Dynamically Stubborn Sets and the Sleep Set Method

Kimmo Varpaaniemi

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

Abstract. *Reachability analysis* is a powerful formal method for analysis of concurrent and distributed finite state systems. It suffers from the *state space explosion problem*, however: the state space of a system can be far too large to be completely generated. This paper considers two promising methods, Valmari's *stubborn set method* and Godefroid's *sleep set method*, to avoid generating all of the state space when searching for *undesirable reachable terminal states*, also called *deadlocks*. What makes deadlocks especially interesting is the fact that the verification of a *safety property* can often be reduced to deadlock detection. The considered methods utilize the *independence of transitions* to cut down on the number of states inspected during the search. These methods have been *combined* by Godefroid, Pirottin, and Wolper to further reduce the number of inspected states.

Petri nets are a widely used model for concurrent and distributed systems. This paper shows that the stubborn set method and the sleep set method can be combined without any of the assumptions previously placed on the stubborn sets as far as the detection of reachable terminal states in *place/transition nets*, a class of Petri nets, is concerned. The obtained result is actually more general and gives a sufficient condition for a method to be compatible with the sleep set method in the detection of reachable terminal states in place/transition nets.

This paper emphasizes the value of *dynamically stubborn sets* as a useful generalization of stubborn sets and presents some results that improve the understanding of the stubborn set method.

1 Introduction

A classic way to detect errors in a system is *testing*. However, it is often difficult to test the system sufficiently even if the system is sequential. If the system is concurrent or distributed, there can be errors that depend on the order of execution of actions in the system. When a test is executed twice in exactly the same circumstances, such errors may remain undetected or occur only in one of the executions.

Reachability analysis inspects the states of a formal abstract model of the system, aiming to find even such elusive errors. The *complete state space* can be seen as a graph having the states that are reachable from a given initial state of the model as vertices, and all the state transitions between the states as edges. Many properties can easily be checked from such graph if it is finite. If the complete state space is infinite, it is still possible to detect errors by inspecting some finite set of states. On the other hand, a finite complete state space can be far too large with respect to the time and other resources needed to inspect all states in the space. The number of states in the complete state space may grow exponentially or superexponentially with respect to some parameter of the model. We thus have the so called *state space explosion problem.*

Fortunately, many properties can be verified without inspecting all reachable states. For example, reachable terminal states can sometimes be found by inspecting only some of the paths from the initial state to the terminal states. A terminal state can be *acceptable* or *undesirable. Undesirable reachable terminal states* are often called *deadlocks*. Godefroid and Wolper [5] among others have shown how the verification of a *safety property* can often be reduced to the detection of deadlocks. Intuitively, a safety property states what should not happen whereas a *liveness property* states what should happen in the modelled system. A detected error such as a deadlock may be caused by an improper design of the modelled system but it is also possible that the model is improper. The problem whether the model corresponds to the modelled system properly is a challenging area of research. We shall not pursue it further in this paper, however.

Valmari's *stubborn set method* [9, 10, 11, 12] and Godefroid's *sleep set method* [1, 3, 4, 5, 2, 14, 15] utilize the *independence of state transitions* of the model to eliminate such paths of the complete state space that are redundant with respect to the verification of a given property. These two methods have been combined by Wolper and Godefroid [14], Godefroid and Pirottin [3], and Wolper, Godefroid, and Pirottin [15] to further reduce the number of states inspected during the search. We shall study the stubborn set method, the sleep set method, and their combination in this paper. We are mainly interested in finding all reachable terminal states by inspecting as few states as possible. All of these methods guarantee that all reachable terminal states are found if the complete state space is finite. It is possible to extend the methods to verify more sophisticated properties, even properties expressed as linear temporal logic formulae [4, 11], but the more properties of the complete state space are preserved, the greater is the number of states inspected during verification. The state space explosion problem often appears even if we limit ourselves to detecting of reachable terminal states.

Petri nets [8] are a widely used model for concurrent and distributed systems. This paper concentrates on the detection of reachable terminal states in *place/transition nets* [8], a class of Petri nets.

In Section 2, we introduce place/transition nets. The presentation does not go beyond what is necessary for the remaining sections. In Section 3, dynamically stubborn sets [7, 12] are shown to be a useful generalization of stubborn sets. Section 4 considers the sleep set method and its combination with the stubborn set method. We conclude in Section 5 by summarizing the results obtained and briefly discussing possible directions for future research.

2 Place/Transition Nets

In this section we give definitions of *place/transition nets* [8] that will be used in later sections.

We shall use "iff" to denote "if and only if". The *power set* (the set of subsets) of a set A is denoted by 2^A. The set of *(total) functions* from a set A to a set B is denoted by $(A \to B)$. The set of natural numbers, including 0, is denoted by N. We shall use ω to denote a formal infinite number, and N_ω to denote $N \cup \{\omega\}$. Relation $\leq$ over N is extended to N_ω by defining

$$\forall n \in N_\omega \;\; n \leq \omega.$$

Addition and subtraction are extended similarly by defining

$$\forall n \in N \;\; \omega + n = \omega \land \omega - n = \omega.$$

Clearly, $\omega \notin N$ since no natural number can be substituted for ω in these conditions in such a way that the conditions would hold.

Definition 1. A *place/transition net* is a 6-tuple $\langle S, T, F, K, W, M_0 \rangle$ such that

- S is the set of *places*,
- T is the set of *transitions*, $S \cap T = \emptyset$,
- F is the set of *arcs*, $F \subseteq (S \times T) \cup (T \times S)$,
- K is the *capacity function*, $K \in (S \to N_\omega)$,
- W is the *arc weight function*, $W \in (F \to (N \setminus \{0\}))$, and
- M_0 is the *initial marking (initial state)*, $M_0 \in \mathcal{M}$ where $\mathcal{M}$ is the set of *markings (states)*, $\mathcal{M} = \{M \in (S \to N) \mid \forall s \in S \ M(s) \leq K(s)\}$.

If $x \in S \cup T$, then the set of *input elements* of x is

$$^\bullet x = \{y \mid \langle y, x \rangle \in F\},$$

the set of *output elements* of x is

$$x^\bullet = \{y \mid \langle x, y \rangle \in F\},$$

and the set of *adjacent elements* of x is $x^\bullet \cup {}^\bullet x$. The function W is extended to a function in $(((S \times T) \cup (T \times S)) \to N)$ by defining $W(x, y) = 0$ iff $\langle x, y \rangle \notin F$. The net is *finite* iff $S \cup T$ is finite.

Unlike Reisig [8], we do not accept $M(s) = \omega$. Such markings would be redundant in finite place/transition nets.

Definition 2. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. A transition t is *enabled at a marking M* iff

$$\forall s \in {}^\bullet t \ M(s) \geq W(s, t)$$

and

$$\forall s \in t^\bullet \ M(s) - W(s, t) + W(t, s) \leq K(s).$$

A transition t *leads (can be fired) from a marking M to a marking M'* ($M[t\rangle M'$ for short) iff t is enabled at M and

$$\forall s \in S \ M'(s) = M(s) - W(s, t) + W(t, s).$$

A transition t is *disabled at a marking M* iff t is not enabled at M. A marking M is *terminal* iff no transition is enabled at M. A marking M is *nonterminal* iff M is not terminal.

Our enabledness condition is weaker than Reisig's enabledness condition [8] that requires $M(s) + W(t, s) \leq K(s)$ instead of $M(s) - W(s, t) + W(t, s) \leq K(s)$.

Finite transition sequences and reachability are introduced in Definition 3. We shall use ε to denote the empty sequence.

Definition 3. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. For any $T_s \subseteq T$,

$$T_s^0 = \{\varepsilon\},$$
$$(\forall n \in N \ T_s^{n+1} = \{\sigma t \mid \sigma \in T_s^n \wedge t \in T_s\}), \text{ and}$$
$$T_s^* = \{\sigma \mid \exists n \in N \ \sigma \in T_s^n\}.$$

T_s^* is called the *set of finite sequences of transitions in T_s*, and T^* is called the *set of finite transition sequences of the net*. A finite transition sequence σ' is a *prefix* of a finite transition sequence σ iff there exists a finite transition sequence σ'' such that $\sigma = \sigma' \sigma''$. A finite transition sequence σ *leads (can be fired) from a marking M to a marking M'* iff $M[\sigma\rangle M'$ where

$$\forall M \in \mathcal{M} \ M[\varepsilon\rangle M, \text{ and}$$

$$\forall M \in \mathcal{M} \ \forall M' \in \mathcal{M} \ \forall \delta \in T^* \ \forall t \in T \ M[\delta t\rangle M' \Leftrightarrow (\exists M'' \in \mathcal{M} \ M[\delta\rangle M'' \wedge M''[t\rangle M').$$

A finite transition sequence σ is *enabled at a marking M* ($M[\sigma\rangle$ for short) iff σ leads from M to some marking. A finite transition sequence σ is *disabled at a marking M* iff σ is not enabled at M. A marking M' is *reachable from a marking M* iff some finite transition sequence leads from M to M'. A marking M' is a *reachable marking* iff M' is reachable from M_0. A marking M' is *globally unreachable* iff M' is not reachable from any other marking in $\mathcal{M}$ than M'. The *(full) reachability graph* of the net is the pair $\langle V, A\rangle$ such that the set of vertices V is the set of reachable markings, and the set of edges A is

$$\{\langle M,t,M'\rangle \mid M \in V \wedge M' \in V \wedge t \in T \wedge M[t\rangle M'\}.$$

A finite transition sequence is merely a string. It can be thought of as occurring as a path in the full reachability graph iff it is enabled at some reachable marking.

Definition 4. Let $\langle S,T,F,K,W,M_0\rangle$ be a place/transition net. Let f be a function from $\mathcal{M}$ to 2^T. A finite transition sequence σ *f-leads (can be f-fired) from a marking M to a marking M'* iff $M[\sigma\rangle_f M'$, where

$$\forall M \in \mathcal{M}\ M[\varepsilon\rangle_f M, \text{ and}$$

$$\forall M \in \mathcal{M}\ \forall M' \in \mathcal{M}\ \forall \delta \in T^*\ \forall t \in T$$
$$M[\delta t\rangle_f M' \Leftrightarrow$$
$$(\exists M'' \in \mathcal{M}\ M[\delta\rangle_f M'' \wedge$$
$$t \in f(M) \wedge M''[t\rangle M').$$

A finite transition sequence σ is *f-enabled at a marking M* ($M[\sigma\rangle_f$ for short) iff σ f-leads from M to some marking. A marking M' is *f-reachable from a marking M* iff some finite transition sequence f-leads from M to M'. A marking M' is an *f-reachable marking* iff M' is f-reachable from M_0. The *f-reachability graph* of the net is the pair $\langle V, A\rangle$ such that the set of vertices V is the set of f-reachable markings, and the set of edges A is

$$\{\langle M,t,M'\rangle \mid M \in V \wedge M' \in V \wedge t \in f(M) \wedge M[t\rangle M'\}.$$

Definition 4 is like a part of Definition 3 except that a transition selection function f determines which transitions are fired. If f is clear from the context or is implicitly assumed to exist and be of a kind that is clear from the context, then the f-reachability graph of the net is called the *reduced reachability graph* of the net. Note that the reduced reachability graph of the net can even be the full reachability graph of the net, e.g. in the case where $f(M) = T$ for each $M \in \mathcal{M}$.

Definition 5. Let $\langle S,T,F,K,W,M_0\rangle$ be a place/transition net. The *set of infinite transition sequences of the net* is the set of functions from N to T, $(N \to T)$. The function ς from $(N \to T) \times N$ to T^* is defined by

$$(\forall \sigma \in (N \to T)\ \varsigma(\sigma,0) = \varepsilon), \quad \text{and}$$
$$(\forall \sigma \in (N \to T)\ \forall n \in N$$
$$\varsigma(\sigma, n+1) = \varsigma(\sigma,n)\sigma(n)).$$

If σ is an infinite transition sequence and $n \in N$, $\varsigma(\sigma,n)$ is called the *prefix of length n of σ*. An infinite transition sequence σ is *enabled at a marking M* ($M[\sigma\rangle$ for short) iff for each $n \in N$, the prefix of length n of σ is enabled at M. An infinite transition sequence σ is *disabled at a marking M* iff σ is not enabled at M. Let f be a function from $\mathcal{M}$ to 2^T. An infinite transition sequence σ is *f-enabled at a marking M* ($M[\sigma\rangle_f$ for short) iff for each $n \in N$, the prefix of length n of σ is f-enabled at M.

An infinite transition sequence is merely a function. It can be thought of as occurring as a path in the full reachability graph iff it is enabled at some reachable marking.

Definition 6. Let $\langle S,T,F,K,W,M_0\rangle$ be a place/transition net. A transition sequence δ is an *alternative sequence of a finite transition sequence σ at a marking M* iff δ is a finite transition sequence, σ is enabled at M, and δ leads from M to the same marking as σ. A transition sequence δ is a *length-secure alternative sequence of a finite transition sequence σ at a marking M* iff δ is an alternative sequence of σ at M and not longer than σ. The functions η and ϑ from $T^* \times \mathcal{M}$ to $2^{(T^*)}$ are defined as follows: for each finite transition sequence σ and marking M, $\eta(\sigma, M)$ is the *set of alternative sequences of σ at M*, and $\vartheta(\sigma, M)$ is the *set of length-secure alternative sequences of σ at M.*

Clearly, for each finite transition sequence σ and marking M, $\vartheta(\sigma, M) \subseteq \eta(\sigma, M)$. Also, $\eta(\sigma, M)$ is empty iff σ is not enabled at M.

Definition 7. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. A transition sequence δ is a *permutation of a finite transition sequence* σ iff δ is a finite transition sequence and for each transition t, the number of t's in δ is equal to the number of t's in σ. A transition sequence δ is an *enabled permutation of a finite transition sequence σ at a marking M* iff δ is a permutation of σ and enabled at M. The function π from $T^* \times \mathcal{M}$ to $2^{(T^*)}$ is defined as follows: for each finite transition sequence σ and marking M, $\pi(\sigma, M)$ is the set of enabled permutations of σ at M.

Clearly, if finite transition sequences are enabled permutations of each other at a marking M, they lead to the same marking from M. So, if a finite transition sequence σ is enabled at a marking M, then $\pi(\sigma, M) \subseteq \vartheta(\sigma, M)$. The set $\pi(\sigma, M)$ can be nonempty even if σ is not enabled at M since some permutation of σ can be enabled at M. The set of length-secure alternative sequences, as well as the set of alternative sequences, of an enabled finite transition sequence σ at a marking can always be partitioned into sets of enabled permutations of sequences at the marking. Of course, only one of those sets is the set of enabled permutations of σ.

Definition 8. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Transitions t and t' *commute at a marking M* iff $M[tt'\rangle$ and $M[t't\rangle$. Transitions t and t' are *independent at a marking M* iff

$$(M[tt'\rangle \wedge M[t't\rangle) \vee (\neg M[t\rangle \wedge \neg M[t'\rangle) \vee$$
$$(M[t\rangle \wedge \neg M[t'\rangle \wedge \neg M[tt'\rangle) \vee$$
$$(M[t'\rangle \wedge \neg M[t\rangle \wedge \neg M[t't\rangle).$$

Our definition of independence corresponds to Godefroid's and Pirottin's [3] definition of conditional independence which in turn is based on Katz's and Peled's [6] corresponding definition. Our definition of independence can be obtained from Godefroid's and Pirottin's definition of valid conditional dependency relations, Definition 5 in [3], by taking the necessary conditions for a triple of two transitions and one state to be in the complement of a valid dependency relation, and substituting terms of place/transition nets for the terms of the model of concurrency in [3] in an obvious way.

The following can clearly be seen from the above.

- Different transitions are independent at a marking iff neither of them can be fired at the marking making the other transition turn from enabled to disabled or from disabled to enabled.
- A transition t commutes with itself at a marking iff tt is enabled at the marking.
- A transition t is independent of itself at a marking iff tt is enabled or t is disabled at the marking.
- Transitions commute at a marking iff they are enabled and independent at the marking.

Definition 9. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. A transition sequence δ is a *neighbour of a finite transition sequence* σ iff there exist transitions t and t', and finite transition sequences σ' and σ'' such that $\sigma = \sigma' t t' \sigma''$ and $\delta = \sigma' t' t \sigma''$. A transition sequence δ is an *enabled neighbour of a finite transition sequence σ at a marking M* iff δ is a neighbour of σ and enabled at M. Let M be a marking and R the binary relation on T^* such that $\sigma R \delta$ iff σ and δ are enabled neighbours of each other at M. The *conditional trace of a finite transition sequence σ at M* is the set of finite transition sequences such that a sequence δ is in the conditional trace of σ at M iff σ is enabled at M and $\sigma R^* \delta$ where R^* is the reflexive-transitive closure of R. A set is a *conditional trace at M* iff the set is the conditional trace of some finite transition sequence at M. The function ι from $T^* \times \mathcal{M}$ to $2^{(T^*)}$ is defined as follows: for each finite transition sequence σ, and marking M, $\iota(\sigma, M)$ is the conditional trace of σ at M.

In other words, a conditional trace is a set of enabled finite transition sequences at a marking that can be obtained from each other by repeatedly interchanging adjacent independent transitions. We did not have to mention independence in Definition 9 since transitions commute at a marking iff they are enabled and independent at the marking. The reflexive-transitive closure of R in Definition 9 is clearly an equivalence relation, and the conditional trace of an enabled finite transition sequence is the equivalence class of the sequence with respect to the equivalence relation. Our definition of a conditional trace corresponds to Godefroid's and Pirottin's [3] definition which in turn is based on Katz's and Peled's [6] corresponding definition. The conditional trace of an enabled finite transition sequence at a marking is naturally a subset of the enabled permutations of the sequence at the marking. Thus $\iota(\sigma, M) \subseteq \pi(\sigma, M)$ holds for each σ and M. Moreover, the set of enabled permutations of an enabled finite transition sequence at a marking can always be partitioned into conditional traces at the marking. Of course, only one of those conditional traces is the conditional trace of the sequence. Note that if a finite transition sequence σ is disabled at a marking M, then the conditional trace of σ at M is empty.

Figure 1 presents the functions η, ϑ, π, and ι in a nutshell.

$\eta(\sigma, M)$	the set of alternative sequences of a finite transition sequence σ at M
$\vartheta(\sigma, M)$	the set of length-secure alternative sequences of a finite transition sequence σ at M
$\pi(\sigma, M)$	the set of enabled permutations of a finite transition sequence σ at M
$\iota(\sigma, M)$	the conditional trace of a finite transition sequence σ at M

Fig. 1. The functions η, ϑ, π, and ι.

Definition 10. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let f be a function from $\mathcal{M}$ to 2^T. Then we say that f *represents all sets of alternative sequences to terminal markings* iff

$$\forall \sigma \in T^* \; \forall M \in \mathcal{M} \; (M[\sigma\rangle \wedge \forall t \in T \; \neg M[\sigma t\rangle) \Rightarrow (\exists \delta \in \eta(\sigma, M) \; M[\delta\rangle_f).$$

Correspondingly, f *represents all sets of length-secure alternative sequences to terminal markings* iff

$$\forall \sigma \in T^* \; \forall M \in \mathcal{M} \; (M[\sigma\rangle \wedge \forall t \in T \; \neg M[\sigma t\rangle) \Rightarrow (\exists \delta \in \vartheta(\sigma, M) \; M[\delta\rangle_f).$$

Respectively, f *represents all sets of enabled permutations to terminal markings* iff

$$\forall \sigma \in T^* \; \forall M \in \mathcal{M} \; (M[\sigma\rangle \wedge \forall t \in T \; \neg M[\sigma t\rangle) \Rightarrow (\exists \delta \in \pi(\sigma, M) \; M[\delta\rangle_f).$$

Finally, f *represents all conditional traces to terminal markings* iff

$$\forall \sigma \in T^* \; \forall M \in \mathcal{M} \; (M[\sigma\rangle \wedge \forall t \in T \; \neg M[\sigma t\rangle) \Rightarrow (\exists \delta \in \iota(\sigma, M) \; M[\delta\rangle_f).$$

The following can clearly be seen from the above.

- A function reperesenting all conditional traces to terminal markings represents all sets of enabled permutations to terminal markings.
- A function representing all sets of enabled permutations to terminal markings represents all sets of length-secure alternative sequences to terminal markings.

- A function representing all sets of length-secure alternative sequences to terminal markings represents all sets of alternative sequences to terminal markings.

3 Dynamically Stubborn Sets

This section is concentrated on *dynamically stubborn* sets [7, 12]. All the stubborn sets that have been defined in the literature are known to be dynamically stubborn. Dynamically stubborn sets seem to have all the nice properties of (statically) stubborn sets except that the definition of dynamic stubbornness does not seem to imply a practical algorithm for computing dynamically stubborn sets.

In Subsection 3.1, we define different levels of dynamic stubbornness and present some small but interesting results concerning dynamically stubborn sets. We base the definitions on Rauhamaa's principles [7]. We also present Godefroid's and Pirottin's definitions of *persistent* and *conditionally stubborn* sets [3] in the context of place/transition nets.

In Subsection 3.2, we show that dynamically stubborn sets are really useful. We consider how reachable terminal markings are preserved by a *dynamically stubborn set selective reachability graph generation*, that is, reachability graph generation that at each encountered marking selects a dynamically stubborn set and fires the enabled transitions in the set, without firing other transitions. All reachable terminal markings are shown to occur in the reduced reachability graph. Also, if there is an infinite path in the full reachability graph, then the reduced reachability graph is shown to have an infinite path, too. We end Subsection 3.2 by presenting a property which is the basis for Valmari's algorithms for detecting *ignored transitions* and eliminating the *ignoring phenomenon* [10]. A transition is ignored at a marking iff the transition is enabled at the marking but not fired at any marking that is reachable from the marking. The existence of ignored transitions is called the ignoring phenomenon.

3.1 Dynamic Stubbornness

We define dynamic stubbornness on the basis of Rauhamaa's principles [7].

Definition 11. Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let M be a marking of the net. A set $T_s \subseteq T$ *fulfils the first principle of dynamic stubbornness* (*D1* for short) *at* M iff

$$\forall \sigma \in (T \setminus T_s)^* \ \forall t \in T_s \ \ M[\sigma t\rangle \Rightarrow M[t\sigma\rangle.$$

A transition t is a *key transition of a set* $T_s \subseteq T$ *at* M iff $t \in T_s$ and

$$\forall \sigma \in (T \setminus T_s)^* \ \ M[\sigma\rangle \Rightarrow M[\sigma t\rangle.$$

A set $T_s \subseteq T$ *fulfils the second principle of dynamic stubbornness* (*D2* for short) *at* M iff T_s has a key transition at M. A set $T_s \subseteq T$ *fulfils the principle of conventional dynamic stubbornness* (*CD* for short) *at* M iff

$$\forall \sigma \in (T \setminus T_s)^* \ \forall \delta \in (T \setminus T_s)^* \ \forall t \in T_s \ \ M[\sigma\delta t\rangle \Rightarrow M[\sigma t\delta\rangle.$$

A set $T_s \subseteq T$ *fulfils the first principle of strong dynamic stubbornness* (*SD1* for short) *at* M iff

$$\forall \sigma \in (T \setminus T_s)^* \ \forall t \in T_s \ \ M[\sigma t\rangle \Rightarrow M[t\rangle.$$

A set $T_s \subseteq T$ *fulfils the second principle of strong dynamic stubbornness* (*SD2* for short) *at* M iff

$$\forall \sigma \in (T \setminus T_s)^* \ \forall t \in T_s \ \ (M[t\rangle \wedge M[\sigma\rangle) \Rightarrow (M[\sigma t\rangle \wedge M[t\sigma\rangle).$$

A set $T_s \subseteq T$ is *dynamically stubborn at* M iff T_s fulfils D1 and D2 at M. A set $T_s \subseteq T$ is *conventionally dynamically stubborn at* M iff T_s fulfils CD and D2 at M. A set $T_s \subseteq T$ is *unconventionally dynamically stubborn at* M iff T_s is dynamically stubborn but not conventionally dynamically stubborn at M. A set $T_s \subseteq T$ is *strongly dynamically stubborn at* M iff T_s fulfils SD1 and SD2 at M and $\exists t \in T_s \ M[t\rangle$.

The principles D1, D2, CD, SD1, and SD2 are illustrated in Figure 2. The principles D1, D2, SD1, and SD2 are Rauhamaa's Principles 1^*, 2^*, 1, and 2, respectively [7]. Clearly, a key transition of a set at a marking is enabled at the marking. Our key transitions are similar to Valmari's key transitions [10]. The difference is that Valmari's key transitions satisfy a condition that can be checked easily and is sufficient but not necessary for a transition to be a key transition in the sense of our definition.

We shall see in Subsection 3.2 that dynamic stubbornness alone is sufficient as far as the detection of reachable terminal markings is concerned, conventional dynamic stubbornness is related to conditional traces, and strongly dynamically stubborn sets are useful when one wants to eliminate the ignoring phenomenon. The term "conventional" refers to the often used heuristic that every sequence of such transitions that are not in a given stubborn set should leave the set stubborn. Valmari has shown that the heuristic has some advantages when the so called *candidate list algorithm* is used for computing stubborn sets [9]. Valmari has also defined dynamically stubborn sets [12]. Valmari's dynamically stubborn sets are considered later in this subsection.

Lemma 12. *If a set is conventionally dynamically stubborn at a marking, the set is dynamically stubborn set at the marking. A set is strongly dynamically stubborn at a marking iff the set is dynamically stubborn at the marking and each enabled transition in the set is a key transition of the set at the marking.*

Proof. See the proof of Lemma 3.2 in [13]. □

Lemma 13. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let M be a marking of the net. A set $T_s \subseteq T$ fulfils SD2 at M iff*

$$\forall \sigma \in (T \setminus T_s)^* \; \forall \delta \in (T \setminus T_s)^* \; \forall t \in T_s$$
$$(M[t\rangle \wedge M[\sigma\delta\rangle) \Rightarrow M[\sigma t \delta\rangle.$$

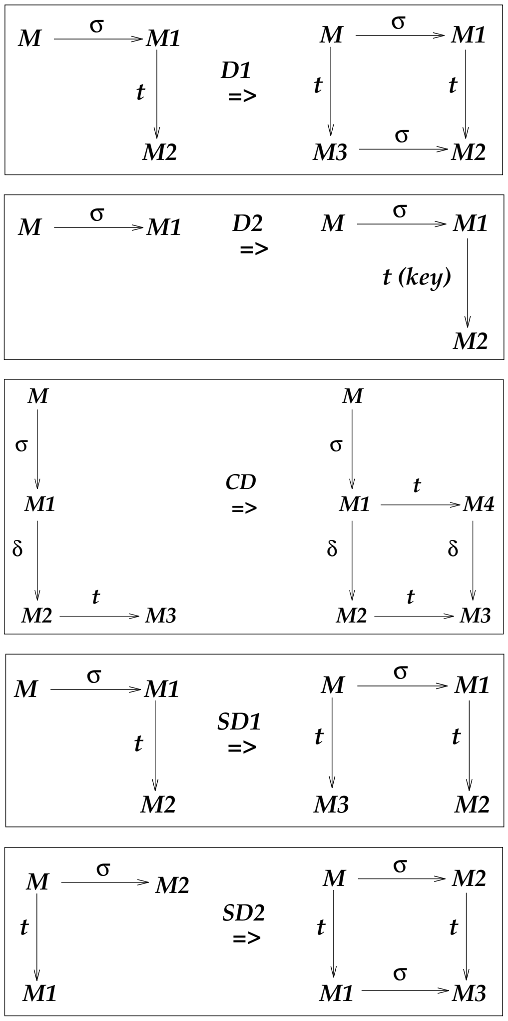

Fig. 2. The principles of dynamic, conventional dynamic, and strong dynamic stubbornness.

Proof. See the proof of Lemma 3.3 in [13]. □

Lemma 14. *If a set is strongly dynamically stubborn at a marking, the set is conventionally dynamically stubborn at the marking.*

Proof. See the proof of Lemma 3.4 in [13]. □

Lemma 15. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let $M \in \mathcal{M}$. A set $T_s \subseteq T$*

is strongly dynamically stubborn at M iff

$\forall \sigma \in (T \setminus T_s)^* \ \forall M' \in \mathcal{M}$
$M[\sigma\rangle M' \Rightarrow$
$(T_s$ *is strongly dynamically stubborn at* $M')$.

Proof. See the proof of Lemma 3.5 in [13]. □

The result in Lemma 15 is new though inspired by Valmari [10]. Lemma 15 states that every sequence of such transitions that are not in a given strongly dynamically stubborn set leaves the set strongly dynamically stubborn. Lemma 16 states the similar result for conventionally dynamically stubborn sets.

Lemma 16. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let $M \in \mathcal{M}$. A set $T_s \subseteq T$ is conventionally dynamically stubborn at M iff*

$\forall \sigma \in (T \setminus T_s)^* \ \forall M' \in \mathcal{M}$
$M[\sigma\rangle M' \Rightarrow$
$(T_s$ *is conventionally dynamically stubborn at* $M')$.

Proof. See the proof of Lemma 3.6 in [13]. □

The result in Lemma 16 is new though inspired by Valmari [10]. There is no lemma analogous to Lemmas 15 and 16 for all dynamically stubborn sets. Lemma 17 states that a set is conventionally dynamically stubborn iff the set is dynamically stubborn and every sequence of such transitions that are not in the set leaves the set dynamically stubborn.

Lemma 17. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let $M \in \mathcal{M}$. A set $T_s \subseteq T$ is conventionally dynamically stubborn at M iff*

$\forall \sigma \in (T \setminus T_s)^* \ \forall M' \in \mathcal{M}$
$M[\sigma\rangle M' \Rightarrow$
$(T_s$ *is dynamically stubborn at* $M')$.

Proof. See the proof of Lemma 3.7 in [13]. □

The result in Lemma 17 is new though inspired by Valmari [9].

Lemma 18. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let M be a marking of the net, and T_s and T_e subsets of T such that*

$$\{t \in T_s \mid M[t\rangle\} \subseteq T_e, \text{ and } T_e \subseteq T_s.$$

If T_s is dynamically stubborn at M, T_e is dynamically stubborn at M. If T_s is conventionally dynamically stubborn at M, T_e is conventionally dynamically stubborn at M. If T_s is strongly dynamically stubborn at M, T_e is strongly dynamically stubborn at M.

Proof. See the proof of Lemma 3.8 in [13]. □

The result in Lemma 18 is new though inspired by Godefroid and Pirottin [3]. Lemma 18 states that if we remove disabled transitions from a dynamically stubborn (conventionally dynamically stubborn, strongly dynamically stubborn) set, the remaining set is dynamically stubborn (conventionally dynamically stubborn, strongly dynamically stubborn). For example, if a dynamically stubborn set is minimal with respect to set inclusion, by Lemma 18 the set consists of enabled transitions only.

We define persistence and conditional stubbornness in such a way that the definitions correspond to the definitions given by Godefroid and Pirottin [3]. Our definitions can be obtained from Godefroid's and Pirottin's Definitions 7 and 8 in [3] by substituting terms of place/transition nets for the terms of the model of concurrency in [3] in an obvious way.

Definition 19. Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let $M \in \mathcal{M}$. A set $T_s \subseteq T$ *fulfils the principle of persistence and conditional stubbornness* (*PE* for short) *at M* iff

$\forall \sigma \in (T \setminus T_s)^* \ \forall t \in T_s$
$\forall t' \in T \setminus T_s \ \forall M' \in \mathcal{M}$
$(M[t\rangle \wedge M[\sigma\rangle M' \wedge M'[t'\rangle) \Rightarrow$
$(t$ and t' are
independent at $M')$.

A set $T_s \subseteq T$ is *persistent at M* iff T_s fulfils PE at M and $\forall t \in T_s \ M[t\rangle$. A set $T_s \subseteq T$ is *conditionally stubborn at M* iff T_s fulfils SD1 and PE at M and $\exists t \in T_s\ M[t\rangle$.

Clearly, the "$M[t\rangle\wedge$" in PE is redundant in the definition of persistence since all transitions in persistent sets are enabled. Looking at PE and proceeding inductively with respect to the length of σ, one observes that "commute" could be substituted for "are independent" in PE. Combining this observation with Lemma 13, one concludes that PE is nothing but SD2. Consequently, we rid ourselves of the concepts of persistence and conditional stubbornness

Lemma 20. *A set fulfils PE at a marking iff the set fulfils SD2 at the marking. A set is conditionally stubborn at a marking iff the set is strongly dynamically stubborn at the marking.*

Proof. See the proof of Lemma 3.10 in [13]. □

The result in Lemma 20 is new.

Lemma 21. *A set is a nonempty persistent set at a marking iff the set is a conditionally stubborn set at the marking and does not contain any transition that is disabled at the marking. The set of enabled transitions of any conditionally stubborn set is a nonempty persistent set.*

Proof. See the proof of Lemma 3.11 in [13]. □

The result in Lemma 21 is due to Godefroid and Pirottin [3] but the proof is new.

Lemma 22. *A set is a nonempty persistent set at a marking iff the set is a strongly dynamically stubborn set at the marking and does not contain any transition that is disabled at the marking. The set of enabled transitions of any strongly dynamically stubborn set is a nonempty persistent set.*

Proof. See the proof of Lemma 3.12 in [13]. □

We now turn to Valmari's dynamically stubborn sets [12]. The prefix AV used in the sequel comes from the name Antti Valmari.

Definition 23. Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let M be a marking of the net. A set $T_s \subseteq T$ *fulfils the principle of AV-strong dynamic stubbornness* (*AVSD* for short) *at M* iff

$$\begin{aligned}&\forall t \in T_s\ \forall t' \in T \setminus T_s\ \forall M' \in \mathcal{M}\\&(M[t\rangle \wedge M'[t\rangle \wedge M'[t'\rangle)\\&\Rightarrow (M[tt'\rangle \wedge M[t't\rangle).\end{aligned}$$

A set $T_s \subseteq T$ is *AV-strongly dynamically stubborn at M* iff T_s fulfils SD1 and AVSD at M and $\exists t \in T_s\ M[t\rangle$.

Our AV-strong dynamic stubbornness is equivalent to Valmari's strong dynamic stubbornness [12], because of the obvious equivalence between our Definition 23 and Valmari's Definition 2.2 in [12]. The principle AVSD is illustrated in Figure 3.

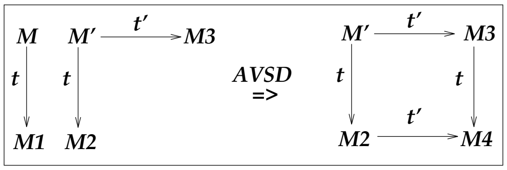

Fig. 3. The principle of AV-strong dynamic stubbornness.

Lemma 24. *If a set is AV-strongly dynamically stubborn set at a marking, the set is strongly dynamically stubborn at the marking.*

Proof. See the proof of Lemma 3.14 in [13]. □

Figure 4 illustrates the classes of dynamically, conventionally dynamically, strongly dynamically, and AV-strongly dynamically stubborn sets at a marking M. The inclusions follow from Lemmas 12, 14, and 24. Depending on M, some or all of the inclusions can be strict, as shown by the examples in [13].

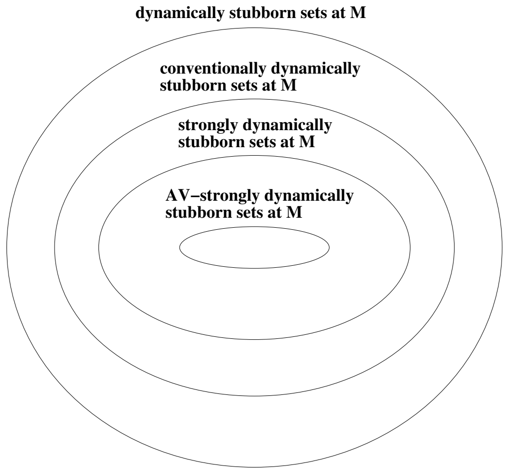

Fig. 4. Four classes of dynamically stubborn sets at a marking.

3.2 Usefulness of Dynamically Stubborn Sets

We now consider what is preserved by a dynamically stubborn set selective reachability graph generation. The results presented in this subsection show the usefulness of dynamically stubborn sets.

Definition 25. Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let f be a function from $\mathcal{M}$ to 2^T. Then we say that f is *dynamically stubborn* iff for each nonterminal marking M, $f(M)$ is dynamically stubborn. Correspondingly, f is *conventionally dynamically stubborn* iff for each nonterminal marking M, $f(M)$ is conventionally dynamically stubborn. Respectively, f is *unconventionally dynamically stubborn* iff f is dynamically stubborn but not conventionally dynamically stubborn. Correspondingly, f is *strongly dynamically stubborn* iff for each nonterminal marking M, $f(M)$ is strongly dynamically stubborn. Finally, f is *AV-strongly dynamically stubborn* iff for each nonterminal marking M, $f(M)$ is AV-strongly dynamically stubborn.

Theorem 26. *Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let f be a dynamically stubborn function from $\mathcal{M}$ to 2^T. Then f represents all sets of enabled permutations to terminal markings.*

Proof. See the proof of Theorem 3.16 in [13]. □

The result in Theorem 26 is due to Valmari [9, 10] but has missed explicit treatment.

Theorem 26 has the consequence that if a finite transition sequence leads from a marking M to a terminal marking, and M occurs in the reduced reachability graph, then an enabled permutation of the sequence occurs in the graph. A dynamically stubborn set selective search thus certainly finds all reachable terminal markings if the net and the set of reachable markings are finite. If the set of reachable markings is infinite but the net is finite and a dynamically stubborn set selective search is performed in a breadth-first order for some time, then reachable terminal markings "near the initial marking" can be found. As we shall see in Section 4, the permutation preserving property makes the stubborn set method compatible with the sleep set method in the detection of reachable terminal markings though a weaker property would suffice.

Theorem 27. *Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let f be a conventionally dynamically stubborn function from $\mathcal{M}$ to 2^T. Then f represents all conditional traces to terminal markings.*

Proof. See the proof of Theorem 3.17 in [13]. □

The result in Theorem 27 is new though inspired by Wolper and Godefroid [14].

Theorem 28. *Let $\langle S, T, F, K, W, M_0 \rangle$ be a place/transition net. Let f be a dynamically stubborn function from $\mathcal{M}$ to 2^T. Then for each marking M and for each infinite transition sequence σ, if σ is enabled at M, there exists an infinite transition sequence σ' such that $M[\sigma'\rangle_f$.*

Proof. See the proof of Theorem 3.18 in [13]. □

The result in Theorem 28 is due to Valmari [10, 12] but has missed explicit treatment. Our proof is like the proof in [12] which is more general than the proof in [10]. Theorem 28 states that for each marking in the reduced reachability graph, if an infinite transition sequence is enabled at the marking, the graph contains an infinite path that starts from the marking. If there is no loop in a finite reduced reachability graph, Theorem 28 implies that the full reachability graph is finite and has no loop either.

We now turn to the *ignoring phenomenon.* A transition is ignored at a marking iff the transition is enabled at the marking but not fired at any marking that is reachable from the marking [10]. The existence of ignored transitions is called the ignoring phenomenon. For example, if there is an isolated transition, we easily get a reduced reachability graph containing only one vertex and one edge, thus leaving the behaviour of the other parts of the net uninvestigated. Valmari has shown [10] that if ignoring does not occur, a transition occurs in the reduced reachability graph iff it occurs in the full reachability graph. Liveness of a transition in the sense defined by Reisig [8] is also preserved if ignoring does not occur [10].

Valmari has developed an algorithm for detecting ignored transitions and an algorithm for eliminating the ignoring phenomenon. The elimination algorithm requires that the chosen dynamically stubborn sets are strongly dynamically stubborn. The detection algorithm does not have that limitation. The detection algorithm works in time at most linear in the number of vertices and edges of the reduced reachability graph. The elimination algorithm works in time at most proportional to the number of vertices and edges of the resulting reduced reachability graph multiplied by the number of transitions of the net [10].

Valmari's algorithms for detecting and eliminating the ignoring phenomenon are based on the property that is stated in Lemma 29.

Lemma 29. *Let $\langle S, T, F, K, W, M_0\rangle$ be a place/transition net. Let f and g be functions from $\mathcal{M}$ to 2^T such that f is dynamically stubborn and*

$$\forall M \in \mathcal{M}\ g(M) \subseteq \{t \in f(M) \mid \forall \delta \in (T \setminus f(M))^*\ M[\delta\rangle \Rightarrow M[\delta t\rangle\}.$$

Then

$$\forall \sigma \in T^*\ \forall t \in T\ \forall M \in \mathcal{M}\ (M[\sigma\rangle_g \wedge M[t\rangle \wedge \forall \delta \in T^*\ \neg M[\delta t\rangle_f) \Rightarrow M[\sigma t\rangle.$$

Proof. See the proof of Lemma 3.19 in [13]. □

The result in Lemma 29 is due to Valmari [10] but has missed explicit treatment. Let's assume that f is a dynamically stubborn function, and there is a path leading from a marking M to a marking M' in the f-reachability graph such that for each edge $\langle M'', t, M'''\rangle$ on the path, t is a key transition of $f(M'')$ at M''. From Lemma 29 it then follows that any transition ignored at M is ignored at M', too.

Let f and g be as in Lemma 29. By Definition 11,

$$\{t \in f(M) \mid \forall \delta \in (T \setminus T_s)^*\ M[\delta\rangle \Rightarrow M[\delta t\rangle\}$$

is then the set of key transitions of $f(M)$ at M and nonempty. We can thus require that $g(M)$ is nonempty when M is nonterminal. Let's further assume that the g-reachability graph is finite. For each transition that is ignored at some marking in the f-reachability graph, by Lemma 29 there is then a terminal maximal strongly connected component of the g-reachability graph such that for each marking in the component, the transition is enabled but not fired at the marking. Valmari's algorithm for detecting ignored transitions inspects the terminal maximal strongly connected components of such g-reachability graph [10].

All enabled transitions of a strongly dynamically stubborn set are key transitions of the set by Lemma 12. If f is a strongly dynamically stubborn function, we can then choose $g(M)$ to be the set of enabled transitions in $f(M)$ with the consequence that the g-reachability graph is the f-reachability graph. Let's further assume that the f-reachability graph is finite. For each transition that is ignored at some marking in the f-reachability graph, there is then a terminal maximal strongly connected component of the f-reachability graph such that for each marking in the component, the transition is enabled but not fired at the marking. Valmari's algorithm for eliminating the ignoring phenomenon utilizes this property [10].

4 Sleep Set Method

In this section we present Godefroid's *sleep set method* [1, 3, 4, 5, 2, 14, 15]. The plain sleep set method preserves at least one sequence from each conditional trace leading from the initial state to a terminal state. To prevent a transition from firing, it is put into a so called sleep set.

Wolper and Godefroid [14], Godefroid and Pirottin [3], and Wolper, Godefroid, and Pirottin [15] have combined the sleep set method with the stubborn set method. The combination is justified by the fact that the stubborn set method alone is sometimes bound to fire independent transitions at a state. The combination presented in [3, 14, 15] is such that at each encountered nonterminal state a nonempty persistent set is computed. Let us recall from Lemma 22 that a set is a nonempty persistent set iff the set is a strongly dynamically stubborn set consisting of enabled transitions only. In [14, 15], persistence is defined on the basis of global independence but since global independence implies independence at each reachable state, the persistent sets in [14, 15] are persistent in the sense defined by Godefroid and Pirottin [3]. As mentioned immediately above Definition 19, our definition of persistence in Definition 19 corresponds to Godefroid's and Pirottin's definition [3]. The plain sleep set method can be thought as a special case of the combined method: a simple heuristic for computing a persistent set is used. We shall not consider the plain sleep set method further.

We concentrate on a generalized version of Wolper's and Godefroid's terminal state detection algorithm [14]. The generalized version is in Figure 5. The intuitive idea of the algorithm is to eliminate such redundant *interleavings of transitions* that are not eliminated by the transition selection function f. We show that the algorithm is guaranteed to find all reachable terminal markings of any finite place/transition net with a finite set of reachable markings. Any dynamically stubborn function is valid for f, but the algorithm is not limited to dynamically stubborn sets. It suffices that f represents all sets of length-secure alternative sequences to terminal markings. The set T_0 can be any subset of transitions that are disabled at the initial marking. The φ in Figure 5 can be any truth-valued function on $\mathcal{M} \times T \times T \times 2^T$ that satisfies: if $\varphi(M, t, t', T_s)$, then either t and t' commute at M and $t' \in T_s$, or tt' is disabled at M. For example, $\varphi(M, t, t', T_s)$ could be

- "either t and t' commute at M and $t' \in T_s$, or tt' is disabled at M",
- "t and t' are independent at M and $t' \in T_s$",
- "t and t' commute at M and $t' \in T_s$", or simply
- "false".

Note that if $M[t\rangle M'$, then tt' is disabled at M iff t' is disabled at M'. So the first alternative in the above list has the effect that if t is fired from M to M' in the algorithm in Figure 5, then the sleep set pushed onto the stack with M' contains all those transitions that are disabled at M'. We shall consider the practicalities related to φ and T_0 later in this section.

```
make Stack empty; make H empty;
push ⟨M_0, T_0⟩ onto Stack;
while Stack is not empty do {
    pop ⟨M, Sleep⟩ from Stack;
    if M is not in H then {
        Fire = {t ∈ f(M) \ Sleep | M[t⟩};
        if Fire and Sleep are both empty then
            print "Terminal state!";
        enter ⟨M, a copy of Sleep⟩ in H;
    }
    else {
        let hSleep be the set associated
            with M in H;
        Fire = {t ∈ hSleep \ Sleep | M[t⟩};
        Sleep = hSleep ∩ Sleep;
        substitute a copy of Sleep for the set
            associated with M in H;
    }
    for each t in Fire do {
        let M[t⟩M';
        tSleep = {t' ∈ T | φ(M, t, t', Sleep)};
        push ⟨M', a copy of tSleep⟩ onto Stack;
        Sleep = {t} ∪ Sleep;
    }
}
```

Fig. 5. A terminal marking detection algorithm.

The algorithm in Figure 5 is similar to Wolper's and Godefroid's algorithm [14]. The only essential differences are that Wolper and Godefroid assume that the set corresponding to $f(M)$ is persistent, the set corresponding to T_0 is empty, and the condition corresponding to $\varphi(M, t, t', T_s)$ is "t and t' are globally independent and $t' \in T_s$".

Theorem 30. *Let $\langle S, T, F, K, W, M_0\rangle$ be a finite place/transition net such that the set of markings reachable from M_0 is finite. Let f be a function from $\mathcal{M}$ to 2^T such that f represents all sets of length-secure alternative sequences to terminal markings. Let T_0 be a subset of transitions that are disabled at M_0. Let φ be a truth-valued function on $\mathcal{M} \times T \times T \times 2^T$ such that for each marking M, for all transitions t and t', and for each $T_s \subseteq T$, if $\varphi(M, t, t', T_s)$, then either t and t' commute at M and $t' \in T_s$, or tt' is disabled at M. Then the algorithm in Figure 5 finds all terminal markings that are reachable from M_0.*

Proof. See the proof of Theorem 4.1 in [13]. □

The result in Theorem 30 is new though inspired by Wolper and Godefroid [14], Godefroid and Pirottin [3], and Wolper, Godefroid, and Pirottin [15]. In [13], an example is given which shows that the statement obtained from Theorem 30 by removing the word "length-secure" is not valid.

Let us recall from Theorem 26 that dynamically stubborn functions represent all sets of enabled permutations to terminal markings. So they represent all sets of length-secure alternative sequences to terminal markings, too. From Theorem 30 it thus follows that the algorithm in Figure 5 is compatible with all dynamically stubborn sets.

Lemma 31. *Let $\langle S, T, F, K, W, M_0\rangle$ be a finite place/transition net. Let φ be a truth-valued function on $\mathcal{M} \times T \times T \times 2^T$ such that for each marking M, for all transitions t and t', and for each $T_s \subseteq T$, if $\varphi(M, t, t', T_s)$, then t and t' are independent at M and $t' \in T_s$. Let $T_0 = \emptyset$. Then in the algorithm in Figure 5, each sleep set associated with a marking contains only transitions that are enabled at the marking.*

Proof. See the proof of Lemma 4.2 in [13]. □

The result in Lemma 31 is due to Wolper and Godefroid [14] despite the differences between the algorithm in Figure 5 and their terminal state detection algorithm.

In Figure 6, an implementation of the algorithm in Figure 5 with respect to φ and T_0 is presented. The algorithm in Figure 6 can be obtained from the algorithm in Figure 5 by making T_0 empty, removing the checking of enabledness from the "else-block", and defining: $\varphi(M, t, t', T_s)$ iff t and t' commute at M

and $t' \in T_s$. We know that transitions commute at a marking iff they are enabled and independent at the marking. Checking commutation should be easier than checking independence. Lemma 31 implies that the algorithm in Figure 5 is equivalent to the algorithm in Figure 6 when T_0 is empty and φ is defined: $\varphi(M, t, t', T_s)$ iff t and t' commute at M and $t' \in T_s$. Lemma 6 thus also implies that in the algorithm in Figure 6, each sleep set associated with a marking contains only transitions that are enabled at the marking.

```
make Stack empty; make H empty;
push ⟨M_0, ∅⟩ onto Stack;
while Stack is not empty do {
    pop ⟨M, Sleep⟩ from Stack;
    if M is not in H then {
        Fire = {t ∈ f(M)\ Sleep | M[t⟩};
        if Fire and Sleep are both empty then
            print "Terminal state!";
        enter ⟨M, a copy of Sleep⟩ in H;
    }
    else {
        let hSleep be the set associated
            with M in H;
        Fire = hSleep \ Sleep;
        Sleep = hSleep ∩ Sleep;
        substitute a copy of Sleep for the set
            associated with M in H;
    }
    for each t in Fire do {
        let M[t⟩M';
        tSleep = {t' ∈ Sleep |
                t and t' commute at M};
        push ⟨M', a copy of tSleep⟩ onto Stack;
        Sleep = {t} ∪ Sleep;
    }
}
```

Fig. 6. A practical implementation of the algorithm in Figure 5 with respect to φ and T_0.

Lemma 32. *Let $\langle S, T, F, K, W, M_0\rangle$ be a finite place/transition net such that the set of markings reachable from M_0 is finite. Let T_0 be a subset of transitions that are disabled at M_0. Let φ be a truth-valued function on $\mathcal{M} \times T \times T \times 2^T$ such that for each marking M, for all transitions t and t', and for each $T_s \subseteq T$, if $\varphi(M, t, t', T_s)$, then either t and t' commute at M and $t' \in T_s$, or tt' is disabled at M. Let's further require that for each marking M, for all transitions t and t', and for each $T_s \subseteq T$, if t and t' commute at M and $t' \in T_s$, then $\varphi(M, t, t', T_s)$. Let's assume that the sets, the set operations (insertion, union, intersection, and difference), the stack, the stack operations, the "for-loop", and the computation of $f(M)$ in the algorithms in Figure 5 and 6 are implemented exactly in the same way. Then the algorithms visit exactly the same markings and fire exactly the same transitions in exactly the same order.*

Proof. See the proof of Lemma 4.3 in [13]. □

The result in Lemma 32 is new though inspired by Wolper and Godefroid [14]. Lemma 32 states that there is no more refined implementation of the algorithm in Figure 5 with respect to φ and T_0 than the algorithm in Figure 6 if φ and T_0 are required to satisfy the assumptions in Theorem 30. Lemmas 31 and 32 suggest the heuristic that each sleep set associated with a marking should only contain transitions that are enabled at the marking.

Let's consider the complexity of the algorithm in Figure 6. The time taken by a check of whether two transitions commute at a marking is at most proportional to ν, where ν is the maximum number of adjacent places of a transition. The cumulative time per marking spent in the "for-loop" is at most proportional to $\nu\rho^2$, where ρ is the maximum number of enabled transitions of a marking, and all visits to the marking are counted. This is based on the fact that each sleep set associated with a marking contains only transitions that are enabled at the marking. The time per visit to a marking spent in the operations related to H is the time of the search for the marking plus a time that is at most proportional to ρ. The searches in H are something that cannot be avoided easily whether or not we use sleep sets at all. It depends much on the net how many times a marking is visited and how many simultaneous occurrences of a marking there are in the stack. One stack element requires space for the marking and at most ρ transitions. It is not necessary to store copies of markings and transitions since pointers suffice. More clever ways to cut down on space consumption in sleep set algorithms have been presented by Godefroid, Holzmann, and Pirottin [2].

The combination of the sleep set method and the stubborn set method can really be better than the plain stubborn set method as far as the number of inspected markings is concerned. More precisely, there can be a dynamically stubborn function f such that $f(M)$ can be computed by using a feasible algorithm such as the *incremental algorithm* [9], and for each dynamically stubborn function g, the number of vertices in the g-reachability graph is greater than the number of markings that are inspected by the algorithm in Figure 6 that uses f. In [13], a simple example showing this is given. The example is essentially the same as can be found in [15]. The statistics in [3, 14, 15] concerning some analyzed protocols do not give direct information for comparing the stubborn set method with the combination of the sleep set method and the stubborn set method.

5 Conclusions

We have studied Valmari's stubborn set method [9, 10, 11, 12] and Godefroid's sleep set method [1, 3, 4, 5, 2, 14, 15] and their combination [3, 14, 15] in place/transition nets [8].

We have shown dynamically stubborn sets [7, 12] to be a useful generalization of stubborn sets. Dynamically stubborn sets seem to have all the nice properties of stubborn sets except that the definition of dynamic stubbornness does not seem to imply a practical algorithm for computing dynamically stubborn sets.

The stubborn set method alone is sometimes bound to fire independent transitions at a state, so the sleep set method can further be used to eliminate redundant interleavings of transitions. We have generalized Wolper's and Godefroid's terminal state detection algorithm [14] and shown that the generalized version detects all reachable terminal markings of any finite place/transition net with a finite full reachability graph, given that the transition selection function represents all sets of length-secure alternative sequences to terminal markings. As already known, dynamically stubborn functions represent all sets of enabled permutations to terminal markings. They thus also represent all sets of length-secure alternative sequences to terminal markings.

Wolper and Godefroid [14], and Wolper, Godefroid, and Pirottin [15] suggest that the stubborn set method and the sleep set are compatible in a broad area of verification. The compatibility should certainly be studied further since all available means should be utilized in attacking the state space explosion problem, and in our opinion, only the detection of reachable terminal states has obtained more than cursory treatment so far. Linear temporal logics seem to form the most central area of research since they have a great expressive power, and the stubborn set method alone as well as the sleep set method alone can be extended to verify properties expressed as linear temporal logic formulae without a next state -operator [4, 11]. The combination of the stubborn set method and the sleep set method should be studied in all those models of concurrency where each of these two methods alone are applicable. Finally, we have the problem of how efficient the algorithms are in practice and what could be done to improve their efficiency.

Acknowledgements

This work has been carried out in the Digital Systems Laboratory of Helsinki University of Technology. This paper is an abridged version of a part of the research report [13]. I am grateful to Professor Leo Ojala for his continuous support and Assistant Professor Mikko Tiusanen for his helpful comments. In addition, I would like to thank Assistant Professor Antti Valmari and Lic.Tech. Marko Rauhamaa for fruitful discussions on the subject of this research.

The financial support received from the Emil Aaltonen Foundation is also gratefully acknowledged.

References

1. Godefroid, P.: *Using Partial Orders to Improve Automatic Verification Methods.* Clarke, E.M. and Kurshan, R.P. (Eds.): Proceedings of the 2nd International Workshop on Computer-Aided Verification, New Brunswick NJ, June 1990. Lecture Notes in Computer Science 531, Springer-Verlag, Berlin 1991, pp. 176–185.
2. Godefroid, P., Holzmann, G.J., and Pirottin, D.: *State Space Caching Revisited.* von Bochmann, G. and Probst, D.K. (Eds.): Proceedings of the 4th International Workshop on Computer-Aided Verification, Montreal, June 1992. Lecture Notes in Computer Science 663, Springer-Verlag, Berlin 1993.
3. Godefroid, P. and Pirottin, D.: *Refining Dependencies Improves Partial-Order Verification Methods.* Courcoubetis, C. (Ed.): Proceedings of the 5th International Conference on Computer-Aided Verification, Elounda, Greece, June/July 1993. Lecture Notes in Computer Science 697, Springer-Verlag, Berlin 1993, pp. 438–449.
4. Godefroid, P. and Wolper, P.: *A Partial Approach to Model Checking.* Proceedings of the 6th Annual IEEE Symposium on Logic in Computer Science, Amsterdam, July 1991. IEEE Computer Society Press, Los Alamitos CA 1991, pp. 406–415.
5. Godefroid, P. and Wolper, P.: *Using Partial Orders for the Efficient Verification of Deadlock Freedom and Safety Properties.* Formal Methods in System Design 2 (1993) 2, pp. 149–164.
6. Katz, S. and Peled, D.: *Defining Conditional Independence Using Collapses.* Theoretical Computer Science 101 (1992) 2, pp. 337–359.
7. 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.
8. Reisig, W.: *Petri Nets: An Introduction.* EATCS Monographs on Theoretical Computer Science 4, Springer-Verlag, Berlin 1985, 161 p.
9. Valmari, A.: *Error Detection by Reduced Reachability graph generation.* Proceedings of the 9th European Workshop on Application and Theory of Petri Nets, Venice, June 1988, pp. 95–112.
10. Valmari, A.: *Stubborn Sets for Reduced State Space Generation.* Rozenberg, G. (Ed.), Advances in Petri Nets 1990. Lecture Notes in Computer Science 483, Springer-Verlag, Berlin 1991, pp. 491–515.
11. Valmari, A.: *A Stubborn Attack on State Explosion.* Formal Methods in System Design 1 (1992) 4, pp. 297–322.
12. Valmari, A.: *Stubborn Sets of Coloured Petri Nets.* Proceedings of the 12th International Conference on Application and Theory of Petri Nets, Gjern, Denmark, June 1991, pp. 102–121.
13. Varpaaniemi, K.: *Efficient Detection of Deadlocks in Petri Nets.* Helsinki University of Technology, Digital Systems Laboratory Report A 26, Espoo, October 1993, 56 p.
14. Wolper, P. and Godefroid, P.: *Partial-Order Methods for Temporal Verification.* Best, E. (Ed.): Proceedings of the 4th International Conference on Concurrency Theory, Hildesheim, August 1993. Lecture Notes in Computer Science 715, Springer-Verlag, Berlin 1993, pp. 233–246.
15. Wolper, P., Godefroid, P., and Pirottin.: *A Tutorial on Partial-Order Methods for the Verification of Concurrent Systems.* Tutorial material of the 5th International Conference on Computer-Aided Verification, Elounda, Greece, June/July 1993, 85 p.