without restrictions (*left*) and alternating (*right*).

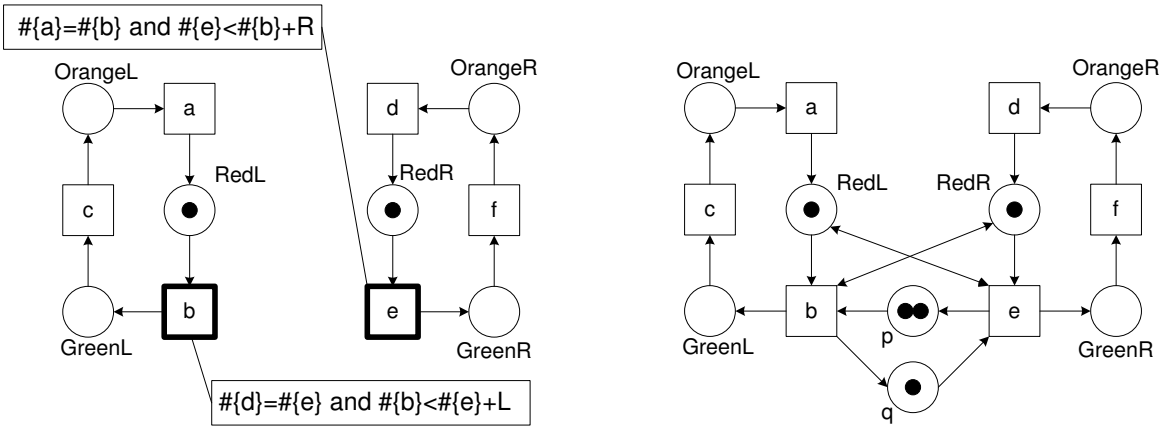

Figure 2. A history-dependent Petri net with parameters R en L (*left*) and the history guards replaced according to Theorem 6.3 for $R = 1$ en $L = 2$ (*right*).

Example 1.1. Figure 1 (left) presents two traffic lights, each modelled by a cycle of three places and three transitions. The places are modeling the states of each traffic light (red, green and yellow), and the transitions change the lights from one color to the next color. We assume that in the initial state both lights are red.

We want the system to be safe and fair, i.e., the traffic lights are never green at the same time, the right traffic light can become green at most R times more than the left traffic light, and similarly, the left traffic light can become green at most L times more than the right traffic light. Usually one takes $R = 1$ and $L = 0$, or $R = 0$ and $L = 1$ implying alternating behavior of the traffic lights. In order to obtain the alternating behavior one traditionally adds control places p and q as in the right-hand side of Figure 1. This figure models the situation with $R = 0$ and $L = 1$. Note that it is not easy to generalize this construction for arbitrary R and L.

Our approach consists in making the guards explicit as shown in left-hand side of Figure 2. To ensure safety, we require that b can fire only if the right traffic light is red, i.e., transitions d and e have fired the same number of times. The guard of b is written then as $\#\{d\} = \#\{e\}$. Similarly, e obtains the guard $\#\{a\} = \#\{b\}$. In order to guarantee fairness, we require that in any history, b fires at most L times

more than e, i.e. $\#\{b\} \leq \#\{e\} + L$, and e fires at most R times more than b, i.e., $\#\{e\} \leq \#\{b\} + R$. To ensure this we add the additional requirement $\#\{b\} < \#\{e\} + L$ to the guard of b and the additional requirement $\#\{e\} < \#\{b\} + R$ to the guard of e. This results in the history dependent Petri net shown in Figure 2 (left).

Using history we can separate the modeling of the standard process information (switching the traffic light to the following color) from additional requirements ensuring the desired behavior. Hence, we believe that introducing history-dependent guards amounts to enhanced modeling comfort. Observe also that global access to the history allows to ease modeling of synchronous choices. Assume that at a certain point a choice has to be made between transitions a and b. Assume further that the only impact of this choice is somewhere later in the process: a' has to be chosen if a has been chosen and b' has to be chosen if b has been chosen. A classical solution of this problem involves creating two places p_a and p_b with the only incoming arc coming from a (b) and the only outgoing arc leading to a' (b'). Rather than cluttering our model with additional places, we set the guard of a' (b') to demand that a (b) has been chosen before.

In this paper we consider two approaches to introduce history into the Petri net model: (1) *token history*, where each individual token carries its own history, i.e., history can be seen as special kind of color, and (2) *global history*, where there is a single centralized history and every transition guard is evaluated on it (like in our traffic lights example). Token history can be used in distributed settings where different components do not have information about the actions of other components. Global history is in fact a special case of token history for transparent systems where all components are aware of the actions of other components.

By introducing history-dependent guards, we increase the expressive power. On the traffic lights example, we can easily see that we can check the emptiness of a place using history: $RedR$ is empty if and only if $\#\{e\} - \#\{d\} = 1$. Hence, we can model inhibitor arcs and consequently our formalism is Turing complete. Since, we are interested not only in modeling but also in verification, we identify a number of important classes of global history nets (e.g. nets with LTL guards) that can be transformed to bisimilar classical Petri nets and provide corresponding transformations. For instance, the history-dependent net one the left-hand side of Figure 2 can be automatically transformed to the classical net on the right-hand side (we took $R = 1$ and $L = 2$).

Due to the Turing completeness, not every history-dependent net can be represented by a classical Petri net. We are still interested in simulation and validation of history-dependent nets. Simulation and validation are however complicated by the fact that the representation of the current state of the system requires in general an unbounded amount of memory, due to the growth of the history. We solve this problem for a Turing complete subclass of global history nets (in which we use event counting, but not event precedence in the guards) by defining a transformation to bisimilar inhibitor nets. Inhibitor nets, though being Turing complete, have a state representation of a fixed length (a marking), which makes the simulation and validation feasible.

The remainder of the paper is organized as follows. After some preliminary remarks in Section 2, we introduce the notion of *event history* together with a *history logic* in Section 3. Section 4 introduces *token history nets* and Section 5 introduces *global history nets*. In Section 6 we show how to map several subclasses of global history nets with counting formulas as guards to classical Petri nets or inhibitor Petri nets, and in Section 7 we describe a transformation of global history nets with LTL+Past guards to classical Petri nets. Finally, we review the related work and conclude the paper.

2. Preliminaries

$\mathbb{N}$ denotes the set of natural numbers and $\mathbb{Z}$ the set of integers.

Let P be a set. A *bag (multiset)* m over P is a mapping $m : P \to \mathbb{N}$. We identify a bag with all elements occurring only once with the set containing the elements of the bag. The set of all bags over P is denoted by $\mathbb{N}^P$. We use $+$ and $-$ for the sum and the difference of two bags and $=, <, >, \leq$ and $\geq$ for the comparison of bags, which are defined in a standard way. We overload the set notation, writing $\emptyset$ for the empty bag and $\in$ for the element inclusion. We write e.g. $m = 2[p] + [q]$ for a bag m with $m(p) = 2$, $m(q) = 1$, and $m(x) = 0$ for all $x \notin \{p, q\}$. As usual, $|m|$ and $|S|$ stand for the number of elements in bag m and in set S, respectively.

For (finite) *sequences* of elements over a set P we use the following notation: The empty sequence is denoted with ϵ; a non-empty sequence can be given by listing its elements. The set of all finite sequences of elements over P is denoted P^*. For a given sequence σ over P the Parikh vector $\overline{\sigma} : P \to \mathbb{N}$ maps every element of σ to the number of its occurrences in σ.

A *transition system* is a tuple $E = \langle S, Act, T \rangle$ where S is a set of *states*, Act is a finite set of *action names* and $T \subseteq S \times Act \times S$ is a *transition relation*. We say that E is finite if S is finite. A *process* is a pair (E, s_0) where E is a transition system and $s_0 \in S$ an initial state. We denote $(s_1, a, s_2) \in T$ as $s_1 \xrightarrow{a}_E s_2$, and we say that a leads from s_1 to s_2 in E. We omit E and write $s \xrightarrow{a} s'$ whenever no ambiguity can arise. For a sequence of action names $\sigma = a_1 \dots a_n$ we write $s_1 \xrightarrow{\sigma} s_2$ when $s_1 = s^0 \xrightarrow{a_1} s^1 \xrightarrow{a_2} \dots \xrightarrow{a_n} s^n = s_2$. Next, $s_1 \xrightarrow{*} s_2$ means that there exists a sequence $\sigma \in T^*$ such that $s_1 \xrightarrow{\sigma} s_2$. We say that s_2 is *reachable* from s_1 if and only if $s_1 \xrightarrow{*} s_2$. Finally, the language of a process (E, s_0), denoted $\mathfrak{L}(E, s_0)$, is defined as $\{\sigma \mid \sigma \in T^*, \exists s : s_0 \xrightarrow{\sigma} s\}$.

Definition 2.1. Let $E_1 = \langle S_1, Act, T_1 \rangle, E_2 = \langle S_2, Act, T_2 \rangle$ be transition systems. A relation $R \subseteq S_1 \times S_2$ is a *simulation* if and only if for all $s_1, s_1' \in S_1$, $s_2 \in S_2$, $s_1 \xrightarrow{a}_{E_1} s_1'$ implies that $s_2 \xrightarrow{a}_{E_2} s_2'$ and $s_1' \; R \; s_2'$ for some $s_2' \in S_2$.

E_1 and E_2 are *bisimilar* if there exists a relation $R \subseteq S_1 \times S_2$ such that both R and R^{-1} are simulations. Two processes (E_1, s_1) and (E_2, s_2) are *bisimilar* if there exists a relation $R \subseteq S_1 \times S_2$ such that both R and R^{-1} are simulations, and $s_1 \; R \; s_2$.

Next we introduce a number of notions related to Petri nets.

Definition 2.2. A *Petri net* N over a fixed set of labels Σ is a tuple $\langle P, T, F, \Lambda \rangle$, where: (1) P and T are two disjoint non-empty finite sets of *places* and *transitions* respectively; we call the elements of the set $P \cup T$ *nodes* of N; (2) $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ is a *flow relation* mapping pairs of places and transitions to the naturals; (3) $\Lambda : T \to \Sigma$ is a *labeling function* that maps transitions of T to action labels from Σ.

An *inhibitor net* is a tuple $\langle P, T, F, \Lambda, I \rangle$ such that $\langle P, T, F, \Lambda \rangle$ is a Petri net and $I \subseteq P \times T$ is a set of *inhibitor arcs*.

We present nets with the usual graphical notation. For any pair of nodes x, y with $F(x, y) \geq 1$, we say that (x, y) is an arc with *weight* $F(x, y)$.

Given a transition $t \in T$, the *preset* ${}^\bullet t$ and the *postset* $t^\bullet$ of t are the *bags* of places where every $p \in P$ occurs $F(p, t)$ times in ${}^\bullet t$ and $F(t, p)$ times in $t^\bullet$. Analogously we write ${}^\bullet p, p^\bullet$ for pre- and postsets of places.

A marking m of N is a bag over P; markings are states (configurations) of a net. A pair (N, m) is called a *marked* Petri net. A transition $t \in T$ is *enabled* in marking m if and only if ${}^\bullet t \leq m$ and moreover, for inhibitor nets, $m(p) = 0$ for any p such that $(p, t) \in I$. An enabled transition t may *fire*. This results in a new marking m' defined by $m' \stackrel{\text{def}}{=} m - {}^\bullet t + t^\bullet$. We interpret a labeled Petri net N as a transition system/process $\langle \mathbb{N}^P, \Lambda(T), \longrightarrow \rangle$ / $(\langle \mathbb{N}^P, \Lambda(T), \longrightarrow \rangle, m_0)$ respectively, where markings play the role of states and labels of the firing transitions play the role of action names. The notion of reachability for Petri nets is inherited from the transition systems. We denote the set of all markings reachable in net N from marking m as $\mathcal{R}_N(m)$. We will drop N and write $\mathcal{R}(m)$ when no ambiguity can arise. A marked net (N, m_0) is called *bounded* if its reachability set is finite.

Finally, the notion of bisimilarity for marked Petri nets is also inherited from processes.

3. Event History and History Logic

In this section we present the general notion of event history. In the coming sections we investigate two kinds of nets that use event history: *token history nets* and *global history nets*.

One might expect an event history to be a totally ordered series of events. However, information on relative order of events registered by different components might be missing. Therefore, we define a history as a partial order.

Definition 3.1. Given a set of action labels Σ, a *history* is a labeled poset, i.e., a triple $\langle E, \prec, \lambda \rangle$, where E is a set of *events* coming from a fixed universe $\mathcal{E}$, $\prec$ is a partial order on E and $\lambda : E \to \Sigma$ is a *labeling function*. If $E = \emptyset$ the corresponding history is called *the empty history* and denoted by ϵ.

Two histories $\langle E_1, \prec_1, \lambda_1 \rangle$ and $\langle E_2, \prec_2, \lambda_2 \rangle$ are *consistent* if and only if the transitive closure of $\prec_1 \cup \prec_2$ is a partial order for $E_1 \cup E_2$ and $\lambda_1(e)$ coincides with $\lambda_2(e)$ for any $e \in E_1 \cap E_2$.

The class of all histories over the given set of action labels Σ is denoted $\mathcal{H}_\Sigma$.

We define two operations to create a new history out of existing histories: *extension and union*.

Definition 3.2. The *extension* $\langle E, \prec, \lambda \rangle :: \ell$ of a history $\langle E, \prec, \lambda \rangle$ with an event labeled by ℓ is the history $\langle E \cup \{e\}, \prec_\ell, \lambda_\ell \rangle$, where e is a new event,[1] $\prec_\ell$ is defined as $\prec \cup \{(x, e) \mid x \in E\}$ and λ_ℓ maps e to ℓ and coincides with λ on E.

The *union* $\langle E_1, \prec_1, \lambda_1 \rangle \cup \langle E_2, \prec_2, \lambda_2 \rangle$ *of consistent histories* is defined as $\langle E_1 \cup E_2, \prec, \lambda_1 \cup \lambda_2 \rangle$, where $\prec$ is the transitive closure of $\prec_1 \cup \prec_2$.

These operations will be used in the next sections on token history and global history for Petri nets. In global history nets each firing of a transition extends the global history. In token history nets, tokens created by a transition firing carry the union of histories of the consumed tokens extended with the firing event.

Next we present a language of history-dependent predicates that will be used in the guards of history-dependent nets. From here on we assume a countable set *Var* of variables to be given.

[1] Note that it is essential that e is a "fresh" identifier not present in E but also not used in any "known" history.

Definition 3.3. Given a set Σ of labels and $x \in \mathit{Var}$, we define a formula φ, a term q and a label expression l over Σ as follows:

$$\begin{array}{lcllllll} \varphi & ::= & \mathit{false} & \mid & \varphi \Rightarrow \varphi & \mid & x \preceq x & \mid & q < q & \mid & l == l \\ q & ::= & \mathbb{N} & \mid & (\#\mathit{Var} : \varphi) & \mid & (q + q) & & & & \\ l & ::= & \Sigma & \mid & \lambda(x) & & & & & & \end{array}$$

Sets of formulas, terms and label expressions over Σ are denoted as $\mathcal{F}_\Sigma$, $\mathcal{Q}_\Sigma$ and $\mathcal{L}_\Sigma$, respectively.

Using the definition above we can define the following short-hand notations in the standard way: true, $\neg$, $\wedge$, $\vee$, $>$, $\geq$, $\leq$, $=$ (comparisons of terms). We omit brackets if this does not introduces ambiguities. The counting operator $\#$ is powerful enough to express the standard quantifiers: We write $\exists x : \varphi$ for $(\#x : \varphi) > 0$ and $\forall x : \varphi$ for $(\#x : \varphi) = (\#x : \mathit{true})$. For a finite set of labels $S = \{s_1, \ldots, s_n\}$, $\ell \in S$ stands for $(\ell == s_1 \vee \ldots \vee \ell == s_n)$ and $\#S$ stands for $(\#x : \lambda(x) \in S)$. Finally $e_1 \prec e_2$ means that $(e_1 \preceq e_2) \wedge \neg(e_2 \preceq e_1)$.

In order to define the semantics we introduce the notion of an *assignment* defined as a mapping of variables from *Var* to events from E. Given a variable x, an event e and an assignment ν, $\nu[x \to e]$ denotes the assignment that coincides with ν for all variables except for x which is mapped to e.

Definition 3.4. Given a history $H = \langle E, \prec, \lambda \rangle$ and an assignment ν, the *evaluation* eval and the truth value of a *formula* are defined by mutual structural induction. The evaluation function eval maps a *term* q to $\mathbb{N}$ as follows:

$$\mathit{eval}(H, \nu, q) = \begin{cases} q & \text{if } q \in \mathbb{N}; \\ \left|\{e \in E \mid \langle H, \nu[x \to e]\rangle \models \varphi\}\right| & \text{if } q \text{ is } \#x : \varphi; \\ \mathit{eval}(H, \nu, q_1) + \mathit{eval}(H, \nu, q_2) & \text{if } q \text{ is } q_1 + q_2. \end{cases}$$

Similarly, eval maps a *label expression* l to Σ:

$$\mathit{eval}(H, \nu, l) = \begin{cases} l & \text{if } l \in \Sigma; \\ \lambda(\nu(x)) & \text{if } l \text{ is } \lambda(x). \end{cases}$$

Finally, the truth value of a *formula* is defined as follows:

- $\langle H, \nu \rangle \models \mathit{false}$ is always false;
- $\langle H, \nu \rangle \models \varphi_1 \Rightarrow \varphi_2$ if not $\langle H, \nu \rangle \models \varphi_1$ or $\langle H, \nu \rangle \models \varphi_2$;
- $\langle H, \nu \rangle \models x_1 \preceq x_2$ if $\nu(x_1) \prec \nu(x_2)$ or $\nu(x_1)$ coincides with $\nu(x_2)$;
- $\langle H, \nu \rangle \models q_1 < q_2$ if $\mathit{eval}(H, \nu, q_1) < \mathit{eval}(H, \nu, q_2)$ ($<$ is the standard order on the naturals);
- $\langle H, \nu \rangle \models l_1 == l_2$ if $\mathit{eval}(H, \nu, l_1)$ coincides with $\mathit{eval}(H, \nu, l_2)$.

One can show that for *closed terms* and *formulas*, i.e., terms and formulas where all variables appear in the scope of $\#$, the result of the evaluation does not depend on ν. Therefore, for a closed term q we

also write $eval(H, q)$ and for a closed formula φ we also write $H \models \varphi$. The set of closed formulas over Σ is denoted $\mathcal{CF}_\Sigma$.

To illustrate our language, we return to the traffic light example from Figure 2. The guards of transitions are formulated as in Definition 3.3. For the case $R = 0$, $L = 1$ the guard of the transition b can alternatively be formulated with the use of the precedence operator $\prec$ as $\forall x : (\lambda(x) = b \Rightarrow \exists y : (\lambda(y) = e \wedge x \prec y))$.

4. Token History Nets

In this section we introduce token history nets as a special class of colored Petri nets [18] with history as color. The tokens of an initial marking have an empty history and every firing of a transition t produces tokens carrying the union of the histories of the consumed tokens extended with the last event, namely the firing of transition t labeled by $\Lambda(t)$.

Definition 4.1. A *token history net* N is a tuple $\langle P, T, F, \Lambda, g\rangle$ such that $N_P = \langle P, T, F, \Lambda\rangle$ is a labeled Petri net and $g : T \to \mathcal{CF}_{\Lambda(T)}$ defines the *transition guards*.
The semantics of a token history net is given by the transition system defined as follows:

$Color$ is the set of possible histories $\langle E, \prec, \lambda\rangle$ over the label set $\Lambda(T)$. A *state* m of a token history net N is a bag of tokens with histories as token colors, i.e., a marking $m : (P \times Color) \to \mathbb{N}$.

The *transition relation* is specified by: $m \xrightarrow{a} m'$ if and only if there exist a transition t with $\Lambda(t) = a$, a history H and two bags $cons$ and $prod$ of tokens such that:

- $H = \bigcup_{(p,c)\in cons} c$ (H is the unified history),
- $cons \leq m$ (tokens from $cons$ are present in m),
- $\sum_{(p,c)\in cons}[p] = {}^\bullet t$ (sufficiently many tokens are consumed from the right places),
- $prod = \sum_{p\in t^\bullet}[(p, H :: \Lambda(t))]$ ($prod$ is the bag of tokens to be produced),
- $m' = m - cons + prod$, and
- $H \models g(t)$ (i.e., the guard evaluates to true given the unified history H).

A token history net is thus defined by attaching a guard to all transitions of a classical Petri net. A transition guard is evaluated on the union H of histories of consumed tokens. Recall that the union of two histories is defined for consistent histories only. We will call a marking *consistent* if the union of all its token histories is defined. The following lemma states that consistency of markings is an invariant property (observe that a transition firing cannot destroy consistency).

Lemma 4.1. Let m be a consistent marking and $m \xrightarrow{*} m'$ for some marking m'. Then m' is consistent.

Proof:
Assume for the sake of contradiction that $m \xrightarrow{a} m'$, m is consistent and m' is not. Then, the only history produced by the firing is $H :: a$, where H is the union of the histories of the tokens consumed. Let $H' = \langle E', \prec', \lambda'\rangle$ be a history of token in m' such that $H :: a$ and H' are not consistent. Let $H :: a$

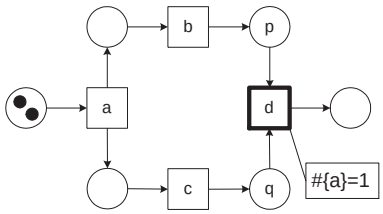

Figure 3. A token history net.

be $\langle E, \prec, \lambda \rangle$. Then, either the transitive closure of $\prec \cup \prec'$ is not a partial order on $E \cup E'$, or there exists $e \in E \cap E'$ such that $\lambda(e)$ does not coincide with $\lambda'(e)$. However, semantics of the extension operation implies in both cases that H and H' are also inconsistent. However, H' is a color of a token in m and H is a union of histories of tokens in m. Hence, by Definition 4.1 m is inconsistent, contradicting the assumption. □

To conclude this section we illustrate the semantics of token history nets.

Example 4.1. Consider the token history net in Figure 3. Firings of transition d are are allowed if and only if there is only one event labeled by a in the union of the histories of tokens consumed from places p and q, i.e. tokens on p and q originate from the same initial token. Let the sequence $abcabc$ fire from the initial marking, which results in the marking $m = [(p, H_1)] + [(p, H_2)] + [(q, H_3)] + [(q, H_4)]$ with $H_1 = \langle \{e_1, e_2\}, \{e_1 \prec e_2\}, \{(e_1, a), (e_2, b)\} \rangle$, $H_2 = \langle \{e_4, e_5\}, \{e_4 \prec e_5\}, \{(e_4, a), (e_5, b)\} \rangle$, $H_3 = \langle \{e_1, e_3\}, \{e_1 \prec e_3\}, \{(e_1, a), (e_3, c)\} \rangle$ and $H_4 = \langle \{e_4, e_6\}, \{e_4 \prec e_6\}, \{(e_4, a), (e_6, c)\} \rangle$. The transition labeled d can fire consuming tokens $[(p, H_1)]$ and $[(q, H_3)]$ since the tokens share event e_1 in their history. The produced token is $[(s, H_5)]$ with $H_5 = \langle \{e_1, e_2, e_3, e_7\}, \{e_1 \prec e_2, e_1 \prec e_3, e_1 \prec e_7, e_2 \prec e_7, e_3 \prec e_7\}, \{(e_1, a), (e_2, b), (e_3, c), (e_7, d)\} \rangle$. This transition cannot fire on e.g. $[(p, H_1)]$ and $[(q, H_4)]$ since the union $H_1 \cup H_4$ contains two events (e_1 and e_4) labeled by a while the guard specifies that the number of a events should be one ($\#\{a\} = 1$). Token history allows thus distinguishing between tokens originating from different firings of the same transition, i.e., mimicking another popular color, namely case identifiers.

5. Global History Nets

In this section we introduce global history nets, where history is a separate object accessible when the guards of transitions are evaluated.

Definition 5.1. A *global history net* N is a tuple $\langle P, T, F, \Lambda, g \rangle$ such that $N_P = \langle P, T, F, \Lambda \rangle$ is a labeled Petri net and $g : T \to \mathcal{CF}_{\Lambda(T)}$ defines the *transition guards*.
The semantics of global history nets is defined as follows:

A *state* of N is a pair (m, H) where m is a marking of N_P and H is a history over $\Lambda(T)$. The *transition relation* is specified by: $(m, H) \xrightarrow{a} (m', H')$ if and only if there exists $t \in T$ such that $\lambda(t) = a$, ${}^\bullet t \leq m$, $H \models g(t)$, $m' = m - {}^\bullet t + t^\bullet$ and H' is $H :: \Lambda(t)$.

Given a global history net N we denote by $\mathcal{S}(N)$ the set of all states of the net. Analogously to marked Petri nets we consider marked global history nets being pairs $(N, (m, H))$ such that N is a global history net and $(m, H) \in \mathcal{S}(N)$. The set of states reachable from (m, H) in N is denoted $\mathcal{R}_N((m, H))$; the set of states reachable from an initial state (m_0, ϵ) is thus $\mathcal{R}_N((m_0, \epsilon))$. We denote by $\mathcal{H}_N$ the set of histories reachable from (m_0, ϵ), i.e., $\{H | \exists m\ (m, H) \in \mathcal{R}_N((m_0, \epsilon))\}$. Elements of $\mathcal{H}_N$ are linearly-ordered finite sets of labeled events.

The interleaving semantics results in the following property:

Proposition 5.1. Let $N = \langle P, T, F, \Lambda, g\rangle$ be a global history net and $(m, \langle E, \prec, \lambda\rangle) \in \mathcal{R}_N((m_0, \epsilon))$. Then $\prec$ is a total order on E.

Note that history does not contain information which transitions exactly have fired, but labels of those transitions only. Therefore, knowing the initial marking and the history, we cannot reconstruct the current marking in general. However, it can easily be seen that if Λ is injective the current marking can be derived from the initial marking an history.

Proposition 5.2. Let $N = \langle P, T, F, \Lambda, g\rangle$ be a global history net such that Λ is injective. Then, for a given H: $(m_1, H), (m_2, H) \in \mathcal{R}_N((m_0, \epsilon))$ implies $m_1 = m_2$.

This proposition implies that by using global history nets with *injective labeling* we are able to express conditions on the marking. To illustrate this, we introduce $\#^\bullet p$ as a shorthand for $\sum_{t \in {}^\bullet p} \#\{\Lambda(t)\}$ for some place p, i.e., $\#^\bullet p$ is the number of tokens *produced* to the place p. Similarly, $\#p^\bullet$ denotes $\sum_{t \in p^\bullet} \#\{\Lambda(t)\}$, i.e., the number tokens *consumed* from p according to the history. (Note that the sum is taken over a bag.) Now, let m_0 be the initial marking of a global history net N where Λ is injective, and assume $(m, H) \in \mathcal{R}_N((m_0, \epsilon))$. Clearly, $m(p) = m_0(p) - \#p^\bullet + \#^\bullet p$ for any $p \in P$. Hence, in transition guards we can express any condition on the current state. For example, by adding the condition $m_0(p) - \#p^\bullet + \#^\bullet p = 0$ we can simulate inhibitor arcs. Since inhibitor nets are known to be Turing complete (cf. [26]), global history nets with unique labels are Turing complete as well.

Corollary 5.1. Global history nets $N = \langle P, T, F, \Lambda, g\rangle$ are Turing complete.

Next we discuss the implications of Corollary 5.1 on the expressive power of token history nets.

5.1. Token history vs. global history

Observe that in general it is impossible to derive the corresponding token histories from the history of a global history net. Consider the net from Figure 3 as a global history net and suppose that its global history is $aabc$. One cannot derive whether the tokens on places p and q will share the history event labeled by a or not. On the other hand, in general it is impossible to reconstruct the corresponding global history from a given a marking of a token history net, since no information is available on the order of truly concurrent firings. So marking m from Example 4.1 can be obtained as a result of firing sequences $abcabc$, $aabbcc$, $abacbc$, etc. and have the corresponding global history. We can however *mimic* a global history net with a token history net.

The key idea behind our construction is adding a new place p^* with one initial token, connected to all transitions. Since the token in p^* is updated at each firing, it will keep a *global log*. Since all transitions

are connected to p^*, their guards will be evaluated on the same history as in the original global history net. We formalize this intuition in the following definition.

Definition 5.2. Let $N = \langle P, T, F, \Lambda, g\rangle$ be a global history net with initial marking m_0. A token history net $N' = \langle P', T, F', \Lambda, g\rangle$ with initial marking m'_0 is called the *log extension* of N if $P' = P \cup \{p^*\}$ (with $p^* \notin P$ being the new place), $F'(n_1, n_2) = F(n_1, n_2)$ for $(n_1, n_2) \in (P \times T) \cup (T \times P)$ and $F'(n_1, n_2) = 1$ for $(n_1, n_2) \in (\{p^*\} \times T) \cup (T \times \{p^*\})$, and $\forall p \in P : m'_0((p, \epsilon)) = m_0(p)$, $m'_0((p^*, \epsilon)) = 1$ and $m'_0((p, x)) = 0$ in all other cases.

It is easy to show that N and N' are indeed bisimilar.

Lemma 5.1. $(N, (m_0, \epsilon))$ and (N', m'_0) as above are bisimilar.

Proof:
Recall that we interpret marked Petri nets as processes, i.e., we need to show that $(\langle \mathbb{N}^P, \Lambda(T), \longrightarrow \rangle, (m_0, \epsilon))$ and $(\langle \mathbb{N}^{P\cup\{p^*\}}, \Lambda(T), \Longrightarrow\rangle, m'_0)$ are bisimilar. As a simulation relation we choose: $(m, H)\, R\, m'$ if and only if $(p^*, H) \in m' \wedge \forall p \in P : m(p) = \#\{x|(p, x) \in m'\}$. By definition of m'_0, $(m_0, \epsilon)\, R\, m'_0$.

Let m_1, m_2, m'_1 and H_1, H_2 be such that $(m_1, H_1)\ R\ m'_1$ and $(m_1, H_1) \xrightarrow{a} (m_2, H_2)$. Since $a \in \Lambda(T)$ there exists $t \in T$, enabled in (m_1, H_1) with $\Lambda(t) = a$. However, t is enabled in (m_1, H_1) if and only if $m_1 \geq {}^\bullet t$ and $H_1 \models g(t)$. Since $(m_1, H_1)\, R\, m'_1$, $m'_1(p^*) = [H_1]$ and for all $p \in P$ there exists x such that $m'_1((p, x)) = m_1(p)$. Hence, t is enabled in m'_1 as well. Let m'_2 be such that $m'_1 \stackrel{a}{\Longrightarrow} m'_2$. By definition of the firing relation for global history nets, $m_2 = m_1 - {}^\bullet t + t^\bullet$ and $H_2 = H_1 :: a$. Definition of the firing relation for token history nets implies therefore that $(m_2, H_2)\ R\ m'_2$. Hence R is a simulation relation. Similarly, one can show that R^{-1} is a simulation relation as well, implying that the nets are bisimilar. □

Corollary 5.2. Token history nets are Turing complete.

Proof:
By Lemma 5.1 and Corollary 5.1. □

It is easy to map both a token history net and a global history net onto a colored Petri net with token values being histories. Figure 4 shows a screenshot of CPN Tools simulating the two traffic lights from Example 1.1 controlled by history. Note that we added place p^* to store the global history.

The remainder of this paper focuses on global history nets.

6. Global History Nets with Counting Formulas Guards

In this section we consider global history nets with guards being *counting formulas*, i.e., formulas that do not explore the precedence of events $\prec$. Formally, a *counting formula* φ is defined as

$$\varphi ::= \mathit{false} \mid \varphi \Rightarrow \varphi \mid q < q \mid l == l$$

where q and l are terms and label expressions as in Definition 3.3.

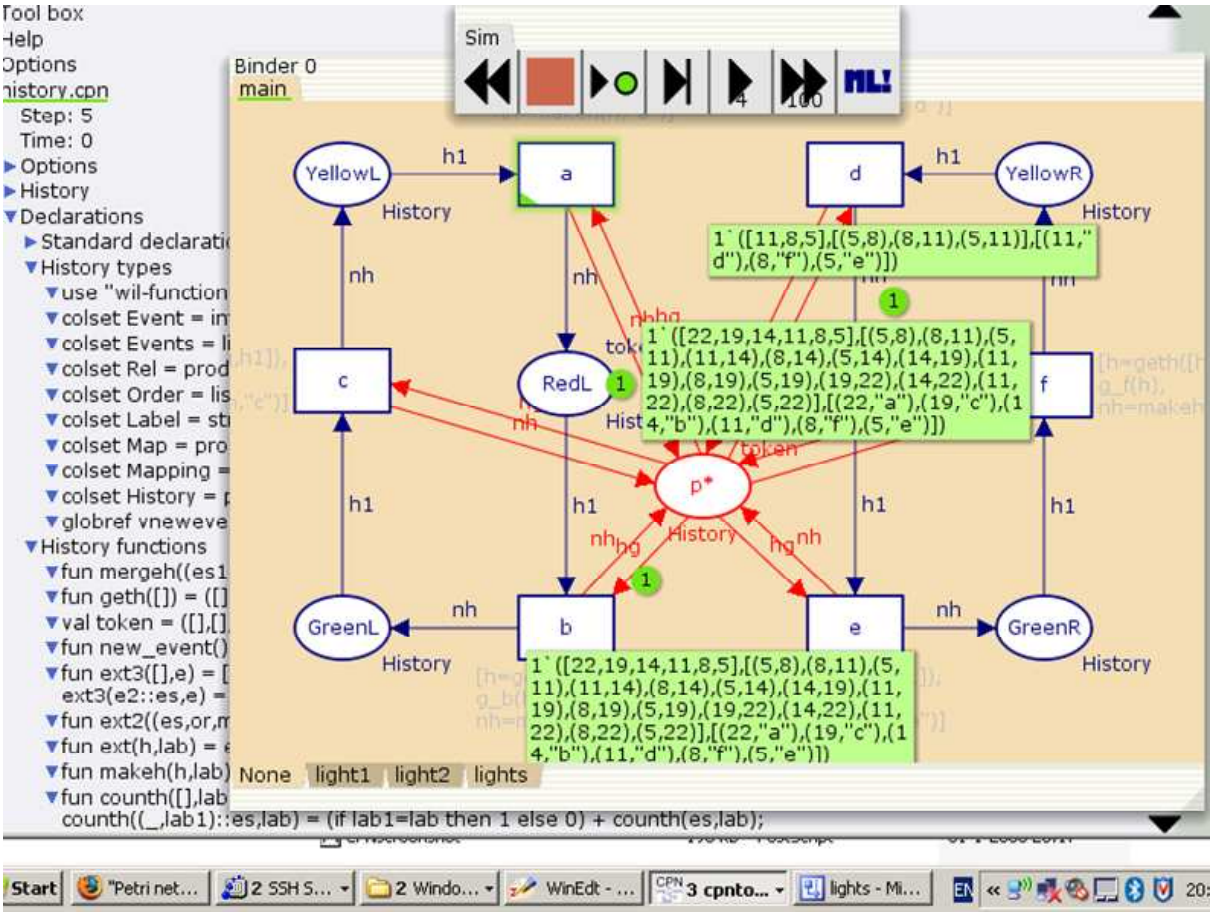

Figure 4. The history-dependent Petri net with parameters R en L and using a global place to record history simulated using CPN Tools.

Note that global history nets with counting formulas guards are Turing complete since they allow zero testing on the marking of a place. To facilitate simulation and validation of these nets, we show that every global history net with counting formulas guards can be transformed into a bisimilar inhibitor net. Furthermore, we identify conditions on the global history net implying that the net can be translated to a bisimilar classical Petri net.

6.1. Nets with counting formulas as guards vs. inhibitor nets

We start with the simplest form of counting formulas, namely $(\#A)\ \rho\ (\#B + k)$ for some $A, B \subseteq \Sigma$, $\rho \in \{\geq, \leq\}$ and $k \in \mathbb{N}$. For the sake of brevity we call these expressions *basic counting formulas* (over A and B). Note that taking B equal to $\emptyset$ we obtain $(\#A)\ \rho\ k$ (since $\#\emptyset = 0$).

Lemma 6.1. Let (N, m_0) be a marked global history net with $N = \langle P, T, F, \Lambda, g\rangle$ such that for any $t \in T$, $g(t)$ is a basic counting formula. There exists a marked inhibitor net (N', m_0') bisimilar to (N, m_0).

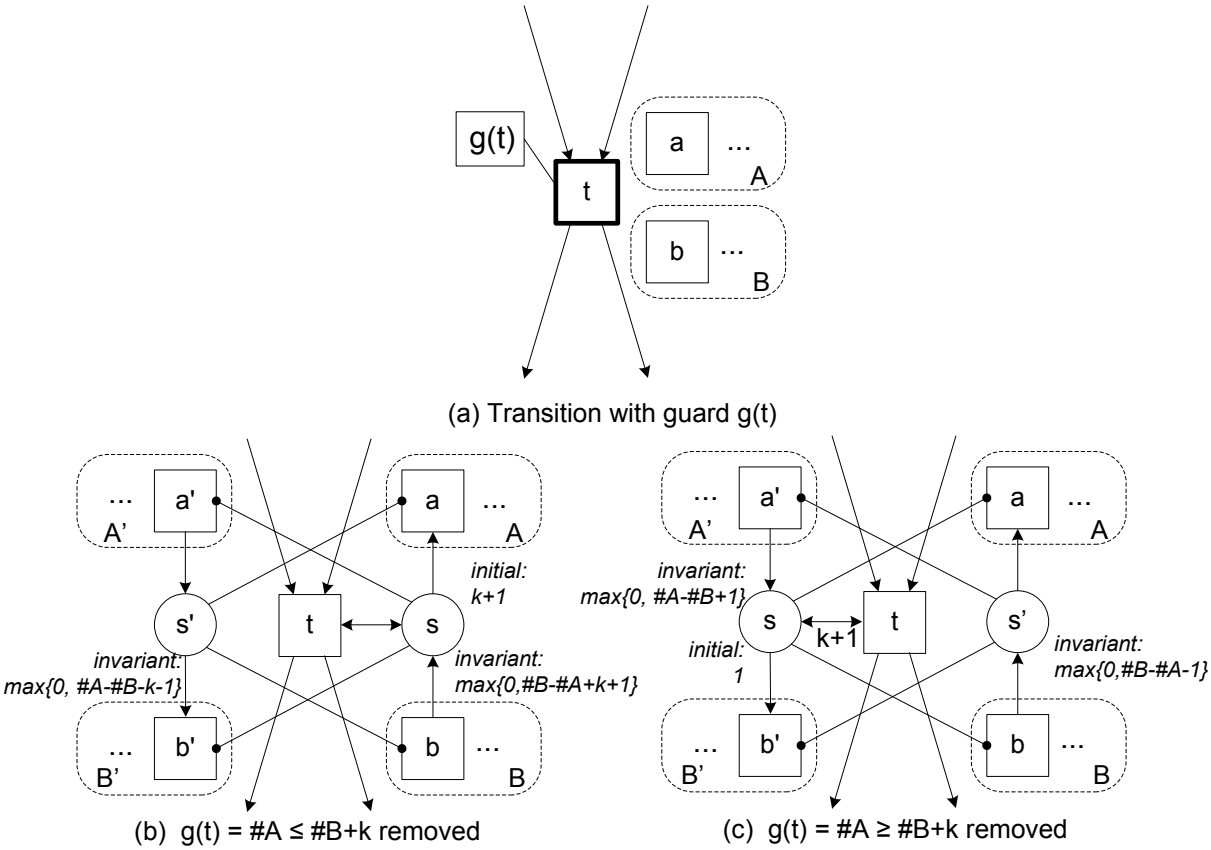

Figure 5. Replace the guard by places s and s', duplicate transitions, and inhibitor arcs.

Proof:
We apply to the net (N, m_0) an iterative process of guard elimination resulting in (N', m_0'). At every iteration step we will replace one of the transition guards of the current net by $true$, adding some places and transitions to preserve the net behavior. The process terminates when all guards are $true$, i.e. we obtained a regular inhibitor net.

Let t be a transition whose guard we eliminate at the next step and let $g(t)$ be $\#A\ \rho\ \#B+k$ for some $A, B \subseteq \Sigma$, $\rho \in \{\geq, \leq\}$ and $k \in \mathbb{N}$. We can assume that A and B are disjoint, since $(\#A)\ \rho\ (\#B + k)$ if and only if $(\#(A \setminus B))\ \rho\ (\#(B \setminus A) + k)$.

Figure 5 shows the basic idea of the eliminating a transition with guard $g(t)$. Consider, for example the case ρ equals $\leq$. Figure 5(a) sketches the relevant features of the initial net and Figure 5(b) shows the net where guard $g(t) = (\#A \leq \#B + k)$ is eliminated. Note that A and B refer to the sets of transitions having a label from A respectively B. For the purpose of illustration, we show a transition with label $a \in A$ and a transition with label $b \in B$ (note that may not be such transitions).

In order to mimic the guard $g(t)$, we add places s and s', where s will contain $\max\{0, \#B - \#A + k+1\}$ tokens while s' will contain $\max\{0, \#A-\#B-k-1\}$ tokens. Note that $g(t) = (\#A \leq \#B+k)$ evaluates to $true$ if and only if there is at least one token in s, therefore we add a bidirectional arc between s and t. In the initial marking $m_0(s) = k + 1$ and $m_0(s') = 0$.

To support the computations on s and s', we need to duplicate all transitions with a label in $A \cup B$, i.e., for every v such that $\Lambda(v) \in A$ or $\Lambda(v) \in B$ we add a transition v' with ${}^\bullet v' = {}^\bullet v$, $v'^\bullet = v^\bullet$, and $\Lambda(v') = \Lambda(v)$. The resulting sets of transitions are referred to as A' and B' in Figure 5(b). It is essential to note that the transitions are mutually exclusive in terms of enabling, i.e., for any marking m and any transition v such that $\Lambda(v) \in A$ or $\Lambda(v) \in B$ either m enables v or m enables v'. Moreover, s and s' are non-blocking, i.e., if $v \in T$ was enabled in the original net, then either v or v' is enabled in the net with inhibitors.

The construction for ρ equal to $\geq$ is similar as shown in Figure 5(c). Note that the initial marking has been updated and that t now tests for the presence of $k+1$ tokens in s where s always contains $\max\{0, \#A - \#B + 1\}$ tokens.

As mentioned above, the transformation is repeatedly applied until no guarded transition is left. The bisimilarity of (N, m_0) and (N', m'_0) can be trivially proven by induction. □

Our interest in transitions with basic counting formulas as guards is motivated by the fact that any non-trivial counting formula is equivalent to a disjunction of conjunctions of basic counting formulas.

Lemma 6.2. Any counting formula φ can be written in disjunctive normal form where the literals are positive basic counting formula (i.e. without negations), so $\varphi \equiv \mathit{true}$ or $\varphi \equiv \mathit{false}$ or $\varphi \equiv \bigvee_i (\bigwedge_j \psi_{i,j})$ and each $\psi_{i,j}$ is a basic counting formula.

Theorem 6.1. Let (N, m) be a marked global history net with $N = \langle P, T, F, \Lambda, g \rangle$ such that for any $t \in T$, $g(t)$ is a counting formula. There exists a marked inhibitor net (N', m') bisimilar to (N, m).

Proof:
By Lemma 6.2 we consider only disjunctions of conjunctions of basic counting formulas. First we transform our net to a net where all guards are conjunctions of basic counting formulas by applying the following construction: Every transition t with a guard $\varphi \vee \psi$ is replaced by transitions t_φ with the guard φ, and t_ψ with the guard ψ, where ${}^\bullet t_\varphi = {}^\bullet t_\psi = {}^\bullet t$, $t_\varphi^\bullet = t_\psi^\bullet = t^\bullet$ and $\Lambda(t_\varphi) = \Lambda(t_\psi) = \Lambda(t)$.

At the next step we eliminate conjuncts from the guards one by one by applying the construction depicted in Figure 5. The only difference is that we apply the construction to a transition t with a guard $(\#A \ \rho \ \#B + k) \wedge \varphi$, and the guard of t in the resulting net is then φ. □

6.2. Boundedness and analyzability of global history nets with counting formulas as guards

Although the construction referred to in the proof of Theorem 6.1 is applicable to any global history net with counting formulas as guards, the resulting net contains inhibitor arcs and therefore, cannot be analyzed easily because of Turing completeness. However, it is well-known that inhibitor arcs can be eliminated in *bounded* inhibitor nets. Boundedness of classical or inhibitor Petri nets is in principle finiteness of its state space. Hence it is interesting to explore "finiteness notions" for global history nets.

Finiteness of $\mathcal{R}_N((m_0, \epsilon))$ for a global history net $N = \langle P, T, F, \Lambda, g \rangle$ does not imply boundedness of the underlying Petri net $(\langle P, T, F, \Lambda \rangle, m_0)$ and vice versa. In Figure 6 we see two global history nets. The underlying Petri net shown in Figure 6(a) is unbounded, while the global history net has a finite state space due to the transition guard. The underlying Petri net shown in Figure 6(b) is bounded, while the global history net has an infinite state space just because it has an unbounded history. Still, the behavior

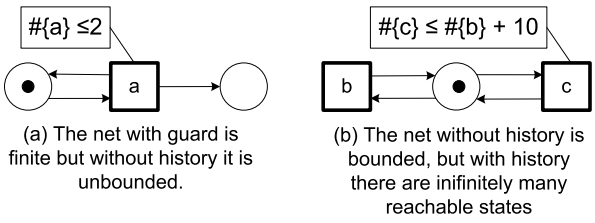

Figure 6. Bounded and unbounded nets

of this net is clearly analyzable, since it is possible to construct a classical Petri net bisimilar to it. The latter observation motivates why we are interested in the existence of a classical Petri net bisimilar to the global history net.

In the two following subsections we discuss sufficient conditions for the existence of a bisimilar Petri net (without history).

6.2.1. Guards depending on the marking only

In this subsection we give conditions on the guards that allow for a transformation into an equivalent bounded Petri net. So global history nets satisfying these conditions will accept regular languages. We consider here guards that depend only on the marking. As stated by Proposition 5.2 if transitions have unique labels, then a marking is uniquely determined by the history.

Definition 6.1. Given a global history net $N = \langle P, T, F, \Lambda, g\rangle$ with Λ being injective, we say that a formula φ is a *marking formula* if there exists a formula ψ, $\varphi \equiv \psi$, such that ψ is a counting formulas and every basic counting formula in ψ is of the form $(\#^\bullet p)\ \rho\ (\#p^\bullet + k)$ or $(\#^\bullet p + k)\ \rho\ (\#p^\bullet)$, for $p \in P$, $k \in \mathbb{N}$ and $\rho \in \{\leq, \geq\}$.

Theorem 6.2. Let $N = \langle P, T, F, \Lambda, g\rangle$ be a global history net with injective Λ such that for any $t \in T$, $g(t)$ is a marking formula. If the underlying Petri net $(\langle P, T, F, \Lambda\rangle, m_0)$ is bounded, then there exists a bounded marked Petri net bisimilar to $(N, (m_0, \epsilon))$.

Proof:
We construct a $N'' = \langle P', T', F'', \Lambda\rangle$ and a marking m_0'' such that (N'', m_0'') bisimilar to (N, m_0). We start by adding a duplicate place p' for every place $p \in P$ such that ${}^\bullet p' = p^\bullet$ and $p'^\bullet = {}^\bullet p$. Since the underlying Petri net is bounded, there exists $b \in \mathbb{N}$ such that for any reachable marking m and any place p, $m(p) \leq b$. We take n larger than the sum of b and the maximum of all constants in the guards. We define m_0' for N' as follows: $\forall p \in P : m_0'(p) = m_0(p) \wedge m_0'(p') = n - m_0(p)$. Observe that $m(p) + m(p') = n$ for any reachable marking m. Moreover, by construction, $\#^\bullet p = \#p'^\bullet$ and $\#p^\bullet = \#^\bullet p'$ for any place p.

Without loss of generality we assume that transition guards are conjunctions of the form $(\#^\bullet p)\ \rho\ (\#p^\bullet + k)$ with $k \geq 0$ and $\rho \in \{\leq, \geq\}$. Indeed, first, proof of Theorem 6.1 shows how general counting formulas can be reduced to basic counting formula. Second, if the guard is of the form $(\#^\bullet p + k)\ \rho\ \#p^\bullet$, by the previous observation, we obtain $(\#p'^\bullet + k)\ \rho\ \#^\bullet p'$, i.e., $(\#^\bullet p')\ \rho'\ (\#p'^\bullet + k)$ with ρ' being the

comparison dual to ρ, i.e. $\rho' \in \{\leq, \geq\} \setminus \{\rho\}$. We denote the resulting net $N' = \langle P', T', F', \Lambda \rangle$. Next we are going to add arcs depending on the guards of N.

We distinguish between two cases. Let $g(t)$ be $(\#{}^\bullet p) \leq (\#p^\bullet + k)$. Then t may fire only if the number of tokens consumed from p does not exceed the number of tokens produced to p by more than k, i.e., the number of tokens produced to p' does not exceed the number of tokens consumed from p' by more than k. In other words, $m'_0(p')$ has at least k tokens. Moreover, if $t \in {}^\bullet p'$ then t may fire only if p' contains at least $F'(p,t)$ tokens. Therefore, we add an arc between p' and t: $F''(p',t) = \max\{F'(p',t), m'_0(p') - k\}$, i.e., $\max\{F(t,p), n - k - m_0(p)\}$. To complete the transformation, observe that we are not allowed to change the behavior of the original net. Thus, we need to *return* tokens to p'. To this end we add an arc between t and p': $F''(t,p') = F'(t,p') + max\{0, m'_0(p) - k - F'(p',t)\}$, i.e., $F(p,t) + max\{0, n - k - m_0(p) - F(t,p)\}$.

Observe that this case also covers the situation when $g(t)$ is $(\#{}^\bullet p) \geq (\#p^\bullet + k)$ and $k = 0$. Therefore, we assume in the second case ($(\#{}^\bullet p) \geq (\#p^\bullet + k)$) that $k > 0$. Similarly to the previous case, we add two arcs between p and t: $F''(p,t) = \max\{F'(p,t), k + m'_0(p)\}$, i.e., $\max\{F(p,t), k + m_0(p)\}$, and $F''(t,p) = F'(t,p) + \max\{0, k + m'_0(p) - F'(p,t)\}$, i.e., $F(t,p) + \max\{0, k + m_0(p) - F(p,t)\}$.

In both cases t can fire if and only if the guard holds and the firing does not change the behavior of the original net. □

6.2.2. Counting formulas with bounded synchronization distance

In this subsection we consider a condition on guards that allows to transform a global history net to a bisimilar Petri net, which is not necessarily bounded. We use here an important concept in Petri nets introduced by Carl Adam Petri: *synchronization distance* [9, 12, 22]. We use a generalization of this notion, the so-called *y-distance* [25].

Definition 6.2. Let (N, m_0) be a Petri net and n be the number of transitions in N. For a *weight vector* $y \in \mathbb{Z}^n$ the *y-distance* of (N, m_0) is defined by

$$D((N, m_0), y) = \sup_{\sigma \in \Delta} y^T \cdot \overline{\sigma},$$

where y^T is the transpose of y, $\overline{\sigma}$ is the Parikh vector of σ and Δ the set of all executable finite firing sequences. The *synchronization set* is

$$\textit{Sync}((N, m_0)) = \{y \in \mathbb{Z}^n \mid D((N, m_0), y) < \infty\}.$$

In the right-hand net of Figure 6 transitions b and c can fire infinitely often. If we take the underlying classical Petri net, the transitions are completely independent of each other and the y-distance is ∞ for any weight vector with at least one positive component. If we consider the global history net instead, the number of the firings of c never exceeds the number of the firings of b by more then 11. Hence the y-distance with $y = \langle -1, 1 \rangle$ is 11. On the other hand, the number of firings of b is not restricted by the number of the firings of c, and the y-distance for $y = \langle 1, -1 \rangle$ is ∞.

For two label sets A and B, the *characteristic weight vector for (A,B)*, denoted $y_{(A,B)}$, is the weight vector with components equal to 1 for transitions with labels in A, -1 for transitions with labels in B and 0 for all other vector components (recall that we may safely assume that A and B are disjoint). We denote the $y_{(A,B)}$-distance by $d(A, B)$ and we call it the *characteristic distance (A,B)*. In [25], an

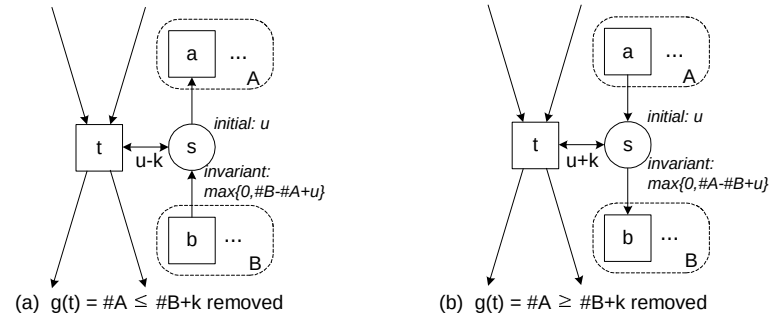

Figure 7. Transforming nets with synchronization distance restrictions

algorithm is given to decide whether $y \in Sync((n, m_0))$ and to determine the y-distance by examining a finite set of vectors.

Theorem 6.3. Let $N = \langle P, T, F, \Lambda, g\rangle$ be a global history net with initial marking m_0 such that for any $t \in T$, $g(t)$ is a disjunction of conjunctions of counting formulas of the form $\#A\ \rho\ \#B + k$ with $\rho \in \{<, \leq, >, \geq\}$, for each of which the following property holds: if ρ is $\leq$ then $d(A, B) < \infty$ and if ρ is $\geq$ then $d(B, A) < \infty$ in the underlying Petri net $(\langle P, T, F, \Lambda\rangle, m_0)$. Then there exists a marked Petri net (N', m'_0) bisimilar to (N, m_0).

Proof:
The proof is done by construction. Disjunctions and conjunctions are taken care of as in Theorem 6.1. Therefore, we restrict our attention to the following special case: the guard of transition t is a basic counting formula of the form $\#A\ \rho\ \#B + k$.

For the first case, where ρ is $\leq$, we set $u = \max\{k, d(A, B)\} + 1$. Note that $u \leq k$ implies that the guard of t will always be evaluated to $true$, and thus may be trivially removed. So we assume that $u > k$. We apply the construction shown at the left-hand side of Figure 7. A new place s is added with $F(b, s) = 1$ for all b such that $\Lambda(b) \in B$, $F(s, a) = 1$ for all a such that $\Lambda(a) \in A$, and $F(s, t) = F(t, s) = u - k$. Furthermore, the initial marking is $m'_0(s) = u$. Transition t can fire if and only if s contains at least $u - k$ tokens. Note that $u - k > 0$ and that for any reachable state (m, H) we have $m'(s) = u + \#B - \#A \geq u - d(A, B) > 0$. Therefore t can fire only if $\#B\ -\ \#A \geq -k$ and the transitions with labels in A or B are thus not restricted in their firings.

The second case, displayed in the right-hand net of Figure 7, is similar: $u = \max\{k, d(B, A)\} + 1$, the arcs are reversed, $F(s, t) = F(t, s) = u + k$ and $m'_0(s) = u$. □

Applying Theorem 6.3 to the global history net one the left-hand side of Figure 2, one obtains the classical net on the right-hand side of the figure (we took $R = 1$ and $L = 2$).

7. Global History Nets with Temporal Logic Formulas as Guards

In this section we take a complementary approach: while in the previous section we considered counting formulas, i.e., excluded ordering of the events $\prec$, here we consider temporal formulas, i.e., restrict

counting to express $\exists$ and $\forall$. Hence, we allow the guards to be first order logic formulas over a finite linearly-ordered set of events. It is well-known that the first order logics on linear traces coincides with LTL+Past [19, 7], where LTL+Past formulas are defined by $\phi ::= \mathit{false} \mid \phi \Rightarrow \phi \mid \phi \ \ \mathit{XU}\ \phi \mid \phi \ \ \mathit{YS}\ \phi \mid A$, $A \subseteq \Sigma$, XU is the temporal operator *neXt-Until*, and YS is the temporal operator *Yesterday-Since* [6, 7]. Semantics of the temporal operators can be expressed in terms of $\mathcal{CF}_\Sigma$. Let N be a global history net, $\langle E, \prec, \lambda\rangle \in \mathcal{H}_N$ and $e \in E$. We define $A(e)$ as $(\lambda(e) \in A)$, $(\phi \ \ \mathit{XU}\ \xi)(e)$ as $\exists e' : ((e \prec e') \wedge \xi(e') \wedge \forall e'' : ((e \prec e'') \wedge (e'' \prec e')) \Rightarrow \phi(e''))$ and $(\phi \ \ \mathit{YS}\ \xi)(e)$ as $\exists e' : ((e' \prec e) \wedge \xi(e') \wedge \forall e'' : ((e' \prec e'') \wedge (e'' \prec e)) \Rightarrow \phi(e''))$. We call a formula involving XU but not YS a *pure future formula*, a formula involving YS but not XU is a *pure past formula* and a formula not involving temporal operators a *pure present formula*.

Traditional temporal operators X ("next"), Y ("yesterday"), U("until"), S ("since"), $\Diamond$ ("eventually"), $\boxminus$ ("once"), $\boxminus$ ("always in the past") and $\Box$ ("always") can be defined based on XU and YS in a usual way, e.g., $X\phi := \mathit{false} \ \ \mathit{XU}\ \phi$, $\phi\ U\psi := \psi \vee (\phi \wedge (\phi \ \ \mathit{XU}\ \psi))$, $\Diamond\phi := \mathit{true}\ U\phi$, $\Box\phi := \neg(\Diamond\neg\phi)$. Using this notation we can express properties such as "every request submitted in the past was eventually granted" $\boxminus(\mathit{request} \Rightarrow \Diamond \mathit{grant})$ and "every future grant should be preceded by a request" $\Box(\mathit{grant} \Rightarrow \boxminus \mathit{request})$.

While standard LTL+Past works on infinite traces, our history is always finite. Therefore, we interpret formulas on a trace we observed so far. We evaluate the formulas with respect to the *most recent* event of a given history. Recall that in this section we restrict our attention to histories produced by firings of global history nets, which are linearly-ordered finite sets of labeled events, the *most recent* event of a given history is always unique.

The following lemma states that for finite traces one can restrict attention to boolean combinations of pure past and pure present formulas. For the sake of brevity we call such combinations *no-future* formulas.

Lemma 7.1. Let ϕ be an LTL+Past formula. Then, there exists an LTL+Past formula ψ such that: (1) ψ is a no-future formula, and (2) for any global history net N, and for any $H \in \mathcal{H}_N$, $H \models \phi$ if and only if $H \models \psi$.

Proof:
By the *separation theorem* [6, 7, 8] there exists a boolean combination of pure past formulas, pure future formulas and pure present formulas ξ equivalent to ϕ, i.e., evaluated to the same value on all infinite traces. Since we restrict our attention to finite traces and evaluate the formulas on the most recent event of the given history, the pure future formulas participating in ξ are evaluated to false. Therefore, we take ξ with all pure future formulas replaced by false as ψ. □

To illustrate the separation theorem consider the following example.

Example 7.1. Let ϕ be $\boxminus(\mathit{request} \Rightarrow \Diamond \mathit{grant})$. First of all, we replace the temporal operators used by XU and YS:

$$((\mathit{request} \Rightarrow \mathit{grant}) \vee (\mathit{true}\ \mathit{XU}\ \mathit{grant})) \vee \neg(\mathit{true}\ \mathit{YS}\ (\neg(\mathit{request} \Rightarrow \mathit{grant}) \wedge \neg(\mathit{true}\ \mathit{XU}\ \mathit{grant})))$$

The "only part" of the formula that has to be rewritten is $(\mathit{true}\ \mathit{YS}\ \neg(\mathit{request} \Rightarrow \mathit{grant} \vee (\mathit{true}\ \mathit{XU}\ \mathit{grant}))$.

Applying the rewrite rules of [7] to $true$ *YS* $\neg$(*request* $\Rightarrow$ *grant* $\vee$ ($true$ *XU grant*)) obtain

$$\begin{array}{c}(((true \wedge \neg \textit{grant})\ \textit{YS}\ \neg(\textit{request} \Rightarrow \textit{grant})) \wedge \neg \textit{grant} \wedge \neg(true\ \textit{XU}\ \textit{grant}))\\ \vee\\ (\neg \textit{grant} \wedge \neg true \wedge ((\neg \textit{grant} \wedge true)\ \textit{YS}\ \neg(\textit{request} \Rightarrow \textit{grant})))\\ \vee\\ (true\ \textit{YS}\ (\neg \textit{grant} \wedge \neg true \wedge true \wedge ((\neg \textit{grant} \wedge true)\ \textit{YS}\ \neg(\textit{request} \Rightarrow \textit{grant}))))\end{array}$$

The second and the third disjuncts are $false$. Hence ϕ is equivalent to

$$\begin{array}{l}((\textit{request} \Rightarrow \textit{grant}) \vee (true\ \textit{XU}\ \textit{grant}))\\ \quad \vee \neg((\neg \textit{grant}\ \textit{YS}\ \neg(\textit{request} \Rightarrow \textit{grant})) \wedge \neg \textit{grant} \wedge \neg(true\ \textit{XU}\ \textit{grant})).\end{array} \quad (1)$$

As explained in the proof of Lemma 7.1 we replace pure future formulas of (1) by $false$. Hence, the following formula is obtained:

$$(\textit{request} \Rightarrow \textit{grant}) \vee \neg((\neg \textit{grant}\ \textit{YS}\ \neg(\textit{request} \Rightarrow \textit{grant})) \wedge \neg \textit{grant}). \quad (2)$$

We can further simplify formula (2) using the following observation. While standard LTL+Past assumes atomic properties, our atomic properties are occurrences of events. Hence, no two events can occur at the same time in a history obtained by firings of a global history net. This means that formulas can be further simplified: $a \wedge b$ can be replaced by $false$, $a \Rightarrow b$ by b, etc. Hence, (2) can be simplified to $\neg$*request* $\vee$ $\neg$(($\neg$*grant* *YS request*) $\wedge$ $\neg$*grant*), i.e., to

$$\neg \textit{request} \vee \neg(\neg \textit{grant}\ \textit{YS}\ \textit{request}) \vee \textit{grant}. \quad (3)$$

7.1. Global history nets with no-future guards

In this subsection we discuss a translation for global history nets with no-future formulas as guards. In order to translate such a global history net to a classical Petri net, we first translate it to a colored Petri nets over a finite set of colors, which can be translated to a classical Petri net [18].

The intuition behind the translation is that one does not need an unbounded number of values to compute the value of a formula. For instance, the value of $\boxminus a$ on a history $H = \langle \{e_1 \dots e_n\}, \prec, \lambda \rangle$ with $e_i \prec e_j$ if and only if $i < j$, can be computed from the value of $\boxminus a$ on $H' = \langle \{e_1 \dots e_{n-1}\}, \prec, \lambda \rangle$ and $a(e_n)$, i.e., $H \models \boxminus a$ can be determined as $(H' \models \boxminus a) \wedge a(e_n)$. This means that in order to compute $\boxminus a$ one should have one extra place with a token recording the value of $\boxminus a$ on H'. The next lemma states that the value of any formula can be computed by using finitely many of such places.

To state the lemma we need an auxiliary notion of the set of subformulas of a given formula. Given a formula ϕ we define the set of subformulas *sub*(ϕ) as follows: *sub*(A) = $\{A\}$, *sub*($false$) = $\{false\}$, *sub*($Y\psi$) = $\{Y\psi\} \cup$ *sub*(ψ), *sub*($\psi\ S\xi$) = $\{\psi\ S\xi\} \cup$ *sub*(ψ) $\cup$ *sub*(ξ) and *sub*($\psi \Rightarrow \xi$) = $\{\psi \Rightarrow \xi\} \cup$ *sub*(ψ) $\cup$ *sub*(ξ).

Lemma 7.2. Let ϕ be an no-future formula. Then, there exist functions *init* : *sub*(ϕ) $\rightarrow \{false, true\}$ and a function *update* : $\Sigma \times$ (*sub*(ϕ) $\rightarrow \{false, true\}$) $\rightarrow$ (*sub*(ϕ) $\rightarrow \{false, true\}$) such that $\epsilon \models \alpha$ evaluates to *init*(α), and $H \models \alpha$ evaluates to *update*($\lambda(e), (H' \models \phi)$)($\alpha$) for any $\alpha \in$ *sub*(ϕ), $H = \langle E, \prec, \lambda \rangle$, where $e \in E$ is the most recent event of H, i.e., $\forall e'((e' \in E) \Rightarrow (e' \preceq e))$, $H' = \langle E \setminus \{e\}, \prec'$ $, \lambda' \rangle$, $\prec' = \prec \setminus \{(e', e) \mid e' \in E\}$ and λ' restricts λ to $E \setminus \{e\}$.

Proof:
We prove the lemma by induction on the structure of ϕ. At each step we construct *init* and *update* assuming that the corresponding functions are available for $\mathit{sub}(\phi) \setminus \phi$.

If ϕ is A then $\mathit{init}(A)$ is false, and $\mathit{update}(\ell, v)(A)$ is true if $\ell \in A$, and false, otherwise. If ϕ is false, we take $\mathit{init}(\mathit{false}) = \mathit{false}$ and $\mathit{update}(\ell, v)(\mathit{false})$ to be false. If ϕ is $\psi \Rightarrow \xi$ we define $\mathit{init}(\psi \Rightarrow \xi)$ as $\mathit{init}(\psi) \Rightarrow \mathit{init}(\xi)$, and $\mathit{update}(\ell, v)(\psi \Rightarrow \xi)$ as $\mathit{update}(\ell, v)(\psi) \Rightarrow \mathit{update}(\ell, v)(\xi)$. Observe that the *update* functions are defined since $\psi, \xi \in \mathit{sub}(\phi)$.

For *Yesterday-Since* we first observe $\psi \ \ \mathit{YS}\ \xi$ evaluates to false on the empty history. Hence, $\mathit{init}(\psi \ \ \mathit{YS}\ \xi) = \mathit{false}$. To define *update* observe $(\psi \ \ \mathit{YS}\ \xi)(e) = \xi(e') \vee (\psi(e') \wedge (\psi \ \ \mathit{YS}\ \xi)(e'))$, where e' is the event preceding e (i.e., $e' \prec e$ and for any e'' differing from e', $e'' \prec e$ implies $e'' \prec e'$). Therefore, $\mathit{update}(\ell, v)(\psi \ \ \mathit{YS}\ \xi) = v(\xi) \vee (v(\psi) \wedge v(\psi \ \ \mathit{YS}\ \xi))$. □

To illustrate the construction proposed above consider the following example:

Example 7.2. We proceed with the running example and define *init* and *update* for (3). For the sake of brevity we denote $\neg\mathit{request} \vee \neg(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request}) \vee \mathit{grant}$ by ψ:

$$
\begin{aligned}
\mathit{init}(\mathit{request}) &= \mathit{false}\\
\mathit{update}(\ell, v)(\mathit{request}) &= \ell \in \{\mathit{request}\}\\
\mathit{init}(\mathit{grant}) &= \mathit{false}\\
\mathit{update}(\ell, v)(\mathit{grant}) &= \ell \in \{\mathit{grant}\}\\
\mathit{init}(\neg\mathit{grant}) &= \mathit{true}\\
\mathit{update}(\ell, v)(\neg\mathit{grant}) &= \neg(\mathit{update}(\ell, v)(\mathit{grant}))\\
\mathit{init}(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request}) &= \mathit{false}\\
\mathit{update}(\ell, v)(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request}) &= v(\mathit{request}) \vee (v(\neg\mathit{grant}) \wedge v(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request}))\\
\mathit{init}(\neg(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request})) &= \mathit{true}\\
\mathit{update}(\ell, v)(\neg(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request})) &= \neg(\mathit{update}(\ell, v)(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request}))\\
\mathit{init}(\neg\mathit{request}) &= \mathit{true}\\
\mathit{update}(\ell, v)(\neg\mathit{request}) &= \neg(\mathit{update}(\ell, v)(\mathit{request}))\\
\mathit{init}(\psi) &= \mathit{true}\\
\mathit{update}(\ell, v)(\psi) &= \mathit{update}(\ell, v)(\neg\mathit{request}) \vee\\
&\qquad \mathit{update}(\ell, v)(\neg(\neg\mathit{grant}\ \mathit{YS}\ \mathit{request})) \vee\\
&\qquad\quad \mathit{update}(\ell, v)(\mathit{grant})
\end{aligned}
$$

Corollary 7.1. Every no-future formula can be represented by finitely many *init* and *update* functions.

Proof:
Every no-future formula can be represented as suggested in Lemma 7.2. Observe that for every $\alpha \in \mathit{sub}(\phi)$ we defined two functions, *init* and *update*. Since $\mathit{sub}(\phi)$ is finite, every ϕ is therefore represented by finitely many functions. □

Let $N = \langle P, T, F, \lambda, g\rangle$ be a global history net, $N' = \langle P', T, F', \lambda, g\rangle$ be the log extension of N (cf. Definition 5.2) and $k = \sum_{t\in T} |\mathit{sub}(g(t))|$. Then, we evaluate N' as a colored Petri net over the following color set: $\{\bullet\} \cup \{false, true\}^k$, i.e., black tokens and boolean vectors of k dimensions. Intuitively, the token at p^* records the values of the functions needed to evaluate the guards, while other places may contain only black tokens ($\bullet$). Construction proposed in Lemma 7.2 implies that the value of the token at p^* can be initialized using *init* and updated using *update*.

Formally, we redefine the semantics of a log extension as follows:

Definition 7.1. Let $N' = \langle P \cup \{p^*\}, T, F', \lambda, g\rangle$ be the log extension of a global history net $N = \langle P, T, F, \lambda, g\rangle$. $Color$ is the set $\{\bullet\} \cup \{false, true\}^k$, for $k = \sum_{t\in T} |\mathit{sub}(g(t))|$. A *state* m of N' is a bag of colored tokens, such that all tokens residing on places from P are black, and tokens at p^* are boolean vectors, i.e., a marking $m : (P \times \{\bullet\} \cup \{p^*\} \times \{false, true\}^k) \to \mathbb{N}$.

The *transition relation* is specified by: $m \overset{a}{\dashrightarrow} m'$ if and only if there exist a transition t with $\lambda(t) = a$, and two bags $cons$ and $prod$ of black tokens such that:

- $v(g(t))$ is $true$, where boolean vector v seen as a function is such that $m(p^*, v) > 0$;
- $cons(p) \leq m(p)$ for any $p \in P$;
- $\sum_{(p,\bullet)\in cons}[p] = {}^\bullet t$;
- $prod = \sum_{p\in t^\bullet}[(p, \bullet)]$;
- $m' = m - cons - (p^*, v) + prod + (p^*, \mathit{update}(a, v))$, where *update* is defined in Lemma 7.2.

Furthermore, for a given marking m_0 of $N = \langle P, T, F, \Lambda, g\rangle$ we define the corresponding marking m'_0 of the log extension N' of N as follows: $m'_0((p, \bullet)) = m_0((p, \bullet))$ for all $p \in P$ and $m'_0((p^*, \mathit{init})) = 1$ where $\mathit{init} : \bigcup_{t\in T} \mathit{sub}(g(t)) \to \{false, true\}$ is defined in Lemma 7.2.

Theorem 7.1. Let $(N, (m_0, \epsilon))$ with $N = \langle P, T, F, \Lambda, g\rangle$ be a marked global history net, and (N', m'_0) be such that $N' = \langle P', T, F', \Lambda, g\rangle$ is the log extension of N and m'_0 is as above. Then, $(\mathbb{N}^P, \Lambda(T), \longrightarrow, m_0)$ and $(\mathbb{N}^{P'}, \Lambda(T), \dashrightarrow, m'_0)$ are bisimilar.

Proof:
Follows from Lemma 7.2. □

Since N' is evaluated over the finite set of colors it can either be translated into a classical Petri net [18] or the corresponding state space can be computed explicitly.

8. Related Work

Histories and related notions such as event systems [28] and pomsets [13, 5] have been used in the past to provide causality-preserving semantics for Petri nets. Unlike our approach, these works did not aim at restricting the firings by means of history-dependent guards. Baldan *et al.* [4] use two different notions of history. First of all, they consider *semi-weighted nets*, i.e., nets where every token can be uniquely identified by means of tokens used to produce it, transition that produces it and the name of the place

where it resides. This idea is similar in spirit to our token history. However, the authors do not make this notion of history explicit nor do they discuss additional operations that can be performed on histories. Neither this notion, nor history as configuration used by the authors in study of causality, can be used to restrict firings of transitions by means of guards as suggested in our approach.

History-dependent automata [21] extend states and transitions of an automaton with sets of local names: each transition can refer to the names associated to its source state but can also generate new names which can then appear in the destination state. This notion of history implies that one cannot refer to firings of other transitions but by means of shared names. We believe that the ability to express dependencies on previous firings explicitly is the principal advantage of our approach.

Operations on pomsets similar to our union and intersection appeared under different names in [11, 23, 27]. The major distinction is due to unimportance of the events' identities in these approaches. Therefore, these operations make use of disjoint sum to define a union and bijectively rename the events to define an intersection. Therefore, these operations are defined for any pomsets. Unlike the existing approaches, we take the identities of the events into account. This guarantees that common parts of histories appear only once in their union, and only truly common events appear in the intersection.

y-distance and related notions were studied starting from [9, 12, 22, 25]. Silva and Murata [24] introduced group-B-fairness, where they extend the synchronization distance notion from single transitions to the groups of transitions, like we do in Subsection 6.2.2. The focus of Silva and Murata's paper is however on *group-B-fair nets*, i.e., nets such that any pair of transition sets from a given transition covering is in a group-B-fair relation. Unlike their work, Theorem 6.3 demands being in a group-B-fair relation only for sets of transitions corresponding to sets of labels used in the guards.

Our transformation of no-future LTL+Past formulas discussed in Section 7.1 is related to verification of LTL formulas on finite traces [10, 14], however, none of these approaches considered *both* future and past temporal operators. Moreover, while Giannakopoulou and Havelund [10] exclude the "next" operator, we do include the corresponding past counterpart. A subsequent paper by Havelund and Ruşu [15] generates an efficient dynamic programming algorithm from a Past LTL formula. However, the semantics of the operator "yesterday" considered by the authors differs from the one used above. For a one-element trace s they interpret $s \models Y\phi$ as $true$ if and only if $s \models \phi$ evaluates to $true$, i.e., a trace consisting of exactly one state is considered like a stationary infinite trace containing only this state. We believe that assuming $s \models Y\phi$ to be $false$ for a one-event history s is more natural in the context of global history nets.

9. Conclusion

In this paper we emphasize the importance of taking history into account while modelling processes. Historical information is present in most state-of-the-art enterprise information systems. Moreover, it allows to separate process information from safety constraints, improving the readability and maintainability of models.

We have provided means to model history-dependent processes by extending the classical Petri nets model and considered two ways of incorporating history: token history nets and global history nets. To provide analysis, simulation and validation facilities, we have put a link from global history nets to classical and inhibitor Petri nets. Namely, we have identified several subclasses of global history nets that can be automatically transformed to classical Petri nets. For the class of global history nets with

counting formulas as guards we have defined a transformation to inhibitor nets. Finally, observe that global history nets are readily implemented in CPN Tools [1].

Future work For the future work we plan to adapt our token net framework for modelling component-based systems. We intend to extend the language of operations on histories by adding projection in order to allow information hiding and intersection to check disjointness/presence of common parts in token histories. The guard language will allow to evaluate conditions both on separate tokens and on their combinations.

We are going to develop a method for transforming broader subclasses of global history nets to classical and inhibitor Petri nets, e.g., we would like to consider global history nets with guards being formulas from temporal logics other than Past+LTL such as FTL [3] and LogLogics [16].

Acknowledgement We are grateful to Jan Hidders and Jan Paredaens for a number of fruitful discussions at the early stages of this research.

References

[1] CPN Tools. `http://wiki.daimi.au.dk/cpntools/cpntools.wiki`.

[2] *16th IEEE International Conference on Automated Software Engineering (ASE 2001), 26-29 November 2001, Coronado Island, San Diego, CA, USA*. IEEE Computer Society, 2001.

[3] R. Armoni, L. Fix, A. Flaisher, R. Gerth, B. Ginsburg, T. Kanza, A. Landver, S. Mador-Haim, E. Singerman, A. Tiemeyer, M. Y. Vardi, and Y. Zbar. The forspec temporal logic: A new temporal property-specification language. In Katoen and Stevens [20], pages 296–211.

[4] P. Baldan, N. Busi, A. Corradini, and G. M. Pinna. Domain and event structure semantics for Petri nets with read and inhibitor arcs. *Theoretical Computer Science*, 323(1-3):129–189, 2004.

[5] E. Best and R. R. Devillers. Sequential and concurrent behaviour in Petri net theory. *Theoretical Computer Science*, 55(1):87–136, 1987.

[6] V. Diekert and P. Gastin. First-order definable languages. In J. Flum, E. Grädel, and T. Wilke, editors, *Logic and Automata: History and Perspective*, Texts in Logic and Games, pages 261–306. Amsterdam University Press, 2008.

[7] D. M. Gabbay. The declarative past and imperative future: Executable temporal logic for interactive systems. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, volume 398 of *Lecture Notes in Computer Science*, pages 409–448. Springer, 1987.

[8] D. M. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal basis of fairness. In *POPL*, pages 163–173, 1980.

[9] H. J. Genrich, K. Lautenbach, and P. S. Thiagarajan. Elements of general net theory. In *Proceedings of the Advanced Course on General Net Theory of Processes and Systems*, pages 21–163, London, UK, 1980. Springer-Verlag.

[10] D. Giannakopoulou and K. Havelund. Automata-based verification of temporal properties on running programs. In *ASE* [2], pages 412–416. Full version available as a technical report.

[11] J. L. Gischer. The equational theory of pomsets. *Theoretical Computer Science*, 61:199–224, 1988.

[12] U. Goltz and W. Reisig. Weighted Synchronic Distances. In C. Girault and W. Reisig, editors, *Selected Papers from the First and the Second European Workshop on Application and Theory of Petri Nets*, volume 52 of *Informatik-Fachberichte*, pages 289–300. Springer Verlag, 1981.

[13] U. Goltz and W. Reisig. The non-sequential behavior of Petri nets. *Information and Control*, 57(2/3):125–147, 1983.

[14] K. Havelund and G. Roşu. Monitoring programs using rewriting. In *ASE* [2], pages 135–143.

[15] K. Havelund and G. Roşu. Synthesizing monitors for safety properties. In Katoen and Stevens [20], pages 342–356.

[16] K. van Hee, O. Oanea, A. Serebrenik, N. Sidorova, and M. Voorhoeve. LogLogics: A logic for history-dependent business processes. *Sci. Comput. Program.*, 65(1):30–40, 2007.

[17] K. van Hee, A. Serebrenik, N. Sidorova, and W. M. P. van der Aalst. History-Dependent Petri Nets. In J. Kleijn and A. Yakovlev, editors, *ICATPN*, volume 4546 of *Lecture Notes in Computer Science*, pages 164–183. Springer, 2007.

[18] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*. Monographs in Theoretical Computer Science. Springer-Verlag, 1997.

[19] J. A. W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD in Philosophy, University of California at Los Angeles, 1968.

[20] J.-P. Katoen and P. Stevens, editors. *Tools and Algorithms for the Construction and Analysis of Systems, 8th International Conference, TACAS 2002, Held as Part of the Joint European Conference on Theory and Practice of Software, ETAPS 2002, Grenoble, France, April 8-12, 2002, Proceedings*, volume 2280 of *Lecture Notes in Computer Science*. Springer, 2002.

[21] U. Montanari and M. Pistore. History-dependent automata: An introduction. In M. Bernardo and A. Bogliolo, editors, *SFM*, volume 3465 of *Lecture Notes in Computer Science*, pages 1–28. Springer, 2005.

[22] C. A. Petri. Interpretations of net theory. Technical Report ISF-Report 75.07, 1975.

[23] V. R. Pratt. Some constructions for order-theoretic models of concurrency. In R. Parikh, editor, *Logic of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 269–283. Springer, 1985.

[24] M. Silva and T. Murata. B-fairness and structural B-fairness in Petri net models of concurrent systems. *J. Comput. Syst. Sci.*, 44(3):447–477, 1992.

[25] I. Suzuki and T. Kasami. Three measures for synchronic dependence in Petri nets. *Acta Inf.*, 19:325–338, 1983.

[26] R. Valk. On the computational power of extended Petri nets. In J. Winkowski, editor, *MFCS*, volume 64 of *Lecture Notes in Computer Science*, pages 526–535. Springer, 1978.

[27] H. Wimmel and L. Priese. Algebraic characterization of Petri net pomset semantics. In A. W. Mazurkiewicz and J. Winkowski, editors, *CONCUR*, volume 1243 of *Lecture Notes in Computer Science*, pages 406–420. Springer, 1997.

[28] G. Winskel. Event structures. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances in Petri Nets*, volume 255 of *Lecture Notes in Computer Science*, pages 325–392. Springer, 1986.