EFFICIENT ALGORITHMS FOR PRE* ON INTERPROCEDURAL PARALLEL FLOW GRAPH

Yutaka Nagashima

Technische Universität München
Computational Science and Engineering
Boltzmannstr. 3, Garching bei Munich, BY Germany
yutaka.nagashima@mytum.de

ABSTRACT

keywords: verification, concurrent system, process algebra, transition system, pre, post, interprocedural parallel flow graph, tree automata, Horn clause, Dawling-Gallier procedure, term rewriting.

The time complexity of model-checking algorithms depends on the size of the transition system, where the size of a transition system is defined as the sum of the number of states and the number of transition rules. This paper explains an efficient algorithm for computing pre* of interprocedural parallel flow graphs. [1]

1. INTRODUCTION

1.1. Structure of the paper

In this paper we show the reader the algorithm for pre* on interprocedural parallel flow graphs that was originally described by Javier E., et al. [1]. The definition of $\mathrm{pre}^*(L)$ is given in Section 3.3 of this paper.

We begin in Section 2 by describing how parallel flow graphs represent concurrent programs.

In section 3, we present the concept of the process algebra, and we explain how to derive the PA-declaration Δ from parallel flow graphs. Furthermore, a transition relation system with 5 inference rules on Δ is presented.

In section 4, we give a description about tree automata that play an important role in our algorithm.

In section 5, a part of the algorithm is shown with the proof in detail.

Finally we conclude this paper by showing an insight about future works.

1.2. Related work

This paper is closely following the paper from Javier E. et al. [1]. In the operational part, so-called Dowling-Gallier procedure [9] is used. Therefore the importance of this paper is not in the development of the algorithm, but in the modifications to the algorithm and its proof in detail.

Due to space limitation, we omit the comparison between the modified version of the algorithm and the original algorithm. For that purpose, however, one can find the original paper from Javier E., et al. [1]

[1] I warmly thank Dr. Alexander Malkis at the Technische Universität München for his valuable advice and help

2. PARALLEL FLOW GRAPHS

In this paper we represent interprocedural control flow of a parallel program by a parallel flow graph system (FGS). A parallel FGS is a set of graphs with hyperedges, where the nodes represent program points, the edges correspond to assignments ($v := Exp$) or call statements ($call\ \Pi_{Exp}$).[2]

Hyperedges of the form $n \to \{n_1, n_2, \ldots, n_k\}$ denote parbegin (parallel begin) commands, while those of the form $\{n_1, n_2, \ldots, n_k\} \to n$ model parend (parallel end) commands. Figure 1 shows an example of parallel FGS. This graph corresponds to the pseudo-code in Table 1.

main()	{ call procedure1;}
procedure1()	$\{x := 1; \parallel y := 2;\}$

Table 1: concrete pseudo-code

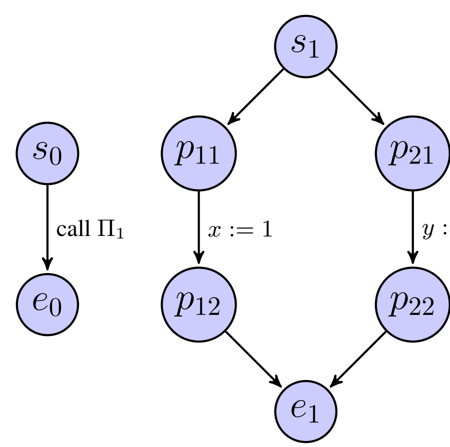

Figure 1: parallel FGS

[2] Strictly speaking, a parallel FGS is a set of hypergraphs since it contains hyperedges. However, we choose the word "graph" since the hyperedges in parallel FGS are expressed by sets of edges, then it may be considered as a set of graphs.

3. THE PROCESS ALGEBRA AND LABELLED TRANSITION SYSTEM

This section explains how to derive the labelled transition system from a given parallel FGS.

3.1. The process algebra

The process algebra is a specification of labelled transitions from terms to terms. To interpret parallel flow graph systems in terms of the process algebra, we need to know their PA-declaration.

3.2. From parallel FGS to PA-declaration Δ

A parallel flow graph system is translated into PA-declaration by the rules in Table 2 [3]. Note that each transition is now labelled with an action a.

in parallel FGS	in PA-declaration
for $n \longrightarrow m$	$N \longrightarrow M$
for $n \xrightarrow{v:=t} m$	$N \xrightarrow{v:=t} M$
for $n \xrightarrow{call\Pi_i(T)} m$	$N \longrightarrow START_i \bullet M$
for end node of procedure Π_i	$END_i \longrightarrow \varepsilon$
for $n \longrightarrow \{m_1, m_2\}$	$N \longrightarrow K \bullet M$, $K \longrightarrow M_1 \parallel M_2$
for $\{m_1', m_2'\} \longrightarrow m$	$M_1' \longrightarrow \varepsilon$, $M_2' \longrightarrow \varepsilon$

Table 2: from parallel FGS to PA-declaration Δ [1, 3]

$N \to M$ is an abbreviation for $N \xrightarrow{\tau} M$. τ denotes a "silent" action, which does not execute an assignment nor call a statement. ε in Table 2 denotes an empty process. An empty process represents a successful termination after the execution of the labelled action a. We need to define terms in the process algebra as follows.

Definition 1 (ε-term)**.** The set of ε-terms is defined as follows.

$$t_\varepsilon = \varepsilon \mid (t_{\varepsilon_1} \bullet t_{\varepsilon_2}) \mid (t_{\varepsilon_1} \parallel t_{\varepsilon_2})$$

Intuitively, ε-terms correspond to processes that do not execute any action [1, 3]. The set of ε-terms is called IsNil.

Definition 2 (T_{PA})**.** The set of all PA-terms over a given set of process constants X is inductively defined as follows.

$$t = X \mid t_\varepsilon \mid (t_1 \bullet t_2) \mid (t_1 \parallel t_2)$$

This definition is similar to the definition of [7, 35]. The set of PA-terms is denoted by T_{PA}.

The rewriting rules over the set of PA-terms are given by the following five rules.

$$\Delta \quad \frac{(X \xrightarrow{a} t) \in \Delta}{X \xrightarrow{a} t}$$

$$\text{seqential1} \quad \frac{t_1 \xrightarrow{a} t_1'}{t_1 \bullet t_2 \xrightarrow{a} t_1' \bullet t_2}$$

$$\text{seqential2} \quad \frac{t_2 \xrightarrow{a} t_2'}{t_1 \bullet t_2 \xrightarrow{a} t_1 \bullet t_2'} \quad (t_1 \in \text{IsNil})$$

$$\text{parallel1} \quad \frac{t_1 \xrightarrow{a} t_1'}{t_1 \parallel t_2 \xrightarrow{a} t_1' \parallel t_2}$$

$$\text{parallel2} \quad \frac{t_2 \xrightarrow{a} t_2'}{t_1 \parallel t_2 \xrightarrow{a} t_1 \parallel t_2'}$$

3.3. pre*, pre, post, and post*

Definition 3 (pre*, pre, post, post*)**.** Given a language L, The set pre*(L), pre(L), post(L), and post*(L) are defined as follows.

pre* (L) = $\{t | t \xrightarrow{*} t' \text{ for some } t' \in L\}$
pre (L) = $\{t | t \to t' \text{ for some } t' \in L\}$
post (L) = $\{t | t' \to t \text{ for some } t' \in L\}$
post* (L) = $\{t | t' \xrightarrow{*} t \text{ for some } t' \in L\}$

where t is a PA-term. This paper focuses on the algorithm for pre*.

4. AUTOMATA AND LANGUAGE

The algorithm works on a given automaton. Before the explanation of tree automata, we need the following definitions.

4.1. Least model and ε-closure

Definition 4 (least model)**.** A model M for a program P is said to be its least model if $M \subseteq M'$ for every model M of P.

Definition 5 (ε-closed language)**.** A language L is ε-closed if the terms t lies in L if and only if $t_\varepsilon \bullet t$, $t_\varepsilon \parallel t$, and $t \parallel t_\varepsilon$ lie in L. Formally, for all $t_\varepsilon \in L$ and $t \in L$

$$t \in L \iff t_\varepsilon \bullet t \in L \iff t_\varepsilon \parallel t \in L \iff t \parallel t_\varepsilon \in L.$$

Definition 6 (ε-closure)**.** The ε-closure of the language L is denoted by $\tilde{L}$ and is defined as follows.

$$\tilde{L} := \bigcap \{M \supseteq L | M \text{ is } \varepsilon\text{-closed}\}$$

4.2. Tree automata

In literature a tree automaton is defined as a tuple, however, tree automata are seen as sets of Horn-clauses in this paper. Horn-clauses are clauses that contain at most one positive literal. Without changing their logical property, Horn-clauses are also expressed as implications. Horn-clauses in this form are called reduction classes.

The automata for a given PA-declaration has the following form. We assume that our automaton $\mathcal{A}$ does not accept ε-term.

1. $q_i(X) \Leftarrow true$
2. $q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)$
3. $q_i(x \bullet y) \Leftarrow q_i(x)$
4. $q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(y)$

[3] As usual, $\parallel$ represents the parallel composition and $\bullet$ denotes the sequential composition in this paper

5. $q_i(x \parallel y) \Leftarrow q_i(x)$
6. $q_i(x \parallel y) \Leftarrow q_i(y)$

where $0 \leqslant i, j, k \leqslant n$ and we fix q_0 as the initial state.

$$L = L_{q_0}$$

Assume that L does not contain ε-term t_ε, every automaton that accepts language L can be transformed into a new automaton that accepts the corresponding $\tilde{L}$.
The only procedure needed for this transformation is to add the following clauses to the original automaton for all states q_i and q_ε. [4]

1. $q_\varepsilon(\varepsilon) \Leftarrow true$
2. $q_i(x \bullet y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)$
3. $q_i(x \parallel y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)$
4. $q_i(x \parallel y) \Leftarrow q_i(x) \wedge q_\varepsilon(y)$

We now the following two facts.

Fact 1 (pre and post are ε-closed)**.** If the language L is ε-closed, then pre*, pre, post, and post* are also ε closed.

Fact 2 (regularity of ε-closure)**.** If the language L is regular, and it does not contain ε-term, then so is its q_ε-closure $\tilde{L}$.

5. THE EFFICIENT ALGORITHM FOR PRE*

Let $\mathcal{A}$ be a tree automaton that does not accept any ε-term and L be the language $\mathcal{A}$ accepts, respectively. $\tilde{\mathcal{A}}$ denotes the new automaton generated by the procedure described above. $(\tilde{L}_{q_i})_{i=0}^n$ is the least model of $\tilde{\mathcal{A}}$. Note that in this paper, we identify L_ε and L_n.

5.1. The declarative part: defining P_A

In the declarative part, we generate logic program P_A by adding clauses in Table 3 to $\tilde{\mathcal{A}}$.
The following three propositions are useful to show why we should compute P_A. For more details, please find Theorem 1 in the original paper [1].

Proposition 1 (8 conditions that define pre*(L))**.** The following 8 implications hold.

1. If $\chi \in L_{q_i}$, then $\chi \in \text{pre*}(L_{q_i})$.[5]
2. If $((X \xrightarrow{a} t) \in \Delta)$ and $(t \in \text{pre*}(L_{q_i}))$,
   then $X \in \text{pre*}(L_{q_i})$.
3. If $((q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(x)) \in \tilde{\mathcal{A}})$
   and $(t_1 \in \text{pre*}(L_{q_j}))$ and $(t_2 \in L_{q_k})$,
   then $t_1 \bullet t_2 \in \text{pre*}(L_{q_i})$.
4. If $((q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}})$
   and $(t_1 \in \text{pre*}(L_{q_i}))$,
   then $(t_1 \bullet t_2) \in \text{pre*}(L_{q_i})$ for $t_2 \in T_{PA}$.
5. If $(t_1 \in \text{pre*}(\text{IsNil}))$ and $(t_2 \in \text{pre*}(L_{q_i}))$,
   then $(t_1 \bullet t_2) \in \text{pre*}(L_{q_i})$.
6. If $((q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(x)) \in \tilde{\mathcal{A}})$
   and $(t_1 \in \text{pre*}(L_{q_j}))$ and $(t_2 \in \text{pre*}(L_{q_k}))$,
   then $(t_1 \parallel t_2) \in \text{pre*}(L_{q_i})$.
7. If $((q_i(x \parallel y) \Leftarrow q_i(x) \in \tilde{\mathcal{A}})$
   and $(t_1 \in \text{pre*}(L_{q_i}))$,
   then $(t_1 \parallel t_2) \in \text{pre*}(L_{q_i})$ for $t_2 \in T_{PA}$.
8. If $((q_i(x \parallel y) \Leftarrow q_i(y) \in \tilde{\mathcal{A}})$
   and $(t_2 \in \text{pre*}(L_{q_i}))$,
   then $(t_1 \parallel t_2) \in \text{pre*}(L_{q_i})$ for $t_1 \in T_{PA}$.

Proof. We prove the above 8 implications one by one.

1. As the premise, we have the following implication in $\tilde{\mathcal{A}}$.

$$\tilde{\mathcal{A}} \models q_i(\chi)$$

   Because $\xrightarrow{*}$ is reflexive,

$$\chi \xrightarrow{0} \chi \text{ where } \tilde{\mathcal{A}} \models q_i(t).$$

   This concludes $\chi \in \text{pre*}(L_{q_i})$
2. Because $t_1 \in \text{pre*}(L_{q_i})$,

$$t \xrightarrow{*} t' \text{ where } \tilde{\mathcal{A}} \models q_i(t').$$

   And there is the following transition.

$$t \xrightarrow{*} t' \text{ where } \tilde{\mathcal{A}} \models q_i(t').$$

   In combination with the first closure of the premise, we know

$$X \xrightarrow{a} t \xrightarrow{*} t' \text{ where } \tilde{\mathcal{A}} \models q_i(t').$$

   i.e., the process constant X can be rewritten to t through t'. Therefore $X \in \text{pre*}(L_{q_i})$.
3. Because $t_1 \in \text{pre*}(L_{q_j})$,

$$t_1 \xrightarrow{*} t_1' \text{ where } \tilde{\mathcal{A}} \models q_j(t_1').$$

   By applying the sequential rule 1 to the above transition repeatedly, the following transition is obtained.

$$t_1 \bullet t_2 \xrightarrow{*} t_1' \bullet t_2 \text{ where } \tilde{\mathcal{A}} \models q_j(t_1').$$

$p_i(\chi) \Leftarrow q_i(\chi)$	
	for each $\chi \in \{\text{process constants of}\Delta\} \cup \{\varepsilon\}$
$p_i(X) \Leftarrow p_i(t)$	
	for each $(X \xrightarrow{a} t) \in \Delta$
$p_i(x_1 \bullet x_2) \Leftarrow p_j(x_1) \wedge q_k(x_2)$	
	for each $(q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$
$p_i(x_1 \bullet x_2) \Leftarrow p_i(x_1)$	
	for each $(q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$
$p_i(x_1 \bullet x_2) \Leftarrow p_\varepsilon(x_1) \wedge p_i(x_2)$	
$p_i(x_1 \parallel x_2) \Leftarrow p_j(x_1) \wedge q_k(x_2)$	
	for each $(q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$
$p_i(x_1 \parallel x_2) \Leftarrow p_i(x_1)$	
	for each $(q_i(x \parallel y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$
$p_i(x_1 \parallel x_2) \Leftarrow p_i(x_2)$	
	for each $(q_i(x \parallel y) \Leftarrow q_i(y)) \in \tilde{\mathcal{A}}$

Table 3: defining predicate

[4] Note that in these additional clauses, the q_is in premise and in conclusion have to have the same index i.

[5] χ is either an empty process or a process constant

Because $t_2 \in L_{q_k}$,

$$\tilde{\mathcal{A}} \models q_k(t_2).$$

As the premise, we have the following implication in $\tilde{\mathcal{A}}$.

$$(q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$$

Substitute $x = t_1'$, $y = t_2$, respectively.
Since the premise of the implication in $\tilde{\mathcal{A}}$ holds, the conclusion also holds, i.e.,

$$\tilde{\mathcal{A}} \models q_i(t_1' \bullet t_2).$$

Therefore the transition is now expressed as follows,

$$t_1 \bullet t_2 \xrightarrow{*} t_1' \bullet t_2 \text{ where } \tilde{\mathcal{A}} \models q_i(t_1' \bullet t_2).$$

This concludes $t_1 \bullet t_2 \in \text{pre*}(L_{q_i})$.

4. Because $t_1 \in \text{pre*}(L_{q_i})$,

$$t_1 \xrightarrow{*} t_1' \text{ where } \tilde{\mathcal{A}} \models q_i(t_1').$$

By applying the sequential rule 1 to the above transition repeatedly, the following transition is obtained.

$$t_1 \bullet t_2 \xrightarrow{*} t_1' \bullet t_2 \text{ where } \tilde{\mathcal{A}} \models q_i(t_1').$$

As the premise, we have the following implication in $\tilde{\mathcal{A}}$.

$$(q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$$

Substitute $x = t_1'$, $y = t_2$, respectively.
Since the premise of the implication in $\tilde{\mathcal{A}}$ holds, the conclusion also holds, i.e.,

$$\tilde{\mathcal{A}} \models q_i(t_1' \bullet t_2).$$

Therefore the transition is now expressed as follows,

$$t_1 \bullet t_2 \xrightarrow{*} t_1' \bullet t_2 \text{ where } \tilde{\mathcal{A}} \models q_i(t_1' \bullet t_2).$$

This concludes $t_1 \bullet t_2 \in \text{pre*}(L_{q_i})$.

5. Because $t_1 \in \text{pre*}(\text{IsNil})$,

$$t_1 \xrightarrow{*} t_\varepsilon \text{ where } \tilde{\mathcal{A}} \models q_\varepsilon(t_\varepsilon).$$

Because $t_2 \in \text{pre*}(L_{q_i})$,

$$t_2 \xrightarrow{*} t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_2').$$

Since we restrict our algorithm to an automaton that defines ε-closed languages,

$$t_2' \in L_{q_i} \iff t_\varepsilon \bullet t_2' \in L_{q_i} \iff \tilde{\mathcal{A}} \models q_i(t_\varepsilon \bullet t_2').$$

By applying the sequential rule 1 to the transition of t_1 repeatedly, the following transitions are obtained.

$$t_1 \bullet t_2' \xrightarrow{*} t_\varepsilon \bullet t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_\varepsilon \bullet t_2')$$

$$t_1 \bullet t_2 \xrightarrow{*} t_\varepsilon \bullet t_2 \text{ where } \tilde{\mathcal{A}} \models q_i(t_\varepsilon) \text{ and } t_2 \in T_{PA}.$$

By applying the sequential rule 2 to the the transition of t_2 repeatedly, the following transition is obtained.

$$t_\varepsilon \bullet t_2 \xrightarrow{*} t_\varepsilon \bullet t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_\varepsilon \bullet t_2').$$

By combining the obtained transitions so far, we know

$$t_1 \bullet t_2 \xrightarrow{*} t_\varepsilon \bullet t_2 \xrightarrow{*} t_\varepsilon \bullet t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_\varepsilon \bullet t_2').$$

This concludes $t_1 \bullet t_2 \in \text{pre*}(L_{q_i})$.

6. Because $t_1 \in \text{pre*}(L_{q_j})$,

$$t_1 \xrightarrow{*} t_1' \text{ where } \tilde{\mathcal{A}} \models q_j(t_1').$$

Because $t_2 \in \text{pre*}(L_{q_k})$,

$$t_2 \xrightarrow{*} t_2' \text{ where } \tilde{\mathcal{A}} \models q_k(t_2').$$

As the premise, we have the following implication in $\tilde{\mathcal{A}}$,

$$(q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$$

Substitute $x = t_1'$, $y = t_2'$, respectively.
Since the premise of the implication in $\tilde{\mathcal{A}}$ holds, the conclusion also holds, i.e.,

$$\tilde{\mathcal{A}} \models q_i(t_1' \parallel t_2').$$

By applying the parallel rule 1 to the transition of the left component a number of times, and the parallel rule 2 to the transition of the right component a number of times, the following transitions are obtained.

$$t_1 \parallel t_2 \xrightarrow{*} t_1' \parallel t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_1' \parallel t_2').$$

This concludes $t_1 \parallel t_2 \in \text{pre*}(L_{q_i})$.

7. Because $t_1 \in \text{pre*}(L_{q_i})$,

$$t_1 \xrightarrow{*} t_1' \text{ where } \tilde{\mathcal{A}} \models q_i(t_1').$$

As the premise, we have the following implication in $\tilde{\mathcal{A}}$.

$$(q_i(x \parallel y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$$

Substitute $x = t_1'$, $y = t_2$, respectively.
Since the premise of the implication in $\tilde{\mathcal{A}}$ holds, the conclusion also holds, i.e.,

$$\tilde{\mathcal{A}} \models q_i(t_1' \parallel t_2)$$

By applying the parallel rule 1 to the transition of the left component, repeatedly, the following transition is obtained.

$$t_1 \parallel t_2 \xrightarrow{*} t_1' \parallel t_2 \text{ where } \tilde{\mathcal{A}} \models q_i(t_1' \parallel t_2).$$

This concludes $t_1 \parallel t_2 \in \text{pre*}(L_{q_i})$.

8. Because $t_2 \in \text{pre*}(L_{q_i})$,

$$t_2 \xrightarrow{*} t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_2').$$

As the premise, we have the following implication in $\tilde{\mathcal{A}}$.

$$(q_i(x \parallel y) \Leftarrow q_i(y)) \in \tilde{\mathcal{A}}$$

Substitute $x = t_1$, $y = t_2'$, respectively.
Since the premise of the implication in $\tilde{\mathcal{A}}$ holds, the conclusion also holds, i.e.,

$$\tilde{\mathcal{A}} \models q_i(t_1 \parallel t_2').$$

By applying the parallel rule 2 to the transition of the right component , repeatedly, the following transition is obtained.

$$t_1 \parallel t_2 \xrightarrow{*} t_1 \parallel t_2' \text{ where } \tilde{\mathcal{A}} \models q_i(t_1 \parallel t_2').$$

This concludes $t_1 \parallel t_2 \in \text{pre*}(L_{q_i})$.

$\square$

Proposition 2 (smallest). Let $(S_i)_{i=0}^n$ be an arbitrary set that satisfy the following 8 conditions where $n+1$ is the number of states in the new automaton. Then, $\mathrm{pre}^*((L_{q_i})_{i=0}^n) \sqsubseteq (S_i)_{i=0}^n$, i.e., $\mathrm{pre}^*(L_{q_i})$ is the smallest set S_i satisfying the following 8 conditions[6]. Note that we identify S_ε and S_n.

1. If $\chi \in L_{q_i}$, then $\chi \in S_i$.
2. If $((X \xrightarrow{a} t) \in \Delta)$ and $(t \in S_i)$, then $X \in S_i$.
3. If $((q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(x)) \in \tilde{\mathcal{A}})$ and $(t_1 \in S_j)$ and $(t_2 \in L_{q_k})$, then $t_1 \bullet t_2 \in S_i$.
4. If $((q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}})$ and $(t_1 \in S_i)$, then $(t_1 \bullet t_2) \in S_i$ for $t_2 \in T_{PA}$.
5. If $(t_1 \in S_\varepsilon)$ and $(t_2 \in S_i)$, then $(t_1 \bullet t_2) \in S_i$.
6. If $((q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(x)) \in \tilde{\mathcal{A}})$ and $(t_1 \in S_j)$ and $(t_2 \in S_k)$, then $(t_1 \parallel t_2) \in S_i$.
7. If $((q_i(x \parallel y) \Leftarrow q_i(x) \in \tilde{\mathcal{A}})$ and $(t_1 \in S_i)$, then $(t_1 \parallel t_2) \in S_i$ for $t_2 \in T_{PA}$.
8. If $((q_i(x \parallel y) \Leftarrow q_i(y) \in \tilde{\mathcal{A}})$ and $(t_2 \in S_i)$, then $(t_1 \parallel t_2) \in S_i$ for $t_1 \in T_{PA}$.

Proof. Let $S_0, \ldots, S_n$ be arbitrary sets satisfying the 8 conditions, where $n+1$ is the number of states of the newly generated ε-closed automaton $\tilde{\mathcal{A}}$. We prove this proposition by showing that for every term t and for every $i = 0, \ldots, n$ if $t \in \mathrm{pre}^*(L_{q_i})$ then $t \in S_i$. Formally,

$$\text{Goal1} \quad \forall k, i, t, t' : \quad ((t \xrightarrow{k} t' \in L_{q_i}) \Rightarrow (t \in S_i))$$

We use double induction to prove this implication.[7] A double induction hypothesis consists of two levels of inductions. In the outer level, we use an induction on the length of transitions. In the inner level, we use an induction on the size of term t.

The base case for the outer level of induction is as follows.

$$\text{BC1} \quad \forall i, t, t' : \quad ((t \xrightarrow{0} t' \in L_{q_i}) \Rightarrow (t \in S_i))$$

Note that BC1 is not yet proved. It is necessary to prove BC1 in the inner level of induction.

Similarly to BC1, we assume an induction hypothesis for the outer level of induction.

$$\text{IH1} \quad \forall k \leq m, i, t, t' : \quad ((t \xrightarrow{k} t' \in L_{q_i}) \Rightarrow (t \in S_i))$$

The whole proof is done by showing that under this assumption (IH1), the following induction step (IS1) holds.

$$\text{IS1} \quad \forall i, t, t' : \quad ((t \xrightarrow{m+1} t' \in L_{q_i}) \Rightarrow (t \in S_i))$$

[6] $\sqsubseteq$ is the component-wise ordering.

[7] It is possible to avoid the use of double induction. For example, one can use induction on lexicographic order.

Note that similarly to the case for BC1 it is necessary to use one more induction on the size of term t in the second level.

Therefore the whole proof for proposition 2 consists of 1 induction on the length of transitions in the first level and 2 inductions on the size of terms t in the second level.

We start with the proof of BC1. To prove BC1, An induction is applied on the size of term t.[8] The base case is as follows.

$$\text{BC2}_1 \quad \forall i, t, t' : \quad ((t \xrightarrow{0} t' \in L_{q_i}) \wedge (|t| = 1) \Rightarrow (t \in S_i))$$

Because $|t| = 1$, $t = \chi$. Since $\xrightarrow{0}$ is the identity, $t = t'$. Therefore BC2_1 is expressed as follows.

$$\text{BC2}'_1 \quad \forall i, t : \quad ((\chi \in L_{q_i}) \Rightarrow (\chi \in S_i))$$

This coincides with the first condition. Therefore BC2_1 holds. To prove BC1 from BC2_1, we introduce the following induction hypothesis (IH2_1).

$$\text{IH2}_1 \quad \forall i, t, t' : \quad ((t \xrightarrow{0} t' \in L_{q_i}) \wedge (|t| \leq n \in \mathbb{N}) \Rightarrow (t \in S_i))$$

Since $\xrightarrow{0}$ is the identity, $t = t'$. Therefore BC2_1 is expressed as follows.

$$\text{IH2}'_1 \quad \forall i, t \quad ((t \xrightarrow{0} t \in L_{q_i}) \wedge (|t| \leq n \in \mathbb{N}) \Rightarrow (t \in S_i))$$

Our current aim is to prove the following induction step (IS2_1) using IH2_1.

$$\text{IS2}_1 \quad \forall i, t \quad ((t \xrightarrow{0} t \in L_{q_i}) \wedge (|t| = n+1) \Rightarrow (t \in S_i))$$

All terms of size $n+1$ have one of the following forms.

$$t = t_1 \bullet t_2$$
$$t = t_1 \parallel t_2$$

where $|t_1| \leq n \in \mathbb{N}$ and $|t_2| \leq n \in \mathbb{N}$. Remember that the automaton $\tilde{\mathcal{A}}$ may have only the following clauses.

1. $q_i(\chi) \Leftarrow true$
2. $q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)$
3. $q_i(x \bullet y) \Leftarrow q_i(x)$
4. $q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(y)$
5. $q_i(x \parallel y) \Leftarrow q_i(x)$
6. $q_i(x \parallel y) \Leftarrow q_i(y)$
7. $q_\varepsilon(\varepsilon) \Leftarrow true$
8. $q_i(x \bullet y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)$
9. $q_i(x \parallel y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)$
10. $q_i(x \parallel y) \Leftarrow q_i(x) \wedge q_\varepsilon(y)$

If t is a sequential composition of two smaller terms t_1 and t_2, one of the following conditions holds.

Case 1: $(q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$ and $\tilde{\mathcal{A}} \models q_j(t_1)$ and $\tilde{\mathcal{A}} \models q_k(t_2)$.

Case 2: $(q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$ and $\tilde{\mathcal{A}} \models q_i(t_1)$.

[8] In this paper the size of a terms t is defined as the number of its leaves when it is seen as a tree. And the size is denoted by $|t|$. For example, the size of term $(t_1 \bullet t_2) \parallel t_3$ is not 5 but 3.

Case 3: $(q_i(x \bullet y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)) \in \tilde{\mathcal{A}}$
and $\tilde{\mathcal{A}} \models q_\varepsilon(t_1)$ and $\tilde{\mathcal{A}} \models q_i(t_2)$.

In case 1,

$$(q_i(x \bullet y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}} \text{ and } \tilde{\mathcal{A}} \models q_j(t_1) \text{ and } \tilde{\mathcal{A}} \models q_k(t_2).$$

Therefore

$$t_2 \in L_{q_k}.$$

Because of IH$2'_1$,

$$t_1 \in S_j.$$

This coincides with the premise of the third condition. Therefore

$$t_1 \bullet t_2 \in S_i.$$

This concludes that IS2_1 holds in this case.
In case 2,

$$(q_i(x \bullet y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}} \text{ and } \tilde{\mathcal{A}} \models q_i(t_1).$$

Because of IH$2'_1$,

$$t_1 \in S_i.$$

This coincides with the premise of the fourth condition. Therefore

$$t_1 \bullet t_2 \in S_i.$$

where $t_2 \in T_{PA}$. This concludes that IS2_1 holds in this case.
Due to space limitation the proof for case 3 is omitted.
If t is a parallel composition of two smaller terms, at least one of the following conditions holds.

Case 1: $(q_i(x \parallel y) \Leftarrow q_j(x) \wedge q_k(y)) \in \tilde{\mathcal{A}}$
and $\tilde{\mathcal{A}} \models q_j(t_1)$ and $\tilde{\mathcal{A}} \models q_k(t_2)$.

Case 2: $(q_i(x \parallel y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}}$ and $\tilde{\mathcal{A}} \models q_i(t_1)$.

Case 3: $(q_i(x \parallel y) \Leftarrow q_i(y)) \in \tilde{\mathcal{A}}$ and $\tilde{\mathcal{A}} \models q_i(t_2)$.

Case 4: $q_i(x \parallel y) \Leftarrow q_\varepsilon(x) \wedge q_i(y)) \in \tilde{\mathcal{A}}$
and $\tilde{\mathcal{A}} \models q_\varepsilon(t_1)$ and $\tilde{\mathcal{A}} \models q_i(t_2)$.

Case 5: $q_i(x \parallel y) \Leftarrow q_i(x) \wedge q_\varepsilon(y)) \in \tilde{\mathcal{A}}$
and $\tilde{\mathcal{A}} \models q_i(t_1)$ and $\tilde{\mathcal{A}} \models q_\varepsilon(t_2)$.

In case 1,

$$(q_i(t_1 \parallel t_2) \Leftarrow q_j(t_1) \wedge q_k(t_2)) \in \tilde{\mathcal{A}} \text{ and } \tilde{\mathcal{A}} \models q_j(t_1) \text{ and } \tilde{\mathcal{A}} \models q_k(t_2).$$

Therefore

$$t_1 \in L_{q_j} \text{ and } t_2 \in L_{q_k}.$$

Because of IH$2'_1$,

$$t_1 \in S_j \text{ and } t_2 \in S_k.$$

This coincides with the premise of the sixth condition. Therefore

$$t_1 \parallel t_2 \in S_i.$$

This concludes that IS2_1 holds in this case.
In case 2,

$$(q_i(x \parallel y) \Leftarrow q_i(x)) \in \tilde{\mathcal{A}} \text{ and } \tilde{\mathcal{A}} \models q_i(t_1).$$

Therefore

$$t_1 \in L_{q_i}.$$

Because of IH$2'_1$,

$$t_1 \in S_i.$$

This coincides with the premise of the seventh condition. Therefore

$$t_1 \parallel t_2 \in S_i.$$

This concludes that IS2_1 holds in this case.
In case 3,

$$(q_i(x \parallel y) \Leftarrow q_i(y)) \in \tilde{\mathcal{A}} \text{ and } \tilde{\mathcal{A}} \models q_i(y).$$

Therefore

$$t_2 \in L_{q_i}.$$

Because of IH$2'_1$,

$$t_2 \in S_i$$

This coincides with the premise of the eighth condition. Therefore

$$t_1 \parallel t_2 \in S_i$$

This concludes that IS2_1 holds in this case.
Due to space limitation the proof for case 4 is omitted.
Due to space limitation the proof for case 5 is omitted.
Therefore, we know that IS2_1 holds in any possible case if t is a parallel composition of smaller terms. From BC2_1, IH2_1 and IS2_1, we know that BC1 holds.
Now we prove that IS1 holds under IH1. To prove IS1, an induction is applied on the size of term t. The base case is as follows.

$$\text{BC}2_2 \quad \forall i, t, t' : \quad ((t \xrightarrow{m+1} t' \in L_{q_i}) \wedge (|t| = 1) \Rightarrow (t \in S_i))$$

Because $|t| = 1$, $t = \chi$. However, an empty process cannot be reduced further. Therefore $t = X$, and BC2_2 is expressed as follows.

$$\text{BC}2'_2 \quad \forall i, t'' : \quad ((X \xrightarrow{m+1} t'' \in L_{q_i}) \Rightarrow (X \in S_i))$$

Because BC$2'_2$ is expressed by an implication, we can assume the following transition as its premise.

$$\forall i, t', t'' \quad X \to t' \xrightarrow{m} t'' \in L_{q_i}$$

Due to IH1,

$$(t' \xrightarrow{m} t'' \in L_{q_i}) \Rightarrow (t' \in S_i).$$

Therefore

$$t' \in S_i$$

Accordingly, the first transition of the assumption is expressed as follows.

$$\forall i, t' \quad X \to t' \in S_i$$

This coincides with the premise of the second condition. Therefore

$$X \in S_i.$$

This concludes that $\mathrm{BC2}'_2$ holds.
To prove IS1 from IH1 and $\mathrm{BC2}'_2$, we introduce the following induction hypothesis ($\mathrm{IH2}_2$).

$$\mathrm{IH2}_2 \quad \forall i, t, t' : ((t \xrightarrow{m+1} t' \in L_{q_i}) \wedge (|t| \leq n \in \mathbb{N}) \Rightarrow (t \in S_i)$$

Our current aim is to prove the following induction step ($\mathrm{IS2}_2$) using $\mathrm{IH2}_1$.

$$\mathrm{IS2}_2 \quad \forall i, t, t' : ((t \xrightarrow{m+1} t' \in L_{q_i}) \wedge (|t| = n+1) \Rightarrow (t \in S_i))$$

Every term of size $n+1$ has one of the following forms.

$$t = t_1 \bullet t_2$$
$$t = t_1 \parallel t_2$$

where $|t_1| \leq n \in \mathbb{N}$ and $|t_2| \leq n \in \mathbb{N}$. Remember that the automaton $\tilde{\mathcal{A}}$ may have the above mentioned 10 clauses. The premise of $\mathrm{IS2}_2$ has to have one of the following forms.

Case A: $\forall i, t_1, t_2, t'_1 \quad t_1 \bullet t_2 \xrightarrow{m+1} t'_1 \bullet t_2 \in L_{q_i}$
where $t_1 \xrightarrow{m+1} t'_1$ and t_2 does not change on any path from $t_1 \bullet t_2$ to $t'_1 \bullet t_2$.

Case B: $\forall i, t_1, t_2, t'_2 \quad t_1 \bullet t_2 \xrightarrow{m+1} t_1 \bullet t'_2 \in L_{q_i}$
where $t_2 \xrightarrow{m+1} t'_2$ and $t_1 \in L_{q_\varepsilon}(= \mathrm{IsNil})$ and t_1 does not change on any path from $t_1 \bullet t'_2$ to $t_1 \bullet t'_2$.

Case C: $\forall i, t_1, t_2, t'_1, t'_2 \quad t_1 \bullet t_2 \xrightarrow{m+1} t'_1 \bullet t'_2 \in L_{q_i}$
where $t_1 \xrightarrow{k} t'_1 \in L_{q_\varepsilon}(= \mathrm{IsNil})$,
and $t_2 \xrightarrow{m-k+1} t'_2$,
for some k s.t. $1 \leq k \leq m$.

The detailed proof is omitted here due to space limitation. By using case distinctions, one can conclude that $\mathrm{IS2}_2$ holds for all cases.
Similarly to the case of sequential composition, one can prove $\mathrm{IS2}_2$ for parallel composition. Therefore $\mathrm{IS2}_2$ holds. Since $\mathrm{IS2}_2$ holds, IS1 also holds under IH1. This concludes the proof of proposition 2. □

Proposition 3 ($\mathrm{pre}^*(L_{q_i})$)**.** The sets $\mathrm{pre}^*(L_{q_i})$ (for $i = \varepsilon, 0, 1, \ldots, n$) are the smallest sets satisfying the 8 conditions.

Proof. Directly from the previous propositions. □

5.2. The Operational Part: $P_A \mapsto SatP_A \mapsto RedP_A$

Due to space limitation, we focuse on the declarative part of the original algorithm in this paper. One can find the operational part of the original algorothm in [1].

6. CONCLUSIONS

We have modified the algorithm in [1] and provided the detailed proof of it. This algorithm is supposed to enable the efficient computation of pre*.

6.1. Open problems and future works

Similarly to what is shown for pre* in this paper, the algorithm in [1] requires modifications in the declarative part for pre, post, and post* as well. Furthermore the detailed proof of operational part is expected to be done.

7. REFERENCES

[1] Javier Esparza and Andreas Podelski ”Efficient Algorithms for pre* and post* on Interprocedural Parallel Flow Graphs”

[2] Luca Aceto, Wan Fokkink, and Chris Verhoef ”Structural Operational Semantics”

[3] Wan Fokkink ”Introduction to Process Algebra”

[4] C.A.R.Hoare ”Communicating Sequential Processes”

[5] J.C.M Baeten ”A Brief History of Process Algebra”

[6] Christel Baier and Joost-Pieter Katoen ”Principle of Model Checking”

[7] Franz Baader and Tobias Nipkow ”Term Rewriting and All That”

[8] Uwe Schöning ”Logic for Computer Scientists”

[9] William F. Dowling and Jean H. Gallier ”Linear-time algorithms for testing the satisfiability of propositional horn formulae”

[10] Joost-Pieter Katoen ”The State Explosion Problem Lecture 5a of Model Checking”