Egyptian Computer Science Journal Vol. 37 No. 6 September 2013 ISSN-1110-2586

De/compositional M_w Automaton-Based Reachability Algorithm

Štefan Korečko, Štefan Hudák, Jozef Doboš, Slavomír Šimoňák

Department of Computers and Informatics,
Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 041 20 Košice, Slovakia
stefan.korecko@tuke.sk, stefan.hudak@tuke.sk, jozef.dobos@student.tuke.sk,
slavomir.simonak@tuke.sk

Abstract

This paper introduces a new reachability problem (RP) solving algorithm for the (pure) place/transition nets. This algorithm combines the existing M_w automaton-based method with the de/compositional approach called T-junction. The existing method is built around a special type of finite state automaton, called M_w automaton, and reduces RP to an instance of modified integer linear programming problem. The T-junction divides a place/transition net into subnets with common transitions. Possibilities of parallel implementation of the new algorithm are also described.

Keywords: *Petri nets, reachability problem, M_w automaton, T-junction de/composition, parallelisation*

1. Introduction

In [3] an original method (algorithm) to analyse and solve the reachability problem (RP) for (pure) place/transition (P/T) nets [1] in general case was introduced. The preliminary version of the method was presented at The Second Workshop on Applications and Theory of Petri Nets in September 1981, at Bad Honnef, Germany. The centrepiece of the method is a structure named *finite state automaton of the type M_w* or simply *M_w automaton*. The analysis of the structure is based on results of automata theory and the convex analysis of the state space represented by M_w. The notion of M_w coincides to some extent to that of the coverability graph [11], but they differ in significant details. The reachability analysis by the method reduces the reachability problem (RP) to the integer linear programming problem (ILP), to be more precise to an instance of the modified ILP (MILP)[3]. The latter adds to ILP a requirement of proving validity of some predicate *con*, which holds if and only if there is a path in the state space leading from the ILP solution point X to the initial point in the hypercube $C(X)$. Recently it was discovered [4] that the M_w automaton allows to coin a new approach to reveal deadlocks in P/T net, based solely on the net in question and its automaton.

Despite of its age the method presented in [3] maintains its originality even compared to the newest approaches to RP solving, such as [7], [8] and [9]: The method [3] is based on a study of linguistic properties of P/T net computations while the other approaches are state-oriented. However, we found a strong relation among results presented in [8], [9] and our ones, for example marked reachability graphs [8] and production graphs [9] have properties similar to M_w and sets of (co-)reachable configurations from [8] are close to prefix and suffix

languages from [3]. The advantages of the method [3] are a systematic way of M_w construction and a reusability of M_w for every RP instance of a given net. In addition, the work [3] also contains a derivation of the worst-case time complexity of RP and an estimation of the lower bound of the RP space complexity. The upper bound of the worst-case time complexity has been established as $O(2^{b^{2^{k+1}}.k^2})$, where k a is number of places of given P/T net and b is some constant, $b>0$. It should be also noted that in the case of unbounded P/T nets the M_w automaton represents a decomposition of state space that reminds the classical Kosaraju-Lambert-Mayr-Sacerdote-Tenney (KLMST) decomposition (the shortcut is borrowed from [9]). This is quite natural because the method [3] and the results of Kosaraju [6] and Mayr [10] originated from the same time period, as it is also noticed in [5].

The complexity is the main obstacle to solve RP in the cases of practical size, expressed in terms of cardinality of the set of places of P/T net in question. To cope with this tremendous complexity three de/compositional approaches to RP solving have been introduced [2], [3]: T-junction, where the net is split into subnets with some common transitions, P-junction, where subnets have some common places and PT-junction with common places and transitions. T-junction turned out to be the most promising one for several reasons. First, it promises the most effective reduction of the complexity, as the number of places plays the prominent role in it. The second one is relative simplicity of backward composition of final result of RP solving. And, finally, it has nice properties with respect to a possible parallel implementation of the algorithm.

This paper extends the existing theory of the T-junction de/composition [2], [3] by introducing a new RP solving algorithm that combines the original method from [3] with the T-junction approach and by reasoning about possibilities of parallelisation of the new algorithm. The rest of the paper is organised as follows: The next section deals with basic definitions and notations. The RP algorithm from [3], which we call here *RP/M_w algorithm*, is briefly described in section 3 and the new one is presented in section 4. In the final section we summarize the contribution of the paper and work related to it and outline some tasks for future research and development.

2. Preliminary

By $\mathbb{N}$ we denote the set of *natural numbers {0,1,2,...}*, by $\mathbb{Z}$ the set of all *integers* and by $\mathbb{Z}^k$ ($\mathbb{N}^k$) the set of k-dimensional (non-negative) *integer vectors*. For these vectors we use the relations $=$, $\leq$ and $<$ defined as

$$q \,\mathbf{op}\, r \Leftrightarrow \forall i (1 \leq i \leq k) : q_i \,\mathbf{op}\, r_i, \mathbf{op} \in \{<, =, \leq\}$$

where $q = (q_1, \ldots, q_k)$ and $r = (r_1, \ldots, r_k)$. By the symbol ω we mean a value greater than any integer. This implies that $\forall a (a \in \mathbb{N}) : \omega + a = \omega - a = \omega$. If $p \leq q$ and $p \neq q$ we say that *q covers p*. The *cardinality* of a set A is denoted as $|A|$.

If some object x has both an "ordinary" and a vector form, we will denote the ordinary form as x and the vector form as $\vec{x}$. The letter "T" in upper index always stands for matrix (vector) transposition. For matrices we use the standard notation, i.e. a matrix A with m rows

and n columns is defined as $A = (A_{i,j})_{m \times n}$ and $A_{i,j}$ is the entry in the i-th row and j-th column of A. $0_{m,n}$ is the *zero matrix* of dimensions $m \times n$ and 0^k is the k-dimensional *zero vector*. If v is a sequence of symbols from V (i.e. a string $v \in V^*$) and $a \in V$ then by $v(a)$ we mean the number of occurrences of a in v.

P/T nets represent one of the most researched and well-known version of Petri nets, a formal language able to specify non-deterministic and concurrent systems. A P/T net can be defined as follows:

Definition 1 ((Pure) P/T net). A *P/T net* is a 5-tuple $N_0 = (P, T, pre, post, m_0)$, where $P = \{p_1, \ldots, p_k\}$ is a finite set of *places*, $T = \{t_1, \ldots, t_n\}$ is a finite set of *transitions*, $pre : P \times T \to \mathbb{N}$ is a *preset function*, $post : P \times T \to \mathbb{N}$ is a *postset function* and $m_0 : P \to \mathbb{N}$ is the *initial marking*. A *pure* P/T net is a P/T net for which it holds that $\neg\exists(p \in P, t \in T) : pre(p,t) > 0 \wedge post(p,t) > 0$.

The functions pre and $post$ have the same purpose as the flow relation and weight functions in the P/T net definition from [1] - to define arcs in P/T net graph. If $pre(p,t) = x$ and $x > 0$ then there is an arc from a place $p \in P$ to a transition $t \in T$ with a weight x. The function $post$ defines arcs from transitions to places in the similar way.

A *marking* of P/T net N_0 is a function $m : P \to \mathbb{N}$. Value of *m(p)* is the number of tokens in the place *p*. Markings can be written as vectors, i.e.

$$\vec{m} = (m(p_1), \ldots, m(p_k))$$

and we distinguish between different markings by using lower and upper indices (m_1, m', etc.). The *initial marking* m_0 is a marking assigned to the net when initialised.

The functions *pre* and *post* can be represented as matrices, namely as

- the *pre-matrix* Pre, $Pre = (Pre_{i,j})_{k \times n}$, $Pre_{i,j} = pre(p_i, t_j)$,
- the *post-matrix* $Post$, $Post = (Post_{i,j})_{k \times n}$, $Post_{i,j} = post(p_i, t_j)$ and
- the *incidence matrix* C, $C = (C_{i,j})_{k \times n}$, $C = Post - Pre$.

Then for each $t_j \in T$ we can define its vector form $\vec{t}_j = (C_{1,j}, \ldots, C_{k,j})$ and related vectors $\vec{t}_j^{\,pre} = (Pre_{1,j}, \ldots, Pre_{k,j})$ and $\vec{t}_j^{\,post} = (Post_{1,j}, \ldots, Post_{k,j})$.

A transition $t_j \in T$ is *enabled* (*feasible*) in a marking m (denoted $m \xrightarrow{t_j}$), if and only if $\vec{m} - \vec{t}_j^{\,pre} \geq 0^k$. When t_j is enabled, it can be *executed* (*fired*). The result of its execution is a new marking m', $\vec{m}' = \vec{m} + \vec{t}_j$. We say that m' is *reachable* from m via t_j and denote it $m \xrightarrow{t_j} m'$. A sequence of transitions $\sigma = t_{j1} t_{j2} .. t_{jr}$ is an *admissible firing sequence* in PN N_0, provided a sequence of markings $m_0, m_1, \ldots, m_r$ exists such that $\forall i (1 \leq i \leq r) : m_{i-1} \xrightarrow{t_{ji}} m_i$. In that case we write

$m_0 \xrightarrow{\sigma} m_r$ or just $m_0 \xrightarrow{*} m_r$ if σ is immaterial. The marking m_r is called a *reachable marking* in N_0 from m_0 (via σ). The *set of reachable markings* of PN N_0 is the set $R(N_0) = \{m \mid m_0 \xrightarrow{*} m\}$. Given a P/T net N_0 and any $q \in \mathsf{N}^k$ a problem whether $q \in R(N_0)$ is called (an instance of) *the reachability problem* of N_0 (with respect to q). We will denote it $RP(q, N_0)$.

The marking m_r can be computed as $\vec{m}_r = \vec{m}_0 + \Psi(\sigma) \cdot C^T$, where $\Psi(\sigma) = (\sigma(t_1), \ldots, \sigma(t_n))$ is the *Parikh mapping* of σ. For any $\sigma \in T^*$ we define $[\sigma]$ as a column vector $[\sigma] = C \cdot \Psi^T(\sigma)$.

The reachability algorithm in [3] is defined for vector addition systems (VAS), which are equivalent to the pure P/T nets. In [3] a k-dimensional VAS has been defined as $W_k = (q_0, W)$, where $q_0 \in \mathsf{N}^k$ is the initial state of W_k and W is a finite set of k- dimensional integer vectors. Given $q, q' \in \mathsf{N}^k$ and $w \in W$, the enabling and execution for VAS can be defined as $q \xrightarrow{w} \Leftrightarrow_{def} q + w \geq 0^k$ and $q \xrightarrow{w} q' \Leftrightarrow_{def} q' = q + w$. The VAS equivalent to N_0 has the form $W_k(N_0) = (\vec{m}_0, \{\vec{t}_1, \ldots, \vec{t}_n\})$.

The requirement that a net has to be pure is only a technical obstacle: we can eliminate self-lops by inserting a dummy place-transition pair into each of them. Because of this we will refer to the *RP/M*$_w$ algorithm as to a reachability algorithm for P/T nets and not for VAS.

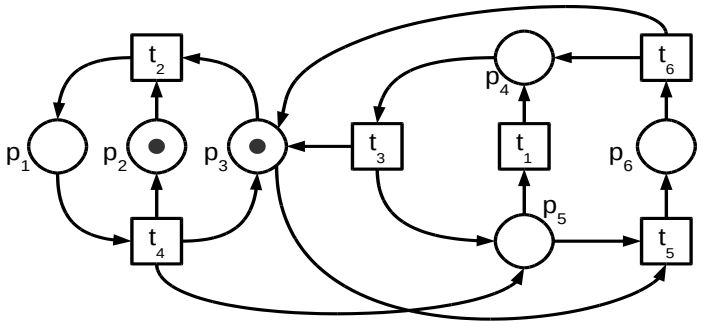

Figure 1. Graph of unbounded P/T net

Figure 1 shows an example of unbounded P/T net. The initial marking of the net is $\vec{m}_0 = (0,1,1,0,0,0)$ and some of the vectors are $\vec{t}_1 = (0,0,0,1,-1,0)$, $\vec{t}_2 = (1,-1,-1,0,0,0)$, $\vec{t}_3^{\,post} = (0,0,1,0,1,0)$, $\vec{t}_5^{\,pre} = (0,0,1,0,1,0)$.

3. Reachability Problem Algorithm

The *RP/M*$_w$ algorithm for an instance $RP(q, N_0)$ can be formulated in six steps:

Step 1. For N_0 construct the finite state automaton (fsa) $M_w(N_0)$. If states of $M_w(N_0)$ are ω-vectors (i.e. vectors containing at least one ω-coordinate), go to Step 3. Otherwise go to Step 2. This step is demonstrated in section 3.1.

Step 2. In M_w search for a state equal to q. If there is such state, go to Step 5, otherwise go to Step 6.

Step 3. In M_w search for a state covering q. If there is such state, go to Step 4, otherwise go to Step 6.

Step 4. Formulate and solve the modified integer linear programming problem $MILP_{N_0}(A, X, B(q))$. If the result is *true* go to Step 5, otherwise go to Step 6. This step is in more detail treated in section 3.2.

Step 5. Answer $q \in R(N_0)$. Stop.

Step 6. Answer $q \notin R(N_0)$. Stop.

The fsa $M_w(N_0)$, created in the first step, represents the state space of N_0. It is defined as $M_w(N_0) = (Q, W, \delta, \rho_0)$ where Q is a set of states, W is an input alphabet, $W = T$, $Q \times W \to Q$ is a transition function and $\rho_0 \in Q$ is the initial state. Each $q \in Q$ has a vector form $\vec{q} \in (\mathbb{N} \cup \{\omega\})^k$. According to the boundedness of N_0, $M_w(N_0)$ can fall into one of the following two classes:

- **RG-like.** If N_0 is bounded then $M_w(N_0)$ is identical to its reachability graph, $Q = R(N_0)$ and $\vec{m}_0 = \vec{\rho}_0$.
- **CG-like.** If N_0 is unbounded then $M_w(N_0)$ is similar but not identical to its coverablity graph. The most significant difference is that there is no extra state for m_0 in M_w. In CG-like M_w each $\rho \in Q$ has a form of an ω-vector and represents an infinite subspace of $R(N_0)$, namely all $m \in R(N_0)$ such that $\vec{m} \le \vec{\rho}$. This implies $\vec{m}_0 \le \vec{\rho}_0$. The states of $M_w(N_0)$ are also called *macrostates* and are mutually incomparable.

3.1. $\mathbf{M_w}$ Construction

In [3] the creation of $M_w(N_0)$ is defined by a sequence of transformations $T^{\omega}_{<x}$ of so-called vector state tree (*VST*) of VAS $W_k(N_0)$, which is identical to the reachability tree of N_0. Any state q in VST is assigned the language $Y_q = \{u \in T^* \mid \exists q' \in \mathbb{N}^k : q \xrightarrow{u} q'\}$, that is called the *suffix language of* q. The transformation $T^{\omega}_{<x}$ replaces an infinite path consisting of subpaths $q \xrightarrow{\pi} q'$, $\pi \in T^*$, such that $q \le q'$, by the loop $\rho \xrightarrow{\pi} \rho$, $\rho = T^{\omega}_{<x}(q) = T^{\omega}_{<x}(q')$, $\rho = \omega_A q$. By $\omega_A q$ we denote a vector obtained from q by replacing all the coordinates listed in A by ω. A, $A \subseteq \{1 \ldots k\}$, is a set of indices in which the strong inequality between q and q' exists. $T^{\omega}_{<x}$ has the invariant property $\forall q'', q \xrightarrow{*} q'' : Y_{q''} \subseteq Y_{T^{\omega}_{<x}(q'')}$. The sequence of transformations is always finite. The theory of M_w construction can be found in [3], here we demonstrate it on an example.

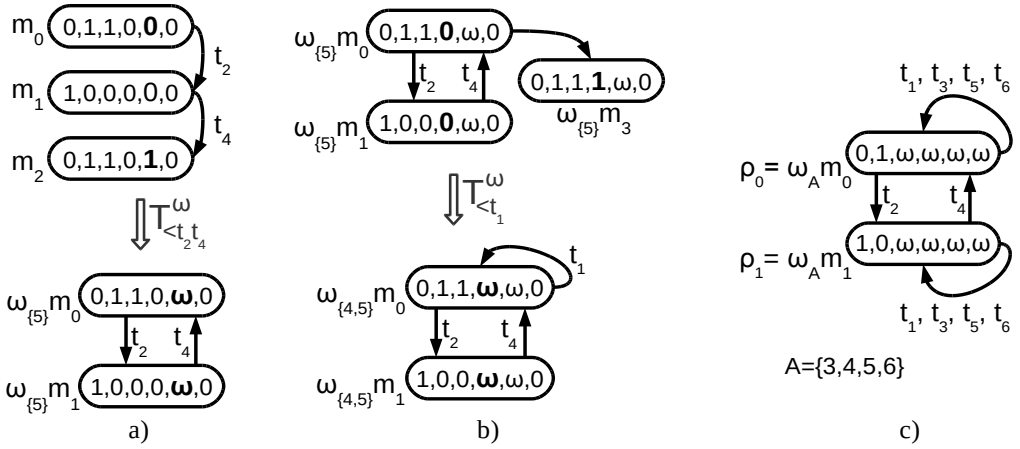

Figure 2: M_w **automaton construction for the net from Fig. 1: step one (a), two (b) and the final result (c).**

Example 1. In practice the construction of M_w automaton proceeds in such a way that we apply transformations as soon as possible. So, in the case of the net from Fig. 1 we apply $T^{\omega}_{<t_2 t_4}$ immediately after finding out that $m_0 \le m_2$. The transformation "omegalizes" the fifth coordinate of the vectors on the corresponding path and create the loop t_2, t_4 (Fig. 2 a). In the second step a new ω replaces fourth coordinate of $\omega_{\{5\}}m_0$ and $\omega_{\{5\}}m_3$, but this ω also propagates to $\omega_{\{5\}}m_1$ (Fig. 2 b). The rest of the transformations consists of steps similar to the second one for t_3, t_5 and t_6 and the resulting M_w automaton is shown in Fig. 2 c).

3.2. Modified ILP Problem

If $M_w(N_0)$ is CG-like then the structure of its state diagram consists of $n \ge 1$ *strongly connected components*[1] (*scc*) and a corresponding RP instance $RP(q, N_0)$ transforms to an instance of a problem[2] we call *modified integer linear programming problem* (*MILP*)

$$MILP_{N_0}(A, X, B(q)) \equiv ILP_{N_0}(A, X, B(q)) \wedge con_{N_0}(A, X, B(q)) \qquad (1)$$

Its first part is an ILP instance and is constructed in the following way: Assume that we found a state ρ of M_w such that $q \le \rho$ and ρ is in the same scc, scc_{ρ_0}, as ρ_0. Also assume that $v, v \in T^*$ is the only path from ρ_0 to ρ. Now we have to find all distinct simple loops in scc_{ρ_0}. By a *simple loop* we mean a non-loop cycle in the graph of M_w. Let's say that the simple loops of scc_{ρ_0} form the set $W_L = \{\ell_1, \ell_2, \ldots, \ell_{d_0}\}$, $\forall i (1 \le i \le d_0) : \ell_i \in T^*$. Then we can formulate the system of linear equations (2).

$$AX = B(q), \quad B(q) = q^T - \vec{m}_0{}^T - [v], \quad A = ([\ell_1], [\ell_2] \cdots [\ell_{d_0}]) \qquad (2)$$

[1] scc of M_w reminds marked reachability graphs from [8] and the structure of M_w is a case of KLMST decomposition.

[2] In [3] the MILP (1) is defined as $MILP_{W_k}(A, X, B(q)) \equiv ILP_{W_k}(A, X_0, B(q)) \wedge con_{W_k}(A, X_0, B_0)$. Modifications have been made to get rid of notation not used in this paper. Actually, W_k is $W_k(N_0)$, X_0 is related to minimal solutions of (2) and $B_0 = B(q) - q$.

Solving the ILP instance means solving (2). If there is a solution X' to (2) that is a non-negative integer vector, then $ILP_{N_0}(A, X, B(q)) = true$.

The ILP solution informs us about the number of loops passed on the way from m_0 to q, but not about their ordering. Therefore an additional test, expressed in the predicate $con_{N_0}(A, X, B(q))$, has been defined. The test is about checking whether there is an admissible firing sequence in N_0 in which number of occurrences of each transition is given by the solution vector X' and the structure of simple loops in scc_{ρ_0}. There is no need to compute con_{N_0} if the reachability set of N_0 is semilinear and results provided in [3] indicate that the class of PN for which it is satisfactory to solve ILP can be even wider, however they require addition examination. Nevertheless, the need to compute con re-occurs when using the T-junction de/compositional approach, as it is described in the next section.

If $M_w(N_0)$ has more than one scc and q is covered by a state ρ that is not in the initial scc_{ρ_0}, we proceed in the similar way but the set W_L and the matrix A in (2) have to include loops from all scc-s on the way from ρ_0 to ρ. It should be also noted that if $q \le \rho$ and $\rho \neq \rho_0$ then there can be more than one path from ρ_0 to ρ. This means more ILP instances to be solved.

Example 2. Let's say that we would like to know whether $q = (0,1,1,1,0,1)$ is a reachable marking of the net N_0 from Fig. 1. After constructing the corresponding M_w (Fig. 2 c) we know that $q \le \rho_0$. The whole $M_w(N_0)$ is one scc with $W_L = \{\ell_1, \ell_2, \ell_3, \ell_5, \ell_6\}$, $\ell_2 = t_2, t_4$ and $\ell_i = t_i$ for $i = 1,3,5,6$. This yields the equation system

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 1 \\ 1 & 0 & -1 & 0 & 1 \\ -1 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_5 \\ X_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

and its solution is $\vec{X} = (2,2,1,1,0)^T$, so $ILP_{N_0} = true$ and $q \in R(N_0)$. The $con_{N_0} = true$, too as there is an admissible sequence $t_2, t_4, t_1, t_2, t_3, t_1, t_4, t_5$ from m_0 to q.

4. RP Solving with T-junction De/composition

Provided that N_0 is as in Definition 1, by T-junction we can split N_0 into subnets N_1, N_2 such that ($z \in \{1,2\}$):

$$\begin{aligned} &N_z = (P_z, T_z, pre_z, post_z, m_{0_z}), \\ &P_1 \cup P_2 = P,\ P_1 \cap P_2 = \varnothing,\ |P_1| = k_1,\ |P_2| = k_2, \\ &T_1 \cup T_2 = T,\ T_1 \cap T_2 = T_s \neq \varnothing, \\ &pre_z = (P_z \times T_z) \lhd pre,\ post_z = (P_z \times T_z) \lhd post, \\ &m_{0_z} = P_z \lhd m_0 \end{aligned} \qquad (3)$$

and we denote this situation as $N_0=TJUNC[T_s(N_1,N_2)]$. The symbol "$\triangleleft$" stands for the domain restriction. The common transitions form the set T_s, which is called the *set of synchronic transitions*.

The relation between solutions of RP for N_1, N_2 and N_0 is formulated in Theorem 1, which is a slightly reformulated version of the similar one from [2]. Here $\sigma|_{T_s}$ is the projection of σ onto T_s, i.e. we obtain $\sigma|_{T_s}$ from σ by dropping away all transitions that don't belong to T_s from σ.

Theorem 1. Let N_0, N_1, N_2 be P/T nets such that $N_0=TJUNC[T_s(N_1,N_2)]$. Then for any $q \in \mathbb{N}^k$ it holds that

$$\begin{aligned} q \in R(N_0) \Leftrightarrow \quad & \exists (q_1,q_2): q_1 \in \mathbb{N}^{k_1} \wedge q_2 \in \mathbb{N}^{k_2} \wedge q = (q_1,q_2) \wedge \\ & q_1 \in R(N_1) \wedge q_2 \in R(N_2) \wedge \\ & \exists (\sigma \in T^*, \sigma_1 \in T_1^*, \sigma_2 \in T_2^*): \sigma_1 = \sigma|_{T_1} \wedge \sigma_2 = \sigma|_{T_2} \wedge \\ & m_{0_1} \xrightarrow{\sigma_1} q_1 \ in \ N_1 \wedge m_{0_2} \xrightarrow{\sigma_2} q_2 \ in \ N_2 \wedge \sigma_1|_{T_s} = \sigma_2|_{T_s} . \end{aligned} \tag{4}$$

In what follows we formulate and prove a new theorem that brings the result of Theorem 1 closer to an actual implementation of a RP algorithm utilizing T-junction. Before doing this we need to specify the sets of places and transitions from (3) in more detail:

$$\begin{aligned} & P = \{p_1,\ldots,p_a,p_{a+1},\ldots,p_k\}, T = \{t_1,\ldots,t_b,\ldots,t_c,\ldots,t_n\} \\ & P_1 = \{p_1,\ldots,p_a\}, P_2 = \{p_{a+1},\ldots,p_k\} \\ & T_1 = \{t_1,\ldots,t_c\}, T_2 = \{t_b,\ldots,t_n\}, T_s = \{t_b,\ldots,t_c\} \end{aligned}$$

For the sake of simplicity we assume that solving RP in N_1 and N_2 yields to only one ILP instance in each of them.

Theorem 2. Let N_0, N_1, N_2 be as in Theorem 1 and $q \in \mathbb{N}^k$, $q_1 \in \mathbb{N}^{k_1}$ and $q_2 \in \mathbb{N}^{k_2}$ such that $q = (q_1,q_2)$. Moreover, let

$$\begin{aligned} & A_z X_z = B(q_z), B(q_z) = q_z^T - m_{0_z}^T - [v_z], \\ & W_{L_z} = \{\ell_{z.1},\ldots,\ell_{z.d_z}\}, A_z = ([\ell_{z.1}],\ldots[\ell_{z.d_z}]) \\ & X_z = (x_{z.1},\ldots,x_{z,d_z})^T \end{aligned} \tag{5}$$

be the IPL instances $ILP_{N_z}(A_z, X_z, B(q_z))$ for $RP(q_z, N_z)$ and

$$A'_z X'_z = B'(q_z) \tag{6}$$

be their explicit solutions, $z = 1,2$. Then it holds that the equation system (7)

$$\begin{pmatrix} A'_1 & 0_{a,d_2} \\ 0_{k-a,d_1} & A'_2 \\ L_1 & -L_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} B'(q_1) \\ B'(q_2) \\ B_{1.2} \end{pmatrix} \tag{7}$$

where

$$L_z = (L_{z_{i,j}})_{(c-b+1)\times d_z},\ L_{z_{i,j}} = \ell_{z.j}(t_{b+i-1}),\ z = 1,2$$
$$B_{1.2} = (B_{1,2_1},\ldots,B_{1,2_{b-c+1}})^T,\ B_{1,2_i} = v_2(t_{b+i-1}) - v_1(t_{b+i-1})$$

has a non-negative integer solution if and only if

$$\exists(\sigma_1 \in T_1^*, \sigma_2 \in T_2^*) : (ILP_{N_1}(A_1, X_1, B(q_1)) = true \wedge ILP_{N_2}(A_2, X_2, B(q_2)) = true \wedge \forall(t \in T_s) : \sigma_1(t) = \sigma_2(t)). \quad (8)$$

Proof of Theorem 2. The equation systems (6) are the explicit solutions of (5), so they are in the row echelon form, i.e. obtained from (5) by Gaussian elimination. They are equivalent to (5) and of the same dimension. Therefore, it is sufficient to prove the theorem for (9) instead of (7).

$$\begin{pmatrix} A_1 & 0_{a,d_2} \\ 0_{k-a,d_1} & A_2 \\ L_1 & -L_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} B(q_1) \\ B(q_2) \\ B_{1.2} \end{pmatrix} \quad (9)$$

The system (9) consists of three sets of equations

$$A_1 X_1 = B(q_1) \quad (10)$$
$$A_2 X_2 = B(q_2) \quad (11)$$
$$\begin{pmatrix} L_1 & -L_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = B_{1.2} \quad (12)$$

The equations (10) and (11) are the ILP instances $ILP_{N_1}(A_1, X_1, B(q_1))$ and $ILP_{N_2}(A_2, X_2, B(q_2))$, so we are left to prove that (12) corresponds to the condition $\forall(t \in T_s) : \sigma_1(t) = \sigma_2(t)$. Let us consider t_{b+i-1} the i-th member of $T_s = \{t_b,\ldots,t_c\}$. In σ_z the transition t_{b+i-1} can be a part of the initial path v_z or of one or more loops from W_{L_z}. While v_z occurs only once in σ_z, the loops can be repeated many times. And the exact number of their repetitions is given by X_z. So, we can write

$$\sigma_z(t_{b+i-1}) = v_z(t_{b+i-1}) + \sum_{j=1}^{d_z} \ell_{z.j}(t_{b+i-1}) \cdot x_{z.j} \quad (13)$$

After applying (13) to the equation $\sigma_1(t_{b+i-1}) = \sigma_2(t_{b+i-1}))$ and some subsequent modifications we get

$$\sum_{j=1}^{d_1} \ell_{1.j}(t_{b+i-1}) \cdot x_{1.j} - \sum_{j=1}^{d_2} \ell_{2.j}(t_{b+i-1}) \cdot x_{2.j} = v_2(t_{b+i-1}) - v_1(t_{b+i-1})$$

which is exactly in the form of the i-th equation from (12) and we are through.

End of proof.

4.1. RP/Tjunc/M_w Algorithm

On the basis of theorems 1 and 2 we can now formulate the following algorithm to solve $RP(q, N_0)$ using T-junction de/composition $N_0=TJUNC[T_s(N_1,N_2)]$:

Step 1. Solve the ILP instances (5) for $RP(q_1, N_1)$ and $RP(q_2, N_2)$ where $q = (q_1, q_2)$, $q_1 \in \mathbb{N}^{k_1}$ and $q_2 \in \mathbb{N}^{k_2}$. If both of them have non-negative integer (nni) solutions, go to Step 2, otherwise go to Step 7.

Step 2. Solve the ILP instance (7). Based on the solution of (7) there are three possible cases:
1. There is no nni solution of (7). Then go to Step 7.
2. There is exactly one nni solution. Then assign the solution to a new vector X, $X \in \mathbb{N}^{d_1+d_2}$ and go to Step 4.
3. There are infinitely many nni solutions, forming an infinite poset (partially ordered set) $Xi_n f$ of natural number vectors. Then put all minimal elements of $Xi_n f$ to a new set $X0$ and go to Step 3.

Step 3. If the set $X0$ doesn't exist or is empty, go to Step 7. Else remove the first element from $X0$, assign it to a new vector X, $X \in \mathbb{N}^{d_1+d_2}$ and go to Step 4.

Step 4. Let X have a form $X = (x_{1.1},\ldots,x_{1.d_1},x_{2.1},\ldots,x_{2.d_2})$. Compute Φ_1, Φ_2 according to (14). Discard X. Go to Step 5.

$$\begin{aligned} \Phi_1 &= (\varphi_1(t_1),\ldots,\varphi_1(t_b),\ldots,\varphi_1(t_c)) \\ \Phi_2 &= (\varphi_2(t_b),\ldots,\varphi_2(t_c),\ldots,\varphi_2(t_n)) \\ \varphi_z(t_i) &= v_z(t_i)+\sum_{j=1}^{d_z}\ell_{z.j}(t_i)\cdot x_{z.j},\ i=1\ldots n,\ z=1,2 \end{aligned} \tag{14}$$

Step 5. For each pair of sequences (σ_1,σ_2) such that $\sigma_z \in T_z^*$, $\Psi(\sigma_z)=\Phi_z$, $z=1,2$, $\sigma_1|_{T_s}=\sigma_2|_{T_s}$ try their admissibility in N_1 and N_2 respectively. If such (σ_1,σ_2) is found that $m_{0_1}\xrightarrow{\sigma_1}q_1$ in N_1 and $m_{0_2}\xrightarrow{\sigma_2}q_2$ in N_2, stop trying and go to Step 6. If no such pair is found after trying all possible (σ_1,σ_2), go to Step 3.

Step 6. Answer $q \in R(N_0)$. Stop.

Step 7. Answer $q \notin R(N_0)$. Stop.

Steps 1 and 2 are a direct application of the result of Theorem 2. Steps 3 to 5 are about a systematic search for a pair of admissible firing sequences that synchronize on transitions from T_s. That is sequences in which not only the number (this is already checked in Step 2) but also the order of transitions from T_s match. In fact, this means checking con_{N_1}, con_{N_2} and also whether subsequences consisting of transitions from T_s are identical. Step 3 shows us that this search is always finite, even if the system (7) has infinitely many solutions: Because this

infinity is caused by groups of loops that allow us to re-enter the same marking again and again, it is sufficient to search only for sequences that enter the marking only once, i.e. for sequences built according to minimal solutions $X0$ of (7) and $X0$ is always finite. (this situation is similar to checking the condition con_{N_0} (con_{W_k}) in the original RP/Mw algorithm [3]). The value $\varphi_z(t_i)$ from (14) is the number of occurrences of t_i in a firing sequence satisfying given solution X of (7).

We illustrate an actual use of the RP/Tjunc/Mw algorithm on a small example.

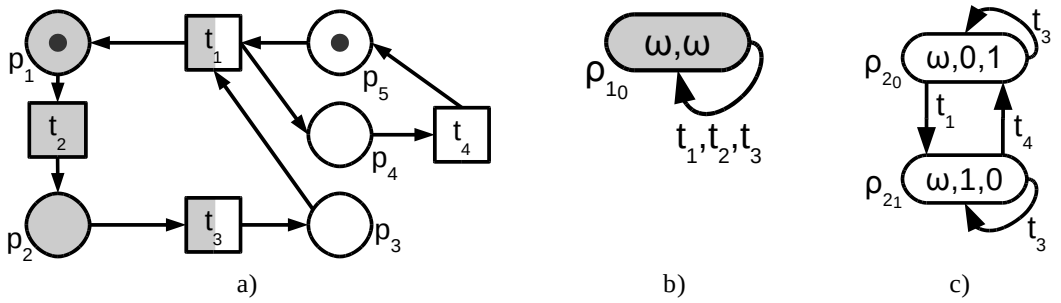

Figure 3. P/T net N_0 (a) and M_w automata of its subnets N_1 (b) and N_2 (c). Grey places and transitions belong to N_1, white to N_2, grey/white to both. $T_s=\{t_1,t_3\}$.

Example 3. Consider the net N_0 in Fig. 3 a). We would like to know whether the vector $q=(0,1,0,1,0)$ is a reachable marking of N_0. We divide N_0 into N_1, N_2 such that

$$P_1=\{p_1,p_2\}, T_1=\{t_1,t_2,t_3\}, \vec{m}_{0_1}=(1,0), q_1=(0,1)$$
$$P_2=\{p_3,p_4,p_5\}, T_2=\{t_1,t_3,t_4\}, \vec{m}_{0_2}(0,0,1), q_2=(0,1,0)$$

and $T_s=\{t_1,t_3\}$. The M_w automata of the subnets are in Fig. 3 b) and c) and their simple loops can be characterized as in Fig. 4 a). We have $v_1=\varepsilon$ (empty string) and $v_2=t_1$. The equation systems for $RP(q_1,N_1)$ and $RP(q_2,N_2)$ are already explicit solutions (i.e. they are in the row echelon form), so we can construct the system of type (7) (Fig. 4 b). The system has infinitely many nni solutions $Xi_n f=\{(r,r+1,r,r,r-1)\mid r\in \mathrm{N}-\{0\}\}$ and $X0=\{(1,2,1,1,0)\}$. After checking pairs of admissible sequences we find $\sigma_1=t_2t_3t_1t_2$, $\sigma_2=t_3t_1$, $\sigma_1|_{T_s}=\sigma_2|_{T_s}=t_3t_1$ and we conclude that $q\in R(N_0)$.

d	$\ell_{1.d}$	$[\ell_{1.d}]^T$	$\ell_{1.d}(t_1)$	$\ell_{1.d}(t_3)$
1	t_1	(1,0)	1	0
2	t_2	(–1,1)	0	0
3	t_3	(0, –1)	0	1

d	$\ell_{1.d}$	$[\ell_{1.d}]^T$	$\ell_{1.d}(t_1)$	$\ell_{1.d}(t_3)$
1	t_3	(1,0,0)	0	1
2	t_1t_4	(–1,0,0)	1	0

a)

$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} x_{1.1} \\ x_{1.2} \\ x_{1.3} \\ x_{2.1} \\ x_{2.2} \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

b)

Figure 4. Loops (a) and ILP instance (b) for example 3.

The example shows us that the RP/Tjunc/M_w algorithm can be used also for finite P/T nets, however it should be thoroughly investigated under which circumstances it is more effective than traversing large state spaces. Regarding the net in Fig. 3 a) it is also interesting that even after some serious trying we were unable to find a case in which the equation system (7) has some nni solution and it is impossible to find σ_1, σ_2 such that $\sigma_1|_{T_s} \neq \sigma_2|_{T_s}$. We observed the similar features in different P/T nets, too. This promises the possibility to exclude steps 3-5 from RP/Tjunc/M_w for some subclasses of P/T nets.

Before formulating Theorem 2 we assumed that solving RP in N_1 and N_2 yields to only one ILP instance in each of them. If we have more instances then we need to run the RP/Tjunc/M_w algorithm for each pair of instances where the first is for N_1 and the second for N_2, until we get the answer $q \in R(N_0)$ or try all of them.

4.2. Extendibility and Parallelisation Possibilities

The RP/Tjunc/M_w algorithm can be easily extended for a T-junction decomposition with more than two subnets, provided that their sets of synchronic transitions are mutually disjoint. For example, the de/composition to three subnets $N_z = (P_z, T_z, pre_z, post_z, m_{0_z})$, $z = 1,2,3$, denoted here as $N_0 = TJUNC[T_{s_{1.2}}(N_1, N_2), T_{s_{2.3}}(N_2, N_3)]$, is defined similarly to (3) with the exception of the sets of synchronic transitions. Here $T_1 \cap T_2 = T_{s_{1.2}}, T_2 \cap T_3 = T_{s_{2.3}}, T_1 \cap T_3 = \varnothing,\ T_{s_{1.2}} \cap T_{s_{2.3}} = \varnothing$. Then instead of the equation system (7) we have

$$\begin{pmatrix} A_1' & 0_{|P_1|,d_2} & 0_{|P_1|,d_3} \\ 0_{|P_2|,d_1} & A_2' & 0_{|P_2|,d_3} \\ 0_{|P_3|,d_1} & 0_{|P_3|,d_2} & A_3' \\ L_1 & -L_2 & 0_{|T_{s_{1,2}}|,d_3} \\ 0_{|T_{s_{2,3}}|,d_3} & L_2 & -L_3 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} = \begin{pmatrix} B'(q_1) \\ B'(q_2) \\ B'(q_3) \\ B_{1.2} \\ B_{2.3} \end{pmatrix}$$

where all the elements are defined similarly to (7). The steps 3 to 5 of the RP/Tjunc/M_w algorithm have to be altered to incorporate checks for both $T_{1,2}$ and $T_{2,3}$.

The algorithm allows a high level of parallelisation with little synchronization: In Step 1 each ILP instance can be solved as a separate task. In Step 2 we can parallelize, too: Before solving the whole equation system we can reduce to the row echelon form the set of equations for each $T_{s_{i,j}}$ (i.e that involving L_i, L_j and $B_{i.j}$). Steps 3 and 4 can be divided into as many parallel tasks as the number of sets of synchronic transitions in a given decomposition is. In Step 5 virtually every single pair (σ_i, σ_j) can be checked in a separated task, but a more practical approach will be to check within one task a set of pairs

$$\sigma_{ij}^{\delta} = \{(\sigma_a, \sigma_b) \mid \sigma_a \in T_i, \sigma_b \in T_j, \sigma_a|_{T_{s_{i,j}}} = \sigma_b|_{T_{s_{i,j}}} = \delta\}.$$

In addition, we can check multiple pairs at once, utilizing GPGPU (general-purpose computing on graphics processing units).

5. Conclusions and Related Work

In this paper we presented a new algorithm, called RP/Tjunc/M_w algorithm, that combines the original RP/M_w algorithm with the T-junction de/compositional approach. The RP/Tjunc/M_w algorithm has been defined for two subnets, but it was also shown how it can be extended to more nets. The algorithm is very suitable for parallelisation, which makes it ideal for modern-day cloud computing and grid environments. It is also possible to utilise GP/GPU for some tasks of the algorithm.

The results presented here are most related to works of G. Tonţ [12] and H. Yen [13], [14]. The work [12] is also based of results from [2], [3] and presents an algorithm for PT-junction decomposition into two nets. The decomposition strategy is based on structural properties of given net, derived from its incidence matrix. The algorithm from [12] can be, after modification, used for the decomposition itself, before applying our one. In [13] and subsequent work, such as [14], a decompositional technique for subclasses of P/T nets with semilinear reachability sets is proposed. The decomposition used in [13] can be regarded as a special case of PT-junction with the property that each admissible firing sequence can be divided into subsequences where *i*-th subsequence consists only from transitions of *i*-th subnet. The technique is not usable in general, but doesn't require the search for concrete admissible firing sequences and RP instances can be decided using ILP.

The RP/M_w and RP/Tjunc/M_w algorithms presented here are partially implemented in the experimental mFDTE/PNtool2 software, which is available from http://fm.fei.tuke.sk/. The recent version of mFDTE/PNtool2 implements the M_w automaton construction and the IPL solving parts of the RP/M_w algorithm. The tool also allows to store M_w automata created for reuse with other instances of RP for given nets. For RP/Tjunc/M_w it provides several strategies of decomposition to subnets and implements its first step. The remaining steps are about to be implemented. We plan implementations for a single computer with and without GP/GPU support and implementation for grid and cluster environments. The future research should also address some open theoretical issues. The first one is a complexity evaluation of the algorithm. The second one is its formalisation for more than two subnets without the requirement of mutually disjoint sets of synchronic transitions. The conditions, under which the property observed during experiments with various nets, namely that there was no need to check synchronisation of firing sequences for given subnets after the solution of (7) is obtained, will also be examined. This will be done primarily with respect of those classes of Petri nets where solving the ILP instance is clearly sufficient to decide about reachability. Results achieved in works of H. Yen [13], [14] can be helpful here.

Acknowledgements. This work has been supported by KEGA grant project No. 050TUKE-4/2012: "Application of Virtual Reality Technologies in Teaching Formal Methods".

References

[1] J. Desel and W. Reisig: "Place/Transition Petri Nets," In Reisig, W., Rozenberg, G. (eds.) Petri Nets. LNCS, vol. 1491, Springer, Heidelberg, 1998, pp. 122-173.

[2] Š. Hudák: "De/compositional Reachability Analysis," Journal of Electrical Engineering, CS ISSN 0013-578X. 45, 1994, pp. 424-431.

[3] Š. Hudák: "Reachability Analysis of Systems Based on Petri Nets," elfa s.r.o. Košice, ISBN 80-88964-07-5, 1999, available from http://fm.fei.tuke.sk/

[4] Š. Hudák: "Verification of Systems: Deadlock Analysis Based on Petri Nets," In Proc. of Workshop on Algebraic, Logical, and Algorithmic Methods of System Modeling, Specification and Verification, ICTERI 2012, pp.321-343, http://ceur-ws.org/Vol-848/

[5] R. Johnsonbaugh, T. Murata: "Petri Nets and Marked Graphs-Mathematical Models of Concurrent Computation," The American Mathematical Monthly, vol. 89, no. 8, 1982, pp. 552-566

[6] S.R. Kosaraju: "Decidability of reachability in vector addition systems," In Proc. 14th Annual ACM STOC, 1982, pp. 267-281

[7] A.E. Kostin: "A Reachability Algorithm for General Petri Nets Based on Transition Invariants," In Mathematical Foundations of Computer Science, LNCS, vol. 4162, Springer-Verlag, 2006, pp.608-621

[8] J. Leroux: "The general vector addition system reachability problem by presburger inductive invariants," Logical Methods in Computer Science, Vol. 6 (3:22), 2010, pp. 1-25, http://www.lmcs-online.org

[9] J. Leroux: "Vector Addition Systems Reachability Problem (A Simpler Solution)," Turing-100. The Alan Turing Centenary, 2012, pp. 214-228

[10] E. Mayr. "An algorithm for the general Petri net reachability problem," In Proc. 13th Annual ACM STOC, 1981, pp. 238-246

[11] T. Murata: Petri Nets: "Properties, Analysis and Applications," Proceedings of the IEEE, vol.77, no.4, 1989, pp.541-580

[12] G. Tonț: "The Algorithmic Procedure of Petri Net Decomposition," Acta Electrotehnica, vol. 46, no. 4, 2005, pp. 173-177

[13] H. Yen, "Introduction to Petri Net Theory," In Z. Esik, C. Martin-Vide, V. Mitrana, (Eds.) Recent Advances in Formal Languages and Applications, Studies in Computational Intelligence 25, Springer, 2006, pp. 343-373

[14] H. Yen, "Path Decomposition and Semilinearity of Petri Nets," Int. J. Found. Comput. Sci., vol.20, no.4, 2009, pp. 581-596