On Choosing a Scapegoat in the Stubborn Set Method

Kimmo Varpaaniemi

Helsinki University of Technology
Digital Systems Laboratory
Otakaari 1, SF-02150, Finland
Kimmo.Varpaaniemi@hut.fi

Abstract. The incremental algorithm for computing stubborn sets for Petri nets produces a stubborn set that may contain unnecessarily many enabled transitions since the algorithm contains nondeterministic choices. These choices involve choosing the starting transition and the choice of a disabling place (the scapegoat). The need for the first choice can be eliminated without affecting the complexity of the algorithm by visiting all the enabled transitions, but the latter choice remains. This paper compares different ways of choosing the scapegoat.

1 Introduction

Reachability analysis is a powerful method to analyze concurrent and distributed systems. Using it we can easily check whether certain properties hold or not. If we are primarily interested in such properties as the existence of deadlocks, we don't necessarily have to generate the complete state space. Even though finding a deadlock in a Petri net is a polynomial space hard problem [6], we can utilize special properties of systems and have a deadlock detection algorithm that is efficient for a large subclass of Petri nets.

Antti Valmari's stubborn set method [2, 4, 5, 7, 8, 9, 10, 11] is a reductive reachability graph generation method which eliminates redundant interleavings of actions. This paper studies the stubborn set method on finite place/transition nets with infinite capacities. A stubborn set consists of transitions. The stubborn set method produces a reduced reachability graph by firing only transitions in a chosen stubborn set. The method preserves all terminal states [4] and the existence of infinite transition sequences [8]. This paper uses the definition of stubbornness in [4] because it is one of the weakest definitions of stubbornness. The weaker the definition, the smaller is the reduced reachability graph. In practice, deadlock detection is often hard enough.

The transitions of the complement of a stubborn set cannot enable any disabled transition of the stubborn set, and there is at least one enabled transition in the stubborn set that cannot be disabled by the transitions of the complement [2]. Some transitions may be *ignored*: there may be live transitions that are never fired. A little stronger definition of stubbornness guarantees that no enabled transition in the stubborn set can be disabled by the transitions of the complement. Then the ignored transitions can be detected and the ignoring phenomenon can be eliminated [9]. Valmari has also presented an algorithm that can be used to decide the truth value of a linear temporal logic formula [10].

Valmari has presented three algorithms for finding a suitable stubborn set: candidate list algorithm [4], incremental algorithm [4, 7] ("algorithm using strongly connected components" [2, 12]), and deletion algorithm [5]. Let's assume that T is the set of transitions, μ is the maximum number of input places of a transition, ν is the maximum of the maximum number of input transitions of a place and the maximum number of output transitions of a place, and ρ is the maximum number of adjacent transitions of a place. The candidate list algorithm finds a stubborn set in candidate list $T_1, \ldots, T_n, T$ in time $O(\mu \sum_{i=1}^{n} |T_i|)$. The incremental algorithm produces a stubborn set in time $O(\mu\nu|T|)$. The produced set is minimal only in a very restricted sense [4]. The deletion algorithm finds a stubborn set which is minimal in the sense that no proper subset of its enabled transitions can be the set of all enabled transitions of any stubborn set. The stubborn set is found in time $O(\mu\rho|T|^2)$ [5, 7]. No one has presented any algorithm that would find a stubborn set having a minimum number of enabled transitions in polynomial time with respect to the number of places and transitions. Such set is not necessarily the best choice [7] but it is difficult to define a better simple goal.

At its best, the stubborn set method reduces the reachability graph very much. For the classical system of n dining philosophers, the size of the whole reachability graph is exponential in n. The stubborn set method generates a graph the size of which is quadratic in n [4].

This paper concentrates on the incremental algorithm. The algorithm contains nondeterministic choices which affect the number of enabled transitions in the result set. These choices involve choosing the starting transition and the choice of a disabling place (the scapegoat). The need for the first choice can be eliminated without affecting the complexity of the algorithm by visiting all the enabled transitions, but the latter choice remains. This paper compares different ways of choosing the scapegoat.

The incremental algorithm is presented in Section 2. Section 3 shows an example in which a net has both extremely bad and many optimal or almost optimal strategies for choosing a scapegoat. Some general strategies for choosing a scapegoat and two ways to estimate the goodness of a given strategy are presented in Section 4.

2 The incremental algorithm

This section presents the incremental algorithm of [4] for finite place/transition nets with infinite capacities. The algorithm does not use all parts of the definition of stubbornness. In fact, it produces so strongly stubborn sets that ignoring elimination [9] could be added to the reachability graph generation algorithm. This paper does not study ignoring elimination because deadlock detection is often hard enough.

Let's give some preliminary definitions first. The power set (the set of subsets) of set A is denoted by 2^A. The set of partial functions from set A to set B is $\{R \subseteq A \times B \mid \forall x\ \forall y\ \forall z\ (\langle x,y\rangle \in R \wedge \langle x,z\rangle \in R) \Rightarrow y = z\}$. The set of functions from set A to set B, denoted by $(A \to B)$, is

$$\{R \subseteq A \times B \mid (\forall x\ \forall y\ \forall z\ (\langle x,y\rangle \in R \wedge \langle x,z\rangle \in R) \Rightarrow y = z) \wedge (\forall x \in A\ \exists y \in B\ \langle x,y\rangle \in R)\}.$$

So there is always an empty function from an empty set to any set but there is no function from a nonempty set to an empty set. The set of natural numbers, including

0, is denoted by N. ω is a formal infinite number. N_ω denotes $N \cup \{\omega\}$. Relation $\leq$ is extended by defining $\omega \leq \omega$ and $\forall n \in N\ n \leq \omega$. Addition and subtraction are extended by defining $\forall n \in N\ \omega + n = \omega$, and $\forall n \in N\ \omega - n = \omega$.

Definition 1. A place/transition net is a sextuple $\langle S, T, F, K, W, M_0 \rangle$ so that S and T are sets, $S \cap T = \emptyset$, $F \subseteq (S \times T) \cup (T \times S)$, $K \in (S \to N_\omega)$, $W \in (F \to (N \setminus \{0\}))$, $M_0 \in (S \to N_\omega)$, and $\forall s \in S\ M_0(s) \leq K(s)$. S is the set of places, T is the set of transitions, F is the set of arcs, K is the capacity function, W is the arc weight function, and M_0 is the initial marking. If $x \in S \cup T$, then the set of input elements of x is ${}^\bullet x = \{y \mid \langle y, x \rangle \in F\}$, and the set of output elements of x is $x^\bullet = \{y \mid \langle x, y \rangle \in F\}$. W is extended to $(((S \times T) \cup (T \times S)) \to N)$ by defining $W(x, y) = 0$ when $\langle x, y \rangle \notin F$. $\{M \in (S \to N_\omega) \mid \forall s \in S\ M(s) \leq K(s)\}$ is the set of markings of the net. The net is finite if and only if $S \cup T$ is finite. Place s has an infinite capacity if and only if $K(s) = \omega$. The net is a net with infinite capacities if and only if each place has an infinite capacity.

If s is a place of a net and M is a marking of the net, then $M(s)$ is called the number of tokens in s at M. If each place has an infinite capacity, then the set of markings of the net is simply $(S \to N_\omega)$.

Definition 2. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Transition t is enabled at marking M ($M[t\rangle$ for short) if and only if $\forall s \in {}^\bullet t\ M(s) \geq W(s, t)$ and $\forall s \in t^\bullet\ M(s) \leq K(s) - W(t, s)$. Transition t can be fired from marking M to marking M' ($M[t\rangle M'$ for short) if and only if $M[t\rangle$ and $\forall s \in S\ M'(s) = M(s) - W(s, t) + W(t, s)$. Transition t is disabled at marking M if and only if t is not enabled at M. Place s is a disabling place of transition t at marking M if and only if $M(s) < W(s, t)$ or $M(s) > K(s) - W(t, s)$. A partial function f from $(S \to N_\omega) \times T$ to S is a scapegoat generator of the net if and only if for each marking M and each disabled transition t at M, $f(M, t)$ is a disabling place of t at M.

If each place has an infinite capacity, then the output places of a transition do not affect the enabledness of the transition. Definition 2 does not define a scapegoat generator to be a function because S can be empty.

Definition 3. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let $\mathcal{M}$ be the set of markings of the net. Relation $[\rangle = \{\langle M, M' \rangle \in \mathcal{M} \times \mathcal{M} \mid \exists t \in T\ M[t\rangle M'\}$. The reflexive-transitive closure of $[\rangle$ is denoted by $[\rangle^*$. Marking M' is reachable from marking M if and only if $M[\rangle^* M'$. The reachability graph of the net is a pair $\langle V, E \rangle$ so that the set of vertices $V = \{M \mid M_0[\rangle^* M\}$, and the set of edges $E = \{\langle M, t, M' \rangle \in V \times T \times V \mid M[t\rangle M'\}$.

Definition 4. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let $\mathcal{M}$ be the set of markings of the net, and T_g a function from $\mathcal{M}$ to 2^T. Relation $[\rangle_{T_g} = \{\langle M, M' \rangle \in \mathcal{M} \times \mathcal{M} \mid \exists t \in T_g(M)\ M[t\rangle M'\}$. The reflexive-transitive closure of $[\rangle_{T_g}$ is denoted by $[\rangle^*_{T_g}$. The T_g-reachability graph of the net is a pair $\langle V, E \rangle$ so that the set of vertices $V = \{M \mid M_0[\rangle^*_{T_g} M\}$, and the set of edges $E = \{\langle M, t, M' \rangle \in V \times T \times V \mid t \in T_g(M) \wedge M[t\rangle M'\}$.

From now on, only finite place/transition nets with infinite capacities will be considered.

Definition 5. Let $\langle S, T, F, K, W, M_0 \rangle$ be a finite place/transition net with infinite capacities. Functions E_1, E_2, E_3 and E_4 from $(S \to N_\omega) \times T \times S$ to 2^T are defined as follows: Let $M \in (S \to N_\omega)$, $t \in T$, and $s \in S$.
$E_1(M, t, s) = \{t' \in {}^\bullet s \mid M(s) \geq W(s, t') \wedge W(t', s) > W(s, t')\}$,
$E_2(M, t, s) = \{t' \in s^\bullet \mid W(s, t') > W(t', s) \vee$
$(W(s, t) > W(t, s) \wedge W(s, t') > M(s) - W(s, t) + W(t, s))\}$,
$E_3(M, t, s) = \{t' \in {}^\bullet s \mid M(s) \geq W(s, t') \wedge (W(t', s) > W(s, t') \vee$
$W(t', s) > W(t, s))\}$, and
$E_4(M, t, s) = \{t' \in s^\bullet \mid W(s, t') > W(t', s)\}$.

Definition 6. Let $\langle S, T, F, K, W, M_0 \rangle$ be a finite place/transition net with infinite capacities. A set $T_s \subseteq T$ is stubborn at marking M if and only if
$\forall t \in T_s \ (\exists s \in {}^\bullet t \ M(s) < W(s, t) \wedge E_1(M, t, s) \subseteq T_s) \vee$
$(M[t\rangle \wedge (\forall s \in {}^\bullet t \ E_2(M, t, s) \subseteq T_s \vee E_3(M, t, s) \subseteq T_s))$
and $\exists \tau \in T_s \ M[\tau\rangle \wedge (\forall s \in {}^\bullet \tau \ E_4(M, \tau, s) \subseteq T_s)$.

Definition 7. Let $\langle S, T, F, K, W, M_0 \rangle$ be a finite place/transition net with infinite capacities. Let G be the set of scapegoat generators of the net. Function E_{12} from $G \times (S \to N_\omega) \times T$ to 2^T is defined by
$\forall f \in G \ \forall M \in (S \to N_\omega) \ \forall t \in T \ E_{12}(f, M, t) = \begin{cases} E_1(M, t, f(M, t)) & \text{if } \neg M[t\rangle, \\ \bigcup_{s \in {}^\bullet t} E_2(M, t, s) & \text{if } M[t\rangle. \end{cases}$
Function R_{12} from $G \times (S \to N_\omega)$ to $2^{T \times T}$ is defined by
$\forall f \in G \ \forall M \in (S \to N_\omega) \ R_{12}(f, M) = \{\langle t, t' \rangle \in T \times T \mid t' \in E_{12}(f, M, t)\}$.
For each scapegoat generator f and marking M, $(R_{12}(f, M))^*$ denotes the reflexive-transitive closure of $R_{12}(f, M)$. Function E^*_{12} from $G \times (S \to N_\omega) \times T$ to 2^T is defined by $\forall f \in G \ \forall M \in (S \to N_\omega) \ \forall t \in T \ E^*_{12}(f, M, t) = \{t' \mid \langle t, t' \rangle \in (R_{12}(f, M))^*\}$. For each scapegoat generator f and marking M, the f-dependency graph at M is a pair $\langle V, E \rangle$ so that the set of vertices $V = T$, and the set of edges $E = R_{12}(f, M)$.

It directly follows from Definitions 2, 5, and 6 that if $\langle S, T, F, K, W, M_0 \rangle$ is a finite place transition nets with infinite capacities, f is a scapegoat generator of the net, M is a marking of the net, and transition $t \in T$ is enabled at M, then $E^*_{12}(f, M, t)$ is stubborn at M. Moreover, no enabled transition t' in $E^*_{12}(f, M, t)$ can be disabled by the transitions of $T \setminus E^*_{12}(f, M, t)$ because no transition in $T \setminus E^*_{12}(f, M, t)$ can decrease the number of tokens in any input place of t'.

The incremental algorithm in [4] can be described as follows:
Let f be a scapegoat generator. Let M be a marking so that there exists an enabled transition at M. The algorithm [4] produces a set T_s so that for some enabled transition τ at M, $T_s = E^*_{12}(f, M, \tau)$, and $\forall t \in T_s \ M[t\rangle \Rightarrow \tau \in E^*_{12}(f, M, t)$. The enabled transitions of T_s are in one strongly connected component of the f-dependency graph at M. The enabled transitions of T_s are found by traversing the f-dependency graph in depth-first order, starting from an enabled transition, applying Tarjan's algorithm for computing strongly connected components [3], and stopping when the first strongly connected component having an enabled transition is found.

The time complexity of the incremental algorithm is $O(\mu\nu|T|)$, where μ is the maximum number of input places of a transition and ν is the maximum of the maximum number of input transitions of a place and the maximum number of output transitions of a place. μ can be $|S|$, ν can be $|T|$, and $|S|$ can be far greater than $|T|$.

Without change in complexity, the incremental algorithm can be optimized: T_s is chosen to be such $E^*_{12}(f, M, \tau)$ that contains the least number of enabled transitions. (As above, τ has to be enabled at M.) All what is needed is to complete the depth-first search and application of Tarjan's algorithm so that all enabled and only enabled transitions are checked in the outermost loop of the search. If the optimized incremental algorithm is used, the scapegoat generator f is the only nondeterministic factor that affects the number of enabled transitions in T_s.

The reachability graph generation algorithm using the incremental algorithm produces a T_g-reachability graph of the net so that for each marking M in the graph, $T_g(M)$ is the T_s at M if there is an enabled transition at M. (If no transition is enabled at M, then any subset of T is valid for $T_g(M)$.)

It follows from [9] and the remark immediately after Definition 7 that the incremental algorithm produces so strongly stubborn sets that ignoring elimination could be added to the reachability graph generation algorithm.

As a preliminary for the following sections, the meaning of a pseudo-random scapegoat generator is described. An explicit description of a pseudo-random scapegoat generator would be far too complicated to be presented here. Therefore, only an implicit description is given. Let $r_1, r_2, r_3, \ldots$ be an infinite pseudo-random number sequence [3]. Let f be the scapegoat generator to be defined. For each marking M and transition t, $f(M, t)$ is not defined earlier than necessary. If $\langle M, t\rangle$ is the ith pair for which $f(M, t)$ has to be defined, then $f(M, t)$ is defined to be the $((r_i \bmod k) + 1)$th disabling place of t at M, where the list of disabling places of t at M is of length k and determined by some fixed list of the input places of t.

3 An example: data base system

Figure 1 presents a data base system of $n \geq 2$ data base managers. The predicate/transition net in the figure is equivalent to the coloured Petri net in [1]. Resource allocation and a great amount of concurrency makes the system inherently very suitable for the stubborn set method. Let's assume the most obvious unfolding into a place/transition net with infinite capacities. The image of a transition instance of the predicate/transition net is a transition of the place/transition net, the image of a place-tuple pair of the predicate/transition net is a place of the place/transition net, etc. The reachability graph of the place/transition net has $n \cdot 3^{n-1} + 1$ vertices and $2n(1 + (n-1) \cdot 3^{n-2})$ edges [7, 9]. The stubborn set method as such is capable of producing a reduced reachability graph having $2n^2 - n + 1$ vertices and $2n^2$ edges [7, 9]. In the reduced reachability graph in question (unique up to isomorphism), the vertex corresponding to the initial marking has n immediate successors. Every other vertex has one and only one immediate successor. From now on, this reduced reachability graph will be called the Λ-graph.

The rest of this section shows how the scapegoat generator affects the size of the stubborn set found and the size of the reduced reachability graph when the incremental algorithm is used. Four scapegoat generators, called α, β, γ, and δ are presented. α is a pathological scapegoat generator not leading to any reduction. β, γ, and δ look much like α but γ and δ lead to the Λ-graph, and β leads close to the size of the Λ-graph. In addition, pseudo-random scapegoat generators are considered. They tend to be almost as bad as α.

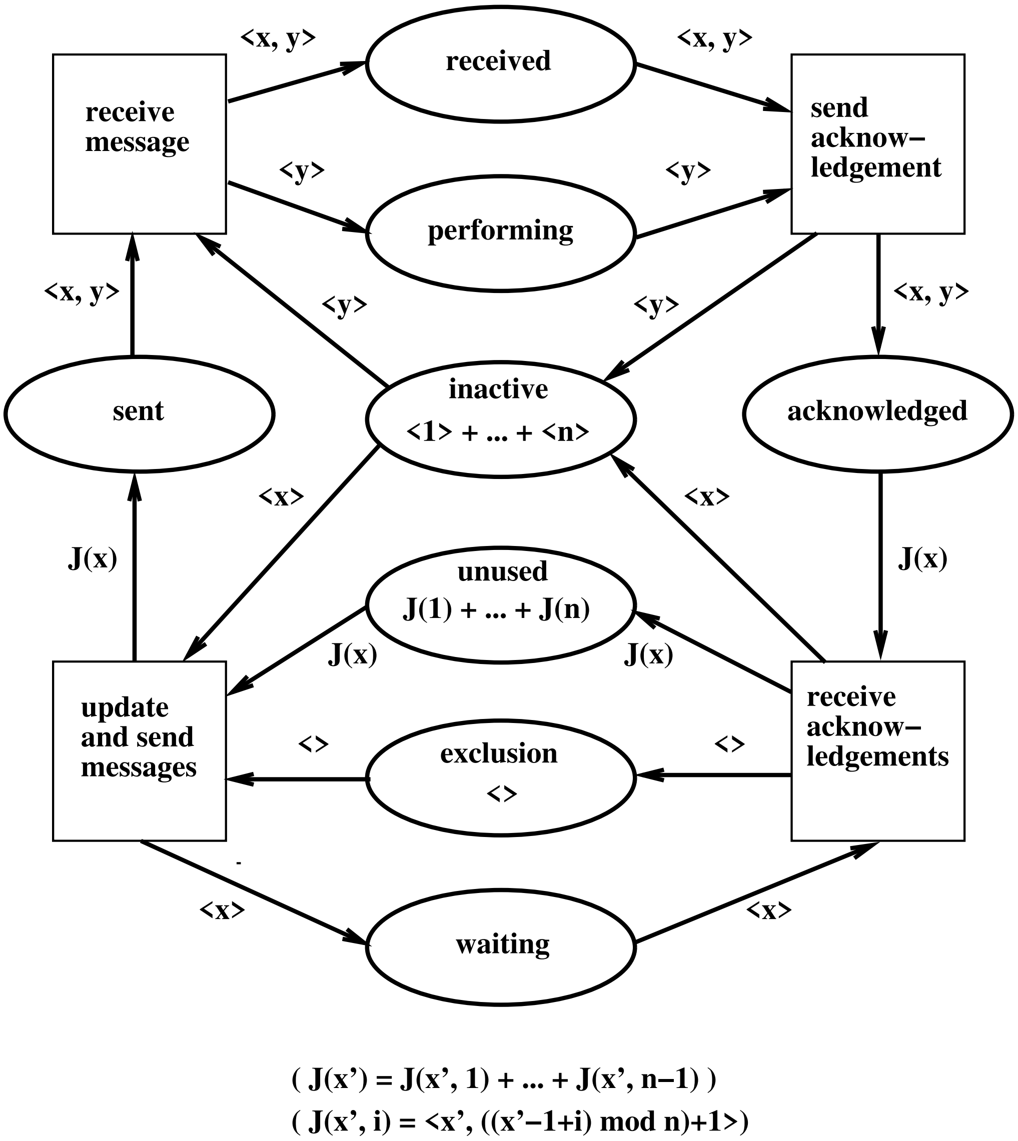

Fig. 1. A data base system.

Scapegoat generator α is defined as follows: $\alpha(\mathrm{mk}', t')$ is defined if and only if transition t' is disabled at marking mk'. Let transition t be disabled at marking mk. Let $\ell(\mathrm{mk}, t, p_1, \ldots, p_j)$ denote the first element in a place list $p_1, \ldots, p_j$ that is a disabling place of t at mk.

$$\alpha(mk,t) = \begin{cases} \ell(\mathrm{mk}, t, \langle s\rangle_{\text{inactive}}, \langle\rangle_{\text{exclusion}}, (M(s,1))_{\text{unused}}, \ldots, (M(s,n-1))_{\text{unused}}) \\ \qquad \text{if } t = \langle s\rangle_{\text{update and send messages}}, \\ \ell(\mathrm{mk}, t, (M(s,1))_{\text{acknowledged}}, \ldots, (M(s,n-1))_{\text{acknowledged}}, \langle s\rangle_{\text{waiting}}) \\ \qquad \text{if } t = \langle s\rangle_{\text{receive acknowledgements}}, \\ \ell(\mathrm{mk}, t, \langle s\rangle_{\text{inactive}}, \langle s,r\rangle_{\text{sent}}) \qquad \text{if } t = \langle s,r\rangle_{\text{receive message}}, \\ \ell(\mathrm{mk}, t, \langle s\rangle_{\text{performing}}, \langle s,r\rangle_{\text{received}}) \;\; \text{if } t = \langle s,r\rangle_{\text{send acknowledgement}}. \end{cases}$$

It is straightforward to show that for each marking mk that is reachable from the initial marking and for each transition t that is enabled at mk, $E^*_{12}(\alpha, \mathrm{mk}, t)$ contains all transitions that are enabled at mk. This means that the incremental algorithm (optimized or not) has no reductive effect. The rotation from s to the next possible manager in the definition of $\alpha(\mathrm{mk}, \langle s\rangle_{\text{receive acknowledgements}})$ is the actual pathological property of α. This kind of stepping from a manager to another manager occurs very often when a pseudo-random scapegoat generator is used. As a result, pseudo-random generators tend to be almost as bad as α.

The pathological property of α can be eliminated by using a fixed manager when possible. Let $H(s,i)$ be $\langle s,i\rangle$ if $s > i$, and $\langle s, i+1\rangle$ if $s \leq i$. Scapegoat generator β is defined as α with the following exception:

$\beta(\mathrm{mk}, t) = \ell(\mathrm{mk}, t, (H(s,1))_{\text{acknowledged}}, \ldots, (H(s,n-1))_{\text{acknowledged}}, \langle s\rangle_{\text{waiting}})$
if $t = \langle s\rangle_{\text{receive acknowledgements}}$ and t is disabled at mk.

When the incremental algorithm (optimized or not) and β are used, the reduced reachability graph has $4(n-2)$ vertices and $8(n-2)$ edges more than the Λ-graph. On the branches where manager 1 or 2 is waiting for acknowledgements, each vertex has exactly one immediate successor. On each of the other $n-2$ branches, the number of vertices having exactly two immediate successors is four, and every other vertex on the branch has exactly one immediate successor. (It is rather straightforward to prove these results.)

Each manager can be considered to be a process. Then those places of the net that represent the states of the managers are the process control places of the net. In scapegoat generator γ, such places have the highest priority. γ is defined as α with the following exception:

$\gamma(\mathrm{mk}, t) = \ell(\mathrm{mk}, t, \langle s\rangle_{\text{waiting}}, (M(s,1))_{\text{acknowledged}}, \ldots, (M(s,n-1))_{\text{acknowledged}})$
if $t = \langle s\rangle_{\text{receive acknowledgements}}$ and t is disabled at mk.

Scapegoat generator δ has been got by considering that if a transition has a unique characteristic input place, then that place should have the highest priority. δ is not pure in that sense but shows the sufficient interchange to transform α into an optimal scapegoat generator. δ is defined as α with the following exception:

$\delta(\mathrm{mk}, t) = \ell(\mathrm{mk}, t, \langle s,r\rangle_{\text{received}}, \langle s\rangle_{\text{performing}})$
if $t = \langle s,r\rangle_{\text{send acknowledgement}}$ and t is disabled at mk.

It is straightforward to show that for each non-initial marking mk that is reachable from the initial marking and for each transition t that is enabled at mk, the set of enabled transitions in $E^*_{12}(\gamma, \mathrm{mk}, t)$ is $\{t\}$, and the set of enabled transitions in $E^*_{12}(\delta, \mathrm{mk}, t)$ is $\{t\}$. As a consequence, the incremental algorithm (optimized or not) produces the Λ-graph.

4 Strategies for choosing a scapegoat

Section 3 suggests three strategies for choosing a scapegoat generator. Scape goat generator β suggests absolute ordering with respect to identity numbers, γ suggests giving a process control place the highest priority, and δ suggests giving a unique characteristic input place the highest priority. All these strategies are fixed order strategies in the sense that always the first possible alternative in a fixed list is chosen.

Some strategies work without knowledge of the modelled system. One of such strategies minimizes the number of such enabled immediate successors of a vertex that are not in any strongly connected component already found in the dependency graph. On the second priority level, it minimizes the number of all immediate successors of the vertex that are not in any strongly connected component already found. On the third priority level, it minimizes the number of those immediate successors of the vertex that have not been visited yet.

A pseudo-random scapegoat generator is probably far from optimal if the incremental algorithm is as instable as in Section 3. On the other hand, it is often useful to compare a given strategy to a pseudo-random strategy to see how good the strategy is.

The deletion algorithm can be used to estimate how good the incremental algorithm could be even though the deletion algorithm may sometimes produce a stubborn set containing more enabled transitions than a stubborn set produced by the incremental algorithm. The choice of an enabled transition to be deleted is a nondeterministic factor [5, 7]. In the case of the net of Section 3, the nondeterminism does not affect the number of enabled transitions in the produced stubborn set.

5 Conclusion

The incremental algorithm for computing stubborn sets for finite place/transition nets with infinite capacities may produce stubborn sets that contain unnecessarily many enabled transitions because the algorithm contains nondeterministic choices. The need for choosing the starting transition can be eliminated without affecting the complexity of the algorithm by visiting all the enabled transitions. Then the choice of a disabling place (the scapegoat) is the only nondeterministic factor that affects the number of enabled transition in the result set of the algorithm.

The considered example suggests three fixed order strategies for choosing a scapegoat: absolute ordering with respect to identity numbers, giving a process control place the highest priority, and giving a unique characteristic input place the highest priority. It is also easy to develop simple strategies that work without knowledge of the modelled system. To see how good a given strategy is, one can compare the strategy to a pseudo-random strategy. The deletion algorithm can be used to estimate how good the incremental algorithm could be even though the deletion algorithm may sometimes produce a stubborn set containing more enabled transitions than a stubborn set produced by the incremental algorithm.

References

1. Jensen, K.: *Coloured Petri Nets and the Invariant Method.* Theoretical Computer Science 14 (1981), pp. 317–336.
2. Rauhamaa, M.: *A Comparative Study of Methods for Efficient Reachability Analysis.* Helsinki University of Technology, Digital Systems Laboratory Report A 14, Espoo 1990, 61 p.

3. Sedgewick, R.: *Algorithms.* Addison-Wesley, Reading MA 1983, 551 p.
4. Valmari, A.: *Error Detection by Reduced Reachability graph generation.* Proceedings of the Ninth European Workshop on Application and Theory of Petri Nets, Venice 1988, pp. 95–112.
5. Valmari, A.: *Heuristics for Lazy State Space Generation Speeds up Analysis of Concurrent Systems.* Mäkelä, M., Linnainmaa, S., and Ukkonen, E. (ed.), Proceedings of the Finnish Artificial Intelligence Symposium (Suomen tekoälytutkimuksen päivät), Vol. 2, Helsinki 1988, pp. 640–650.
6. Valmari, A.: *Some Polynomial Space Complete Concurrency Problems.* Tampere University of Technology, Software Systems Laboratory Report 4, Tampere 1988, 34 p.
7. Valmari, A.: *State Space Generation: Efficiency and Practicality.* Doctoral thesis, Tampere University of Technology Publications 55, Tampere 1988, 170 p.
8. Valmari, A.: *Eliminating Redundant Interleavings during Concurrent Program Verification.* Proceedings of Parallel Architectures and Languages Europe '89 Vol. 2, Lecture Notes in Computer Science 366, Springer-Verlag, Berlin 1989, pp. 89–103.
9. Valmari, A.: *Stubborn Sets for Reduced State Space Generation.* Proceedings of the 10th International Conference on Application and Theory of Petri Nets, Vol. 2, Bonn 1989, pp. 1–22.
10. Valmari, A.: *A Stubborn Attack on State Explosion.* Proceedings of the Workshop on Computer-Aided Verification, Rutgers University, Vol. 1, DIMACS Technical Report 90–31, New Brunswick NJ 1990, 15 p.
11. Valmari, A.: *Stubborn Sets of Coloured Petri Nets.* Proceedings of the 12th International Conference on Application and Theory of Petri Nets, Gjern, Denmark 1991, pp. 102–121.
12. Varpaaniemi, K., and Rauhamaa, M.: *The Stubborn Set Method in Practice.* Jensen, K. (Ed.), Proceedings of the 13th International Conference on Application and Theory of Petri Nets, Sheffield 1992. Lecture Notes in Computer Science 616, Springer-Verlag, Berlin 1992, pp. 389–393.