6-2012

Complexity of the soundness problem of bounded workflow nets

Guan Jun LIU

Jun SUN
Singapore Management University, junsun@smu.edu.sg

Yang LIU

Jin Song DONG

Citation

LIU, Guan Jun; SUN, Jun; LIU, Yang; and DONG, Jin Song. Complexity of the soundness problem of bounded workflow nets. (2012). *Proceedings of the 33rd International Conference, PETRI NETS 2012, Hamburg, Germany, June 25-29*. 92-107.
Available at: https://ink.library.smu.edu.sg/sis_research/5013

Complexity of the Soundness Problem of Bounded Workflow Nets

Guan Jun Liu[1], Jun Sun[1], Yang Liu[2], and Jin Song Dong[3]

[1] ISTD, Singapore University of Technology and Design, Singapore 138682
{guanjun_liu,sunjun}@sutd.edu.sg
[2] Temasek Lab, National University of Singapore, Singapore 117417
tslliuya@nus.edu.sg
[3] School of Computing, National University of Singapore, Singapore 117417
dongjs@comp.nus.edu.sg

Abstract. Classical workflow nets (WF-nets) are an important class of Petri nets that are widely used to model and analyze workflow systems. Soundness is a crucial property that guarantees these systems are deadlock-free and bounded. Aalst *et al.* proved that the soundness problem is decidable, and proposed (but not proved) that the soundness problem is EXPSPACE-hard. In this paper, we show that the satisfiability problem of Boolean expression is polynomial time reducible to the liveness problem of bounded WF-nets, and soundness and liveness are equivalent for bounded WF-nets. As a result, the soundness problem of bounded WF-nets is co-NP-hard.

Workflow nets with reset arcs (reWF-nets) are an extension to WF-nets, which enhance the expressiveness of WF-nets. Aalst *et al.* proved that the soundness problem of reWF-nets is undecidable. In this paper, we show that for bounded reWF-nets, the soundness problem is decidable and equivalent to the liveness problem. Furthermore, a bounded reWF-net can be constructed in polynomial time for every linear bounded automaton (LBA) with an input string, and we prove that the LBA accepts the input string if and only if the constructed reWF-net is live. As a result, the soundness problem of bounded reWF-nets is PSPACE-hard.

Keywords: Petri nets, workflow nets, workflow nets with reset arcs, soundness, co-NP-hardness, PSPACE-hardness.

1 Introduction

In the recent decade, workflow nets (WF-nets) have been widely applied to (inter-organizational) workflow management systems and business process management systems to model and analyze their operational processes [1]-[4], [11]-[14], [16]. WF-nets can well characterize their system features such as concurrency, choices, and synchronous/asynchronous communication. To enhance the expressive power, some extensions to WF-nets, such as workflow nets with reset or inhibitor arcs (reWF-nets and inWF-nets, respectively) [17,18], are proposed which can express priority, preemption, or cancelation.

Soundness [1]-[4] is an important property of these systems which, informally speaking, reflects whether the designed systems are correct. For instance, the soundness of

WF-nets guarantees that the designed systems are deadlock-free and bounded. Aalst *et al.* [4] defined eight notions of soundness. This work considers the classical soundness [4] that means any run of the designed systems is always finished correctly and each action has a right/chance to be executed.

It has been proven that the soundness of WF-nets is decidable [3,11]. Aalst proposed (but not proved) in [3] that the soundness problem is EXPSPACE-hard, based on the work in [5], which shows that the problems of reachability, liveness, and deadlock of Petri nets are all EXPSPACE-hard. In addition, Aalst proved in [3] that the soundness problem of a workflow net can be decided in polynomial time if the workflow net is a free-choice one [6].

Generally, enhancing the models' expressive power increases the complexity of deciding their properties. Aalst *et al.* [4] proved that the soundness problems of reWF- and inWF-nets are both undecidable.

Our contribution. In this paper, we show that the satisfiability problem of Boolean expression (SAT problem) is polynomial time reducible to the liveness problem of bounded WF-nets, and the soundness and liveness are equivalent for bounded WF-nets, thereby proving that the soundness problem of bounded WF-nets is co-NP-hard. Based on the Linear Bounded Automaton Acceptance problem (LBA Acceptance problem), we show that the soundness problem of bounded reWF-nets is decidable but PSPACE-hard. To the best of our knowledge, it is the first time to propose and prove these conclusions for WF- and reWF-nets.

Organization. The remainder of the paper is organized as follows. Section 2 reviews the definitions of Petri nets, WF- and reWF-nets, and the SAT and LBA Acceptance problems. Section 3 proves the co-NP-hardness of the soundness of bounded WF-nets and Section 4 proves the PSPACE-hardness of the soundness of bounded reWF-nets. Section 5 concludes this paper.

2 Preliminary

In this section, we review the definitions of Petri nets, WF-nets, reWF-nets, SAT problem, and LBA Acceptance problem. For more details, please refer to [4] and [9].

2.1 Petri Nets

Let $\mathbb{N} = \{0,\ 1,\ 2, \cdots\}$ be the set of nonnegative integers, Given $m \in \mathbb{N}$ and $m > 0$, let $\mathbb{N}_m = \{1,\ 2, \cdots, m\}$ be the set of integers from 1 to m.

Definition 1. *A net is a 3-tuple $N = (P,\ T,\ F)$ where P is a set of places, T is a set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs, $P \cup T \neq \varnothing$, and $P \cap T = \varnothing$.*

A transition t is called an *input transition* of a place p and p is called an *output place* of a transition t if $(t,\ p) \in F$. *Input place* and *output transition* can be defined similarly. Given a net $N = (P,\ T,\ F)$ and a node $x \in P \cup T$, ${}^\bullet x = \{y \in P \cup T \mid (y,\ x) \in F\}$ and $x^\bullet = \{y \in P \cup T \mid (x,\ y) \in F\}$ are called the *pre-set* and *post-set* of x, respectively.

A *marking* of $N = (P,\ T,\ F)$ is a mapping M: $P \to \mathbb{N}$. $p \in P$ is *marked* at M if $M(p) > 0$. A marking may be viewed as a $\mid P \mid$-dimensional non-negative integer vector in which every element represents the number of tokens in corresponding place at this marking, e.g., $M = (1,\ 0,\ 6,\ 0)$ over $P = \{p_1,\ p_2,\ p_3,\ p_4\}$ represents that at M, p_1, p_2, p_3, and p_4 have 1, 0, 6, and 0 tokens, respectively. Notice, we assume a total order on the set of places P so that the i-th entry in the vector corresponds to the i-th place in the ordered set. When the number of places is very large and the distribution of tokens is sparse, the above two kinds of presentation of a marking are relatively complex. For convenience, M is denoted as $M = \sum_{p \in P} M(p) \cdot p$ in this paper. For the above example, it is written as $M = p_1 + 6p_3$.

If $\forall p \in {}^\bullet t$: $M(p) > 0$, then t is said to be *enabled* at M, which is denoted as $M[t\rangle$. Firing an enabled transition t produces a new marking M', which is denoted as $M[t\rangle M'$, such that $M'(p) = M(p) - 1$ if $p \in {}^\bullet t \setminus t^\bullet$; $M'(p) = M(p) + 1$ if $p \in t^\bullet \setminus {}^\bullet t$; and $M'(p) = M(p)$ otherwise.

A marking M_k is said to be reachable from a marking M if there exists a firing sequence $\sigma = t_1 t_2 \cdots t_k$ such that $M[t_1\rangle M_1[t_2\rangle \cdots\rangle M_{k-1}[t_k\rangle M_k$. $M[\sigma\rangle M_k$ represents that M reaches M_k after firing sequence σ. The set of all markings reachable from M in a net N is denoted as $R(N,\ M)$.

A net N with an *initial marking* M_0 is called a *Petri net*, and denoted as $(N,\ M_0)$.

Definition 2. *Given a Petri net* $(N,\ M_0) = (P,\ T,\ F,\ M_0)$, $t \in T$ *is called live if* $\forall M \in R(N,\ M_0)$, $\exists M' \in R(N,\ M)$: $M'[t\rangle$. *A Petri net* $(N,\ M_0)$ *is called live if every transition is live. It is called bounded if* $\forall p \in P$, $\exists k \in \mathbb{N}$, $\forall M \in R(N,\ M_0)$: $M(p) \leq k$.

2.2 WF-Nets

WF-nets are an important subclass of Petri nets and widely studied and applied in academic and industrial systems. Each WF-net has a source place representing the beginning of a task and a sink place representing the ending of the task.

Definition 3. *A net* $N = (P,\ T,\ F)$ *is a WF-net if*

1. N *has two special places* i *and* o *where* $i \in P$ *is called source place such that* ${}^\bullet i = \varnothing$ *and* $o \in P$ *is called sink place such that* $o^\bullet = \varnothing$; *and*
2. $N^E = (P,\ T \cup \{b\},\ F \cup \{(b,\ i),\ (o,\ b)\})$ *is strongly connected.*

For instance, Fig. 1(a) is a WF-net in which i and o are its source and sink places, respectively. This WF-net may be seen as a composition of three subsystems. The left and right subsystems produce parts and the middle assembles them.

Definition 4. *Let* $N = (P,\ T,\ F)$ *be a WF-net,* $M_0 = i$, *and* $M_d = o$. N *is sound if the following requirements hold:*

1. $\forall M \in R(N,\ M_0)$: $M_d \in R(N,\ M)$; *and*
2. $\forall t \in T$, $\exists M \in R(N,\ M_0)$: $M[t\rangle$.

Aalst [3] proves that the soundness is equivalent to the liveness and boundedness for WF-nets.

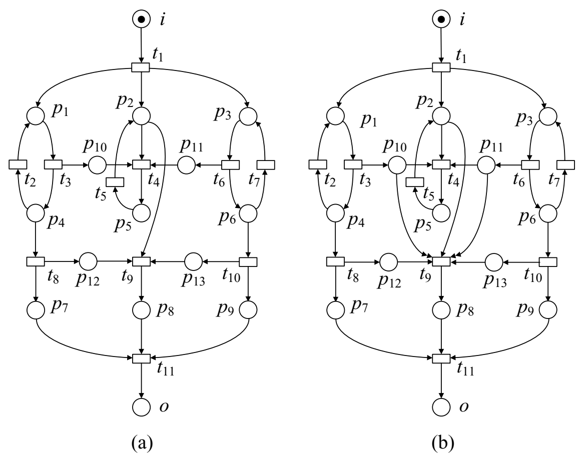

Fig. 1. (a) A WF-net; and (b) an reWF-net

Theorem 1. *Let $N = (P, T, F)$ be a WF-net, $N^E = (P, T \cup \{b\}, F \cup \{(b, i), (o, b)\})$, and $M_0 = i$. Then, N is sound if and only if (N^E, M_0) is live and bounded.*

Therefore, the following conclusion is obvious.

Corollary 1. *Let $N = (P, T, F)$ be a WF-net such that $(N^E, M_0) = (P, T \cup \{b\}, F \cup \{(b, i), (o, b)\}, i)$ is bounded. Then, N is sound if and only if (N^E, M_0) is live.*

2.3 reWF-Nets

reWF-nets are an extension to WF-nets in which some reset arcs are added. A reset arc can delete all tokens from related places and then the next computing can be started.

Definition 5. *A 4-tuple $N = (P, T, F, R)$ is an reWF-net if*

1. *$N = (P, T, F)$ is a WF-net; and*
2. *$R \subseteq [P \setminus \{o\} \times T]$ is the set of reset arcs.*

A reset arc is represented by a double-headed arrow in an reWF-net chart, and denoted as $[p, t]$ formally in order to differ from (p, t) which is the notation of an arc in general Petri nets. We denote $^\circ t = \{p \in P \mid [p, t] \in R\}$, $\forall t \in T$, as the set of places that connect with t by reset arcs. For example, Fig. 1(b) is an reWF-net that has two reset arcs $[p_{10}, t_9]$ and $[p_{11}, t_9]$. $^\circ t_9 = \{p_{10}, p_{11}\}$.

Rules of enabling and firing a transition are defined as follows:

Given an reWF-net $N = (P, T, F, R)$ and a marking M, if $\forall p \in {}^\bullet t$: $M(p) > 0$, then t is said to be *enabled* at M, which is denoted as $M[t\rangle$. Firing an enabled transition t produces a new marking M', which is denoted as $M[t\rangle M'$, such that $M'(p) = 0$ if $p \in {}^\circ t$; $M'(p) = M(p) - 1$ if $p \notin {}^\circ t \wedge p \in {}^\bullet t \setminus t^\bullet$; $M'(p) = M(p) + 1$ if $p \notin {}^\circ t \wedge p \in t^\bullet \setminus {}^\bullet t$; and $M'(p) = M(p)$ otherwise.

Obviously, the enabling rule of transitions of reWF-nets identifies with that of general Petri nets, but, after firing a transition, the tokens in those places that connect with the transition by reset arcs are all removed.

Other notions of reWF-nets, such as liveness, boundedness, and soundness, are the same as those of Petri nets and WF-nets, and are omitted here.

2.4 SAT Problem

The SAT problem, which is NP-complete [9], is used in this paper. Assume that there are n Boolean variables x_1, $x_2,\cdots$, and x_n. A literal l is a variable x or its negation $\neg x$. An expression G of conjunctive normal form (CNF) is a conjunction of m different terms and each term is a disjunction of different literals not containing a complementary pair x and $\neg x$. An expression H of disjunctive normal form (DNF) is a disjunction of m different terms and each term is a conjunction of different literals not containing a complementary pair x and $\neg x$.

Our proof is based on the 3SAT problem, i.e., each term has exactly three literals.

3SAT Problem: For a CNF expression G in which each term has exactly three literals, is there an assignment of variables such that $G = 1$?

For convenience, an equivalent problem, which is constructed by negating the CNF expression G, is used instead of the above problem. That is, DNF expressions are considered. This problem is denoted by $\overline{\text{3SAT}}$ [15].

$\overline{\textbf{3SAT}}$ Problem: For a DNF expression H in which each term has exactly three literals, is there an assignment of variables such that $H = 0$?

Without loss of generality, it is assumed that $m > 3$ (notice, m is the number of terms in the formula) and each variable and its negation are both in H. Additionally, we need the following assumption.

◇ There is no variable x such that it or its negation $\neg x$ occurs in all terms.

This assumption is reasonable. If there exists a variable such that it or its negation occurs in all terms, we may produce two expressions H' and H'' by assigning this variable the value 1 and 0, respectively. Then, $H = 0$ if and only if $H' = 0 \vee H'' = 0$, while the problem of $H' = 0 \vee H'' = 0$ belongs to 2SAT problem that can be decided in polynomial time [9].

2.5 LBA Acceptance Problem

An LBA is a Turing machine that has a finite tape containing initially a test string with a pair of bound symbols on either side.

Definition 6. *An 8-tuple $\Omega = (Q,\ \Gamma,\ \Sigma,\ \Delta,\ q_0,\ q_f,\ \#,\ \$)$ is an LBA if*

1. *$Q = \{q_0,\ q_1, \cdots,\ q_m,\ q_f\}$, $m \geq 0$, is a set of control states where q_0 is the initial state and q_f is the accept state;*
2. *$\Gamma = \{a_1,\ a_2, \cdots,\ a_n\}$, $n > 0$, is a tape alphabet;*
3. *$\Sigma \subseteq \Gamma$ is an input alphabet;*
4. *$\Delta \subseteq Q \times \Gamma \times \{R,\ L\} \times Q \times \Gamma$ is a set of transitions where R and L represent respectively that the read/write head moves right or left by one cell; and*
5. *$\#$ and $\$$ are two bound symbols that are next to the left and right sides of an input string, respectively.*

We assume that there is no transition from the accept state q_f because the computation is finished correctly once q_f is reached. We also assume that there is a transition sequence $(q_0, \_, \_, q', \_)$, $(q', \_, \_, q'', \_)$, $\cdots$, $(q^{(k)}, \_, \_, q_f, \_)$ in an LBA. Otherwise, the accept state is never reached.

If an LBA is at state p with the read/write head scanning a cell in which symbol a is stored, and there is a transition $\delta = (p,\ a,\ R,\ q,\ b) \in \Delta$, then firing δ causes: 1) the read/write head erase a from the cell, write b in the cell, and move right by one cell; and 2) the LBA be at state q.

LBA Acceptance Problem: For an LBA with a test string, does it accept the string?

This problem is PSPACE-complete even if the LBA is deterministic [9].

3 co-NP-Hardness of the Soundness of Bounded WF-Nets

In this section, we prove that the soundness problem of bounded WF-nets is co-NP-hard based on the $\overline{3\text{SAT}}$ problem.

Let $x_1, x_2, \cdots$, and x_n be n variables and $H = D_1 \vee D_2 \vee \cdots \vee D_m = (l_{1,1} \wedge l_{1,2} \wedge l_{1,3}) \vee (l_{2,1} \wedge l_{2,2} \wedge l_{2,3}) \vee \cdots \vee (l_{m,1} \wedge l_{m,2} \wedge l_{m,3})$ be a DNF expression. The $\overline{3\text{SAT}}$ problem is reducible in polynomial time to the liveness problem of bounded WF-nets, thereby proving that the soundness problem of bounded WF-nets is co-NP-hard by Corollary 1.

For each term D_k, $k \in \mathbb{N}_m$, let $\Psi(D_k)$ denote the set of subscripts of three variables in D_k. For example, if $D = \neg x_1 \wedge x_3 \wedge x_6$, then $\Psi(D) = \{1,\ 3,\ 6\}$.

For each DNF expression H, a WF-net can be constructed by the following method.

Construction 1

- $P = \{i,\ o,\ p_0\}$
  $\cup \{p_k,\ p'_k,\ v_k,\ v'_k,\ c_k,\ c'_k \mid k \in \mathbb{N}_n\}$
- $T = \{b,\ t_0,\ t'_0\}$
  $\cup \{d_j,\ d'_j \mid j \in \mathbb{N}_m\}$
  $\cup \{t_k,\ t'_k,\ e_k,\ e'_k \mid k \in \mathbb{N}_n\}$
- $F = \{(i,\ t_0), (t'_0,\ p_0), (o,\ b), (b,\ i)\}$
  $\cup \{(p_0,\ d_j),\ (d'_j,\ o) \mid j \in \mathbb{N}_m\}$
  $\cup \{(t_0,\ p_k),\ (p'_k,\ t'_0) \mid k \in \mathbb{N}_n\}$
  $\cup \{(d_j,\ v_k),\ (v'_k,\ d'_j) \mid k \in \mathbb{N}_n \setminus \Psi(D_j),\ j \in \mathbb{N}_m\}$
  $\cup \{(p_k,\ t_k),\ (p_k,\ t'_k),\ (t_k,\ p'_k),\ (t'_k,\ p'_k) \mid k \in \mathbb{N}_n\}$

$\cup\{(t_k,\ c_k),\ (t'_k,\ c'_k),\ (c_k,\ e_k),\ (c'_k,\ e'_k)\mid k\in\mathbb{N}_n\}$
$\cup\{(v_k,\ e_k),\ (v_k,\ e'_k),\ (e_k,\ v'_k),\ (e'_k,\ v'_k)\mid k\in\mathbb{N}_n\}$
$\cup\{(c_k,\ d_j)\mid l_{j,1}=x_k\vee l_{j,2}=x_k\vee l_{j,3}=x_k,\ k\in\mathbb{N}_n,\ j\in\mathbb{N}_m\}$
$\cup\{(c'_k,\ d_j)\mid l_{j,1}=\neg x_k\vee l_{j,2}=\neg x_k\vee l_{j,3}=\neg x_k,\ k\in\mathbb{N}_n,\ j\in\mathbb{N}_m\}$

– $M_0=i$

For example, given a DNF expression $H_0=(\neg x_3\wedge x_4\wedge x_5)\vee(x_1\wedge\neg x_2\wedge x_3)\vee(\neg x_1\wedge x_2\wedge x_4)\vee(\neg x_3\wedge\neg x_4\wedge\neg x_5)$, the constructed Petri net is shown as Fig. 2. Notice that, strictly speaking, the Petri net constructed by Construction 1 is not a WF-net but a trivial extension of a WF-net, i.e., if transition b, as a bridge connecting source and sink places, is deleted, then the resulting net is a WF-net. Sometimes, we do not distinguish between a WF-net and its trivial extension if no ambiguity is produced.

Intuitively, the constructed Petri net can be viewed as the composition of two partners. One (e.g., the left section of Fig. 2, i.e., the subnet generated by $\{i,\ t_0,\ t'_0\}\cup\{t_k,\ t'_k,\ p_k,\ p'_k,\ c_k,\ c'_k\mid k\in\mathbb{N}_n\}$) is to set an assignment for variables, and another (e.g., the right section of Fig. 2, i.e., the subnet generated by $\{o,\ p_0\}\cup\{e_k,\ e'_k,\ v_k,\ v'_k,\ c_k,\ c'_k\mid k\in\mathbb{N}_n\}\cup\{d_j,\ d'_j\mid j\in\mathbb{N}_m\}$) is to decide whether H is 0 under the assignment. Variables x_k and $\neg x_k$, $k\in\mathbb{N}_n$, are represented by places c_k and c'_k, respectively. When c_k (resp. c'_k) has a token, it means $x_k=1$ (resp. $\neg x_k=1$).

At the initial marking $M_0=i$, only t_0 is enabled. After firing t_0, only t_1, t'_1, t_2, t'_2, $\cdots$, t_n, and t'_n are enabled, but for the pair t_k and t'_k, firing one will disable another since p_k has only one token. Firing t_k (resp. t'_k) means assigning the value 1 to x_k (resp. $\neg x_k$) since a token is moved into c_k (resp. c'_k). Obviously, x_k and $\neg x_k$ are not assigned true at the same time. In a word, this partner is to set an assignment for variables. Only after each variable is assigned a value, transition t'_0 is enabled. Only after firing t'_0, another partner can start to decide whether the expression H equals to 0 under the corresponding assignment.

Transitions d_1, d_2, $\cdots$, and d_m represent terms D_1, D_2, $\cdots$, and D_m respectively because c_k or c'_k ($k\in\mathbb{N}_n$) is an input place of d_j ($j\in\mathbb{N}_m$) if and only if x_k or $\neg x_k$ occurs in D_j. If none of d_1, d_2, $\cdots$, and d_m is enabled under an assignment, then it means $H=0$ under this assignment. Notice that, this case ($H=0$) implies that the Petri net is not live. If $H=1$ under certain assignment, i.e., some terms are true under this assignment, then the corresponding transitions in $\{d_1,\ d_2,\cdots,\ d_m\}$ are enabled, but only one of these enabled transitions can be fired since p_0 has only one token. Let d_j be an enabled transition under this assignment. Then, after firing d_j, we have that for each $k\in\mathbb{N}_n\setminus\Psi(D_j)$, a token is put into v_k, the token assigned to c_k or c'_k is still retained, and tokens in ${}^\bullet d_j$ are removed. For removing the token from c_k or c'_k where $k\in\mathbb{N}_n\setminus\Psi(D_j)$, e_k or e'_k is competent, i.e., if c_k is marked, then only e_k is enabled, otherwise, only e'_k is enabled. After firing e_k or e'_k, $\forall k\in\mathbb{N}_n\setminus\Psi(D_j)$, d'_j can be fired. Notice that, after firing d'_j, only o has one token, and other places have no tokens. This decision is ended. Finally, firing b means that the initial marking is returned, i.e., a new decision may be started.

In what follows, it is proven that there is an assignment of variables such that $H=0$ if and only if the constructed Petri nets is not live.

Lemma 1. *There is an assignment of variables such that $H=0$ if and only if Petri net $(P,\ T,\ F,\ M_0)$ constructed by Construction 1 is not live.*

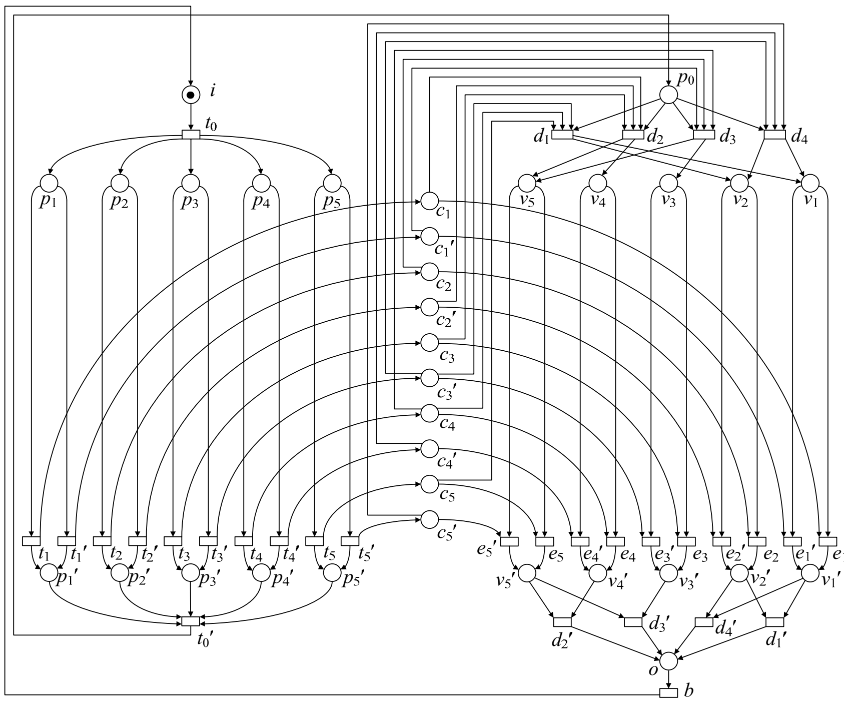

Fig. 2. The WF-net corresponding to $H_0 = (\neg x_3 \wedge x_4 \wedge x_5) \vee (x_1 \wedge \neg x_2 \wedge x_3) \vee (\neg x_1 \wedge x_2 \wedge x_4) \vee (\neg x_3 \wedge \neg x_4 \wedge \neg x_5)$ where x_k (resp. $\neg x_k$), $k \in \mathbb{N}_5$, corresponds to c_k (resp. c_k')

Proof: (***only if***) Let $(\kappa_1, \kappa_2, \cdots, \kappa_n)$ be an assignment of $(x_1, x_2, \cdots, x_n)$ such that $H = 0$, where $\kappa_k = 1$ or $\kappa_k = 0, \forall k \in \mathbb{N}_n$. Then, after firing the transition sequence $t_0\tau_1\tau_2 \cdots \tau_n t_0'$, where $\tau_i = t_i$ if $\kappa_i = 1$ or $\tau_i = t_i'$ if $\kappa_i = 0$, there is no enabled transition. This is because: if there is still an enabled transition after firing $t_0\tau_1\tau_2 \cdots \tau_n t_0'$, this transition must be the one in $\{d_1, d_2, \cdots, d_m\}$, which means that there is a term whose value is 1 under the assignment, thereby making $H = 1$. A contradiction is produced.

(***if***) (by contradiction) Assume that no assignment fulfills $H = 0$, i.e., for each assignment $(\kappa_1, \kappa_2, \cdots, \kappa_n)$, there always exists a term that is true. Through the analysis in the paragraph above this lemma, we know that for each assignment, (P, T, F, M_0) always returns to its initial marking. Therefore, b, t_0, t_0', t_1, t_1', $\cdots$, t_n, and t_n' are obviously live. For each d_j, we can fire the corresponding transitions in $\{t_1, t_1', \cdots, t_n, t_n'\}$ to move tokens into the input places of d_j, thereby ensuring that d_j can be fired. That is, d_j and d_j' are also live. What remains is to show that e_1, e_1',$\cdots$, e_n, and e_n' are live. Obviously, if the pre-set of v_k is not empty, then e_k and e_k' are live, $\forall k \in \mathbb{N}_n$. By the previous assumption (i.e., there is no variable x such that it or its negation $\neg x$ occurs in each term), we know that for each v_k, its pre-set is not empty, because there always is a term D_j such that $k \in \mathbb{N}_n \setminus \Psi(D_j)$, i.e., x_k and $\neg x_k$ are not in D_j. Hence, e_k and e_k' are also live, $\forall k \in \mathbb{N}_n$. □

Notice that, by the above conclusion we know that Petri net $(P,\ T,\ F,\ M_0)$ constructed by Construction 1 is live if and only if for each assignment of variables there is $H = 1$. The problem on deciding whether there is always $H = 1$ for each assignment of variables is co-NP-complete [9].

Corollary 2. *Petri net $(P,\ T,\ F,\ M_0)$ constructed by Construction 1 is live if and only if $H = 1$ for each assignment of variables.*

Lemma 2. *Let $(P,\ T,\ F,\ M_0)$ be the Petri net constructed for a DNF H by Construction 1. Then, $(P,\ T \setminus \{b\},\ F \setminus \{(o,\ b), (b,\ i)\},\ M_0)$ is a bounded WF-net.*

Proof: It is obvious that for each transition $t \in T \setminus \{b\}$ (resp. each place $p \in P$) in the net $(P,\ T \setminus \{b\},\ F \setminus \{(o,\ b), (b,\ i)\})$, there is a directed path from i to o such that t (resp. p) occurs in it. Therefore, $(P,\ T,\ F)$ is strongly connected. Therefore, $(P,\ T \setminus \{b\},\ F \setminus \{(o,\ b), (b,\ i)\})$ is a WF-net.

Obviously, $i,\ o,\ p_0,\ p_1,\ p_1', \cdots,\ p_n,\ p_n',\ v_1,\ v_1',\ \cdots,\ v_n$, and v_n' are all bounded in $(P,\ T,\ F,\ M_0)$ because the two subnets generated by them represent the state transition of the two partners and are easily shown to be bounded. We only need to observe places $v_1,\ v_1',\ \cdots,\ v_n$, and v_n'. In the case that $(P,\ T,\ F,\ M_0)$ is not live: all deadlock states satisfy that p_0 has a token, each pair c_k and c_k' has a token, and others have no token. In the case that $(P,\ T,\ F,\ M_0)$ is live: before o is marked, each pair c_k and c_k' has at most one token, and when o is marked, all tokens in $c_1,\ c_1',\ \cdots,\ c_n$, and c_n' are removed. Hence, $(P,\ T,\ F,\ M_0)$ is bounded, thereby $(P,\ T \setminus \{b\},\ F \setminus \{(o,\ b), (b,\ i)\},\ M_0)$ bounded. $\square$

Theorem 2. *The soundness problem for bounded WF-nets is co-NP-hard.*

Proof: For each DNF expression in which there are n variables and m terms and each term has 3 literals, it is easy to compute that the constructed WF-net has $6n + 3$ places, $4n + 2m + 3$ transitions, and $2mn + 14n - m + 4$ arcs. Therefore, the WF-net can be constructed in polynomial time (i.e., $O(2mn + 24n + m + 10)$). Therefore, it is known by Lemma 2 and Corollaries 1 and 2 that the soundness problem of bounded WF-net is co-NP-hard. $\square$

4 PSPACE-Hardness of the Soundness of Bounded reWF-Nets

Obviously, the reachability, liveness, and soundness are all decidable for bounded reWF-nets since we can construct their reachability graph by which these properties can be decided. However, we are to show that they are PSPACE-hard. First, we prove that the liveness and soundness are equivalent for bounded reWF-nets.

Lemma 3. *Let $N = (P,\ T,\ F,\ R)$ be an reWF-net such that $(N^E,\ M_0) = (P,\ T \cup \{b\},\ F \cup \{(b,\ i),\ (o,\ b)\},\ R,\ i)$ is bounded. Then, N is sound if and only if $(N^E,\ M_0)$ is live.*

Proof: (***only if***) Please see Lemma 5.1 in [4].

(***if***) First, let $M_d = o$. Because $(N^E,\ M_0)$ is live, b is live. Hence, $\exists M \in R(N^E,\ M_0)$: $M[b\rangle$. Hence, $M \geq M_d$. Let $M_0[\sigma\rangle M$ and $M[b\rangle M'$. Next, we use the contradiction method to prove $M = M_d$. Assume that there is place $p \in P$ such that $M(p) > M_d(p) = 0$. Then, $M'(p) > M_d(p) = 0$ and $M'(i) = 1 = M_0(i)$ since there is no reset arc between b and p, i.e., firing b does not empty p. Hence, σb can fire infinitely, thereby making place p unbounded. This contradicts the boundedness of $(N^E,\ M_0)$. Hence, for each marking $M \in R(N^E,\ M_0)$, if $M \geq M_d$, then $M = M_d = o$. Hence, $R(N^E,\ M_0) = R(N,\ M_0)$. Hence, by using the liveness of $(N^E,\ M_0)$, we can easily prove that 1) $\forall M \in R(N,\ M_0)$: $M_d \in R(N,\ M)$; and 2) $\forall t \in T, \exists M \in R(N,\ M_0)$: $M[t\rangle$. Hence, N is sound. □

Next, we prove that the soundness problem is PSPACE-hard for bounded reWF-nets.

Given an LBA $\Omega = (Q,\ \Gamma,\ \Sigma,\ \Delta,\ q_0,\ q_f,\ \#,\ \$)$ with an input string S, an reWF-net can be constructed. The construction below is refered as **Construction 2**. We first assume that the length of S is l, $l \geq 0$, and the i-th element of S is denoted as S_i. $Q = \{q_0,\ q_1, \cdots,\ q_m,\ q_f\}$, $m \geq 0$. $\Gamma = \{a_1,\ a_2, \cdots,\ a_n\}$, $n > 0$. Cells storing $\#S\$$ are labeled $0, 1, \cdots, l$, and $l+1$, respectively.

– $P = \{i,\ o,\ p_0,\ p'_0\}$
 $\cup\{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\}$
 $\cup\{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$

A token in $A_{0,\#}$ (resp. $A_{l+1,\$}$) means that the tape cell 0 (resp. $l+1$) stores $\#$ (resp. \$). Therefore, once the computation starts, $A_{0,\#}$ (resp. $A_{l+1,\$}$) has a token until the computation ends since the two special symbols are not allowed to be replaced by other symbols. A token in $A_{i,j}$ means that the symbol in the cell i is a_j. A token in $B_{i,j}$ means that the read/write head is on the cell i and the machine is at state q_j. Notice that, in the computing process, the LBA is only at one state at any time, thus only one place in $\{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ is marked by one token in the modeling process. Once the computation finishes, i.e., the LBA accepts the input string S, the token in $\{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ is moved into p_0. The purpose of having p'_0 will be discussed later.

The sets of transitions, arcs, and reset arcs of the reWF-net are constructed as follows.

– b is a transition such that $^\bullet b = \{o\}$ and $b^\bullet = \{i\}$.
– t_s is a transition such that $^\bullet t_s = \{i\}$ and $t_s^\bullet = \{A_{0,\#},\ A_{l+1,\$},\ B_{0,0}\} \cup \{A_{i,j} \mid S_i = a_j,\ i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\}$. In fact, $t_s^\bullet$ corresponds to the initial configuration of the LBA, i.e., tokens in $\{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid S_i = a_j,\ i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\}$ represent the input string with the two bound symbols, and the token in $B_{0,0}$ represents that the machine is at the initial state q_0 and the read/write head is on the leftmost cell.
– t'_s is a transition such that $^\bullet t'_s = \{p'_0\}$ and $t'^\bullet_s = \{p_0\} \cup \{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$. Later, we will explain the reason of having t'_s.
– t_e is a transition such that $^\bullet t_e = \{p_0\}$ and $t_e^\bullet = \{p'_0\}$. Only t_e connects with reset arcs such that $^\circ t_e = \{p_0\} \cup \{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$. The token in p_0 means that the computation ends, and then firing t_e will empty all places in $\{p_0\} \cup \{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$, i.e., the tape is emptied and the machine is not at any state.
– t'_e is a transition such that $^\bullet t'_e = \{p'_0\}$ and $t'^\bullet_e = \{o\}$

For each transition $\delta \in \Delta$, we construct transitions of the reWF-net. We first consider transitions in Δ that make the LBA to enter the accept state q_f.

- If δ is the form of $(q_h,\ \#,\ R,\ q_f,\ \#)$, $h \in \{0,\ 1, \cdots,\ m\}$, i.e., the LBA halts correctly and the read/write head is on the leftmost cell, we construct a transition t of the reWF-net such that ${}^\bullet t = \{A_{0,\#},\ B_{0,h}\}$ and $t^\bullet = \{A_{0,\#},\ p_0\}$.
- If δ is the form of $(q_h,\ \$,\ L,\ q_f,\ \$)$, $h \in \{0,\ 1, \cdots,\ m\}$, i.e., the LBA halts correctly and the read/write head is on the rightmost cell, we construct a transition t such that ${}^\bullet t = \{A_{l+1,\$},\ B_{l+1,h}\}$ and $t^\bullet = \{A_{l+1,\$},\ p_0\}$.
- If δ is the form of $(q_h,\ a_j,\ L/R,\ q_f,\ a_k)$, $h \in \{0,\ 1, \cdots,\ m\}, j,\ k \in \mathbb{N}_n$, i.e., the LBA halts correctly but the read/write head is possibly on any cell, we construct for each cell r ($r \in \mathbb{N}_l$) a transition t_r. Formally, $\forall\, r \in \mathbb{N}_l$, a transition t_r is constructed such that ${}^\bullet t_r = \{A_{r,j},\ B_{r,h}\}$ and $t_r^\bullet = \{A_{r,k},\ p_0\}$.

Next, we consider other transitions in Δ that have no p_f.

- If δ is the form of $(q_h,\ \#,\ R,\ q_i,\ \#)$, $h,\ i \in \{0,\ 1, \cdots,\ m\}$, i.e., the read/write head scans $\#$, $\#$ is rewritten, the read/write head moves right, and the state is changed into q_i from q_h. For this δ, we construct a transition t of the reWF-net such that ${}^\bullet t = \{A_{0,\#},\ B_{0,h}\}$ and $t^\bullet = \{A_{0,\#},\ B_{1,i}\}$.
- If δ is the form of $(q_h,\ \$,\ L,\ q_i,\ \$)$, $h,\ i \in \{0,\ 1, \cdots,\ m\}$, i.e., the read/write head scans \$, \$ is rewritten, the read/write head moves left, and the state is changed into q_i from q_h. For this δ, we construct a transition t such that ${}^\bullet t = \{A_{l+1,\$},\ B_{l+1,h}\}$ and $t^\bullet = \{A_{l+1,\$},\ B_{l,i}\}$.
- If δ is the form of $(q_h,\ a_j,\ R,\ q_i,\ a_k)$, $h, i \in \{0,\ 1, \cdots,\ m\}, j,\ k \in \mathbb{N}_n$, then we construct l transitions, i.e., we should consider each cell. Formally, $\forall\, r \in \mathbb{N}_l$, a transition t_r is constructed such that ${}^\bullet t_r = \{A_{r,j},\ B_{r,h}\}$ and $t_r^\bullet = \{A_{r,k},\ B_{r+1,i}\}$.
- If δ is the form of $(q_h,\ a_j,\ L,\ q_i,\ a_k)$, $h,\ i \in \{0,\ 1, \cdots,\ m\}, j,\ k \in \mathbb{N}_n$, then we also construct l transitions, i.e., $\forall\, r \in \mathbb{N}_l$, a transition t_r is constructed such that ${}^\bullet t_r = \{A_{r,j},\ B_{r,h}\}$ and $t_r^\bullet = \{A_{r,k},\ B_{r-1,i}\}$.

In the running process of the Petri net, a marking over $\{p_0\} \cup \{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ corresponds to a configuration of the LBA, i.e., tokens in $\{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\}$ correspond to the string on the tape, and the token in $\{p_0\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ represents the current state of the LBA.

We know that the LBA halts correctly and accepts the input string when a token enters p_0, and at this marking, only t_e is enabled. Firing t_e makes all places in $\{p_0\} \cup \{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ emptied because t_e have reset arcs with these places. However, some transitions, which correspond to Δ, are not necessarily enabled in the computing process. It is because for an LBA with an acceptable input string, not all transitions of the LBA are used in the deciding process. Therefore, we use t_s' to produce a token for each place in $\{p_0\} \cup \{A_{0,\#},\ A_{i,j},\ A_{l+1,\$} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$, which makes all transitions corresponding to Δ have an enabling right again. Clearly, once t_e is fired, all places in $\{p_0\} \cup \{A_{0,\#},\ A_{i,j},\ A_{l+1,\$} \mid i \in \mathbb{N}_l,\ j \in$

$\mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ are emptied again. This guarantees that all transitions corresponding to Δ are live when the LBA accepts the input string. Obviously, if t'_e and then b are fired, the initial marking, $M_0 = i$, is returned.

Notice that, places in $\{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$ are set in order to consider all possible cases for each cell, but because Δ is finite, some of these places are not used or only have input or output transitions. This makes the constructed net not strongly connected. Therefore, we add a transition d such that

– for each place p in $\{A_{0,\#},\ A_{l+1,\$},\ A_{i,j} \mid i \in \mathbb{N}_l,\ j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0,\ 1, \cdots,\ l+1\},\ j \in \{0,\ 1, \cdots,\ m\}\}$, $p \in {}^\bullet d$ and $p \in d^\bullet$.

This ensures that the constructed net is strongly connected. Obviously, d does not influence the behavior of the constructed Petri net, and only after firing t'_s, d is enabled. Note that, even though there is no transition d, there also exists a direct path from place i to place o, which is guaranteed by the previous assumption (i.e., for an LBA, there always exists a transition sequence $(q_0, \_, \_, q', \_), (q', \_, \_, q'', \_), \cdots, (q^{(k)}, \_, \_, q_f, \_)$.).

For example, the LBA $\Omega_0 = (Q,\ \Gamma,\ \Sigma,\ \Delta,\ q_0,\ q_f,\ \#,\ \$)$ can produce the language $\{a^{i_1}b^{i_1}a^{i_2}b^{i_2} \cdots a^{i_m}b^{i_m} \mid i_1, i_2, \cdots, i_m, m \in \mathbb{N}\}$, where

– $Q = \{q_0,\ q_1,\ q_2,\ q_3,\ q_f\}$
– $\Gamma = \{a,\ b,\ X\}$
– $\Sigma = \{a,\ b\}$
– $\Delta = \{(q_0,\ \#,\ R,\ q_1,\ \#),\ (q_1,\ \$,\ L,\ q_f,\ \$),\ (q_1,\ X,\ R,\ q_1,\ X),\ (q_1,\ a,\ R,\ q_2,\ X),$
$(q_2,\ a,\ R,\ q_2,\ a),\ (q_2,\ X,\ R,\ q_2,\ X),\ (q_2,\ b,\ L,\ q_3,\ X),\ (q_3,\ a,\ L,\ q_3,\ a),$
$(q_3,\ X,\ L,\ q_3,\ X),\ (q_3,\ \#,\ R,\ q_1,\ \#)\}$

Notice that, this LBA accepts the empty string. For the LBA with the input string ab, the Petri net constructed by Construction 2 is $(N^E,\ M_0) = (P,\ T,\ F,\ R,\ i)$ where

– $P = \{i,\ o,\ p_0,\ p'_0\}$
$\cup\{A_{0,\#},\ A_{3,\$},\ A_{i,j} \mid i \in \mathbb{N}_2,\ j \in \mathbb{N}_3\}$
$\cup\{B_{i,j} \mid i,\ j \in \{0,\ 1,\ 2,\ 3\}\}$
– $T = \{b,\ t_s,\ t'_s,\ t_e,\ t'_e,\ d\}$
$\cup\{t_0,\ t_1,\ t_2,\ t_{i,j} \mid i \in \mathbb{N}_7,\ j \in \mathbb{N}_2\}$
– $F = \{(p_0,\ t_e),\ (t_e,\ p'_0),\ (p'_0,\ t'_e),\ (t'_e,\ o),\ (o,\ b),\ (b,\ i)\}$
$\cup\{(i,\ t_s),\ (t_s,\ A_{0,\#}),\ (t_s,\ A_{3,\$}),\ (t_s,\ A_{1,1}),\ (t_s,\ A_{2,2}),\ (t_s,\ B_{0,0})\}$
$\cup\{(t_0,\ p_0),\ (t_0,\ A_{3,\$}),\ (A_{3,\$},\ t_0),\ (B_{3,1},\ t_0)\}$
$\cup\{(t_1,\ B_{1,1}),\ (t_1,\ A_{0,\#}),\ (A_{0,\#},\ t_1),\ (B_{0,0},\ t_1)\}$
$\cup\{(t_2,\ B_{1,1}),\ (t_2,\ A_{0,\#}),\ (A_{0,\#},\ t_2),\ (B_{0,3},\ t_2)\}$
$\cup\{(t_{1,1},\ B_{2,1}),\ (t_{1,1},\ A_{1,3}),\ (A_{1,3},\ t_{1,1}),\ (B_{1,1},\ t_{1,1})\}$
$\cup\{(t_{1,2},\ B_{3,1}),\ (t_{1,2},\ A_{2,3}),\ (A_{2,3},\ t_{1,2}),\ (B_{2,1},\ t_{1,2})\}$
$\cup\{(t_{2,1},\ B_{2,1}),\ (t_{2,1},\ A_{1,3}),\ (A_{1,1},\ t_{2,1}),\ (B_{1,1},\ t_{2,1})\}$
$\cup\{(t_{2,2},\ B_{3,1}),\ (t_{2,2},\ A_{2,3}),\ (A_{2,1},\ t_{2,2}),\ (B_{2,1},\ t_{2,2})\}$
$\cup\{(t_{3,1},\ B_{2,2}),\ (t_{3,1},\ A_{1,1}),\ (A_{1,1},\ t_{3,1}),\ (B_{1,2},\ t_{3,1})\}$
$\cup\{(t_{3,2},\ B_{3,2}),\ (t_{3,2},\ A_{2,1}),\ (A_{2,1},\ t_{3,2}),\ (B_{2,2},\ t_{3,2})\}$
$\cup\{(t_{4,1},\ B_{2,2}),\ (t_{4,1},\ A_{1,3}),\ (A_{1,3},\ t_{4,1}),\ (B_{1,2},\ t_{4,1})\}$

$\cup\{(t_{4,2},\ B_{3,2}),\ (t_{4,2},\ A_{2,3}),\ (A_{2,3},\ t_{4,2}),\ (B_{2,2},\ t_{4,2})\}$
$\cup\{(t_{5,1},\ B_{0,3}),\ (t_{5,1},\ A_{1,3}),\ (A_{1,2},\ t_{5,1}),\ (B_{1,2},\ t_{5,1})\}$
$\cup\{(t_{5,2},\ B_{1,3}),\ (t_{5,2},\ A_{2,3}),\ (A_{2,2},\ t_{5,2}),\ (B_{2,2},\ t_{5,2})\}$
$\cup\{(t_{6,1},\ B_{0,3}),\ (t_{6,1},\ A_{1,1}),\ (A_{1,1},\ t_{6,1}),\ (B_{1,3},\ t_{6,1})\}$
$\cup\{(t_{6,2},\ B_{1,3}),\ (t_{6,2},\ A_{2,1}),\ (A_{2,1},\ t_{6,2}),\ (B_{2,3},\ t_{6,2})\}$
$\cup\{(t_{7,1},\ B_{0,3}),\ (t_{7,1},\ A_{1,3}),\ (A_{1,3},\ t_{7,1}),\ (B_{1,3},\ t_{7,1})\}$
$\cup\{(t_{7,2},\ B_{1,3}),\ (t_{7,2},\ A_{2,3}),\ (A_{2,3},\ t_{7,2}),\ (B_{2,3},\ t_{7,2})\}$
$\cup\{(p'_0,\ t'_s),\ (t'_s,\ A_{0,\#}),\ (t'_s,\ A_{3,\$})\}$
$\cup\{(t'_s,\ A_{i,j})\mid i\in\mathbb{N}_2,\ j\in\mathbb{N}_3\}$
$\cup\{(t'_s,\ B_{i,j})\mid i,\ j\in\{0,\ 1,\ 2,\ 3\}\}$
$\cup\{(d,\ A_{0,\#}),\ (d,\ A_{3,\$}),\ (A_{0,\#},\ d),\ (A_{3,\$},\ d)\}$
$\cup\{(d,\ A_{i,j}),\ (A_{i,j},\ d)\mid i\in\mathbb{N}_2,\ j\in\mathbb{N}_3\}$
$\cup\{(d,\ B_{i,j}),\ (B_{i,j},\ d)\mid i,\ j\in\{0,\ 1,\ 2,\ 3\}\}$

– $R=\{[A_{0,\#},\ t_e],\ [A_{3,\$},\ t_e],\ [p_0,\ t_e]\}$
$\cup\{[A_{i,j},\ t_e]\mid i\in\mathbb{N}_2,\ j\in\mathbb{N}_3\}$
$\cup\{[B_{i,j},\ t_e]\mid i,\ j\in\{0,\ 1,\ 2,\ 3\}\}$

Fig. 3 shows another constructed Petri net that corresponds to the above LBA Ω_0 with an empty string. Notice, the following (reset) arcs are not drawn in Fig. 3:

$t_s'^{\bullet}=\{p_0,\ A_{0,\#},\ A_{1,\$},\ B_{0,0},\ B_{0,1},\ B_{0,2},\ B_{0,3},\ B_{1,0},\ B_{1,1},\ B_{1,2},\ B_{1,3}\}$
${}^{\circ}t_e=\{p_0,\ A_{0,\#},\ A_{1,\$},\ B_{0,0},\ B_{0,1},\ B_{0,2},\ B_{0,3},\ B_{1,0},\ B_{1,1},\ B_{1,2},\ B_{1,3}\}$
${}^{\bullet}d=d^{\bullet}=\{A_{0,\#},\ A_{1,\$},\ B_{0,0},\ B_{0,1},\ B_{0,2},\ B_{0,3},\ B_{1,0},\ B_{1,1},\ B_{1,2},\ B_{1,3}\}$

Clearly, if there are no transition d as well as the related arcs, the constructed net is not strongly connected. Notice that, because the input string is an empty one, there are no transitions for $(q_1,\ X,\ R,\ q_1,\ X)$, $(q_1,\ a,\ R,\ q_2,\ X)$, $(q_2,\ a,\ R,\ q_2,\ a)$, $(q_2,\ X,\ R,\ q_2,\ X)$, $(q_2,\ b,\ L,\ q_3,\ X)$, $(q_3,\ a,\ L,\ q_3,\ a)$, and $(q_3,\ X,\ L,\ q_3,\ X)$. Transitions t_0, t_1, and t_2 correspond to $(q_0,\ \#,\ R,\ q_1,\ \#)$, $(q_3,\ \#,\ R,\ q_1,\ \#)\}$, and $(q_1,\ \$,\ L,\ q_f,\ \$)$, respectively. Firing t_s produces respectively one token for $A_{0,\#}$, $A_{1,\$}$, and $B_{0,0}$, which represents the initial configuration of the LBA, i.e., the tape stores an empty string (i.e., only two bound symbols are on the tape), the machine is at state q_0, and the read/write head is on the leftmost cell. At this marking, only transition t_0 is enabled. t_0 corresponds to $(q_0,\ \#,\ R,\ q_1,\ \#)$. At the initial configuration, only $(q_0,\ \#,\ R,\ q_1,\ \#)$ is enabled. After firing $(q_0,\ \#,\ R,\ q_1,\ \#)$, the configuration of the LBA is that the machine is at state q_1 and the read/write head moves right (i.e., it is moved on the cell storing \$). This is in accordance with t_0 because firing t_0 removes the token from $B_{0,0}$ and put a token into $B_{1,1}$. At this marking, only t_2 is enabled, which is also in accordance with the LBA since at the corresponding configuration only $(q_1,\ \$,\ L,\ q_f,\ \$)$ is valid. Firing t_2 moves a token into p_0, which means the LBA accepts this input string.

Lemma 4. *An LBA accepts an input string if and only if the Petri net constructed by Construction 2 is live.*

Proof: (***only if***) Because the LBA accepts the input string, we have that for each marking $M\in R(N,\ M_0)$ such that $M_0[\sigma\rangle M$ but t_e is not in σ, there is a reachable marking

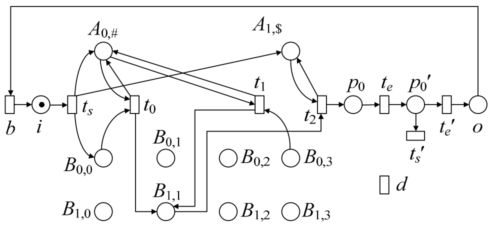

Fig. 3. The reWF-net corresponding to the LBA Ω_0 with the empty string

$M' \in R(N, M)$ such that p_0 is marked at M'. At marking M', only transition t_e is enabled. After firing t_e, marking $M'' = p_0'$ is reached. At marking M'', only t_e' or t_s' is enabled. Firing t_s' produces a token for each place in $\{p_0\} \cup \{A_{0,\#}, A_{i,j}, A_{l+1,\$} \mid i \in \mathbb{N}_l, j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0, 1, \cdots, l+1\}, j \in \{0, 1, \cdots, m\}\}$, which makes transition d and transitions corresponding to Δ have an enabling right. Clearly, once t_e is fired again, all places in $\{p_0\} \cup \{A_{0,\#}, A_{i,j}, A_{l+1,\$} \mid i \in \mathbb{N}_l, j \in \mathbb{N}_n\} \cup \{B_{i,j} \mid i \in \{0, 1, \cdots, l+1\}, j \in \{0, 1, \cdots, m\}\}$ are emptied again. If firing t_e' at marking M'' and then firing b, the initial marking, $M_0 = i$, is returned. Therefore, the constructed Petri net is live.

(*if*) By Construction 2 we know that when the constructed Petri net is live, there is a reachable marking such that transition t_e is enabled at this marking, i.e., this marking marks place p_0. By Construction 2 we know that if place p_0 can be marked by some reachable marking, then the LBA accepts the input string. □

Lemma 5. *Let (P, T, F, R, M_0) be the Petri net constructed for an LBA with an input string by Construction 2. Then, $(P, T \setminus \{b\}, F \setminus \{(o, b), (b, i)\}, R, M_0)$ is a bounded reWF-net.*

Proof: Clearly, (P, T, F, R, M_0) is strongly connected after deleting all reset arcs. Therefore, $(P, T \setminus \{b\}, F \setminus \{(o, b), (b, i)\}, R, M_0)$ is an reWF-net. For boundedness, we only need to observe the constructed transitions corresponding to Δ. Since each of them has two input places and two output places, they neither increase nor decrease the number of tokens at any time. □

Theorem 3. *The soundness problem for bounded reWF-nets is PSPACE-hard.*

Proof: It is derived by Lemmas 3, 4, and 5. Notice that, the reWF-net can be constructed in $O(l \cdot (m + n + k))$ time where $l = \mid S \mid$, $m = \mid Q \mid$, $n = \mid \Gamma \mid$, and $k = \mid \Delta \mid$. □

The soundness of bounded reWF-nets is decidable but PSPACE-hard. However, it is still an open problem whether the boundedness problem of reWF-nets is decidable. What we know is that the boundedness problem is undecidable for general Petri nets with reset arcs [7,8].

Similarly, based on the LBA Acceptance problem, we can prove that the soundness problem of bounded WF-nets with inhibitor arcs is PSPACE-hard. The boundedness problem is undecidable for general Petri nets with inhibitor arcs [10,19].

5 Conclusion

The SAT problem is shown to be polynomial time reducible to the soundness problem for bounded WF-nets. This implies the latter is co-NP-hard. The soundness problem of bounded reWF-nets is proven to be PSPACE-hard by reducing the LBA Acceptance problem to it in polynomial time. Co-NP-hardness (resp. PSPACE-hardness) of the soundness problem of bounded WF-nets (resp. reWF-nets) shows a lower limit of the complexity. Therefore, future work focuses on finding if the soundness problem is co-NP-complete for bounded WF-nets and PSPACE-complete for bounded reWF- and inWF-nets.

The results in this paper is meaningful in theory. However, WF- and reWF-nets are of the strong application background in industry and often have special structures such as AND-join and OR-split [3]. Therefore, it is our future work to explore efficient analysis methods for specially structural WF- and reWF-nets.

Acknowledgement. We would like to thank Professor W.M.P. van der Aalst who encourages us to explore the complexity of the soundness problem for general WF-nets, and also thank four anonymous reviewers whose constructive comments improve the qualities of this paper. This paper was partially supported by research grant "SUTD-ZJU/RES/05/201".

References

1. Van der Aalst, W.M.P.: Interorganizational Workflows: An Approach Based on Message Sequence Charts and Petri Nets. System Analysis and Modeling 34, 335–367 (1999)
2. Van der Aalst, W.M.P.: Loosely Coupled Interorganizational Wokflows: Modeling and Analyzing Workflows Crossing Organizational Boundaries. Inf. Manage. 37, 67–75 (2000)
3. Van der Aalst, W.M.P.: Structural Characterizations of Sound Workflow Nets. Computing Science Report 96/23, Eindhoven University of Technology (1996)
4. Van der Aalst, W.M.P., Van Hee, K.M., Ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of Workflow Nets: Classification, Decidability, and Analysis. Formal Aspects of Computing 23, 333–363 (2011)
5. Cheng, A., Esparza, J., Palsberg, J.: Complexity Results for 1-safe Nets. Theoretical Computer Science 147, 117–136 (1995)
6. Desel, J., Esparza, J.: Free Choice Petri Nets. Cambridge Tracts in Theoretical Computer Science, vol. 40. Cambridge University Press, Cambridge (1995)
7. Dufourd, C., Finkel, A., Schnoebelen, P.: Reset Nets Between Decidability and Undecidability. In: Larsen, K.G., Skyum, S., Winskel, G. (eds.) ICALP 1998. LNCS, vol. 1443, pp. 103–115. Springer, Heidelberg (1998)
8. Dufourd, C., Jančar, P., Schnoebelen, P.: Boundedness of Reset P/T Nets. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 301–310. Springer, Heidelberg (1999)
9. Garey, M.R., Johnson, D.S.: Computer and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company (1976)
10. Hack, M.: Petri Net Languages. Technical Report 159. MIT (1976)
11. van Hee, K.M., Sidorova, N., Voorhoeve, M.: Generalised Soundness of Workflow Nets Is Decidable. In: Cortadella, J., Reisig, W. (eds.) ICATPN 2004. LNCS, vol. 3099, pp. 197–215. Springer, Heidelberg (2004)

12. Kang, M.H., Park, J.S., Froscher, J.N.: Access Control Mechanisms for Inter-organizational Workflow. In: Proc. of the Sixth ACM Symposium on Access Control Models and Technologies, pp. 66–74. ACM Press, New York (2001)
13. Kindler, E.: The ePNK: An Extensible Petri Net Tool for PNML. In: Kristensen, L.M., Petrucci, L. (eds.) PETRI NETS 2011. LNCS, vol. 6709, pp. 318–327. Springer, Heidelberg (2011)
14. Kindler, E., Martens, A., Reisig, W.: Inter-operability of Workflow Applications: Local Criteria for Global Soundness. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) BPM 2000. LNCS, vol. 1806, pp. 235–253. Springer, Heidelberg (2000)
15. Ohta, A., Tsuji, K.: NP-hardness of Liveness Problem of Bounded Asymmetric Choice Net. IEICE Trans. Fundamentals E85-A, 1071–1074 (2002)
16. Tiplea, F.L., Bocaneala, C.: Decidability Results for Soundness Criteria of Resource-Constrained Workflow Nets. IEEE Trans. on Systems, man, and Cybernetics, Part A: Systems and Humans 42, 238–249 (2011)
17. Verbeek, H.M.W., Van der Aalst, W.M.P., Ter Hofstede, A.H.M.: Verifying Worklows with Cancellation Regions and OR-joins: An Approach Based on Relaxed Soundness and Invariants. Computer Journal 50, 294–314 (2007)
18. Verbeek, H.M.W., Wynn, M.T., Van der Aalst, W.M.P., Ter Hofstede, A.H.M.: Reduction Rules for Reset/Inhibitor Nets. BMP Center Report BMP-07-13, BMP-center.org (2007)
19. Van der Vlugt, S., Kleijn, J., Koutny, M.: Coverability and Inhibitor Arcs: An Example. Technical Report 1293, University of Newcastle Upon Tyne (2011)