Addendum to Dynamic Partial-Order Reduction for Model Checking Software

Cormac Flanagan
University of California at Santa Cruz
cormac@cs.ucsc.edu

Patrice Godefroid
Microsoft Research
pg@microsoft.com

On page 6 of our POPL'2005 paper, we wrote that "sleep sets can be added exactly as described in [10]". Specifically, sleep sets can be added to the algorithm of Figure 3 as follows:

- line 5 should be replaced with

  let $E = \{q \in \mathit{enabled}(\mathit{pre}(S,i)) \mid q = p$ or $\exists j \in \mathit{dom}(S) : j > i$ and $q = \mathit{proc}(S_j)$ and $j \rightarrow_S p\} \setminus \mathit{Sleep}(\mathit{pre}(S,i))$;

- line 7 should be replaced with

  else add all $q \in (\mathit{enabled}(\mathit{pre}(S,i)) \setminus \mathit{Sleep}(\mathit{pre}(S,i)))$ to $\mathit{backtrack}(\mathit{pre}(S,i))$;

The rules for defining and manipulating sleep sets are the same as in [10].

The correctness of this combination can be proved as follows. The definition of $E(S,i,p)$ (see the appendix) becomes:

$$\{\ q \in \mathit{enabled}(\mathit{pre}(S,i)) \mid q = p \text{ or } \exists j \in \mathit{dom}(S) : j > i \text{ and } q = \mathit{proc}(S_j) \text{ and } j \rightarrow_S p\} \setminus \mathit{Sleep}(\mathit{pre}(S,i))$$

The definition of $PC(S,j,p)$ then becomes:

if
  S is a transition sequence from s_0 in A_G
  and $i = \mathit{max}(\{i \in \mathit{dom}(S) \mid S_i$ is dependent and co-enabled with $\mathit{next}(\mathit{last}(S),p)$ and $i \not\rightarrow_S p\})$
  and $i \le j$
then
  if $E(S,i,p) \neq \emptyset$
  then $\mathit{backtrack}(\mathit{pre}(S,i)) \cap E(S,i,p) \neq \emptyset$
  else $\mathit{backtrack}(\mathit{pre}(S,i)) = \mathit{enabled}(\mathit{pre}(S,i)) \setminus \mathit{Sleep}(\mathit{pre}(S,i))$

The postcondition PC for Explore(S) becomes:

$$\forall p\ \forall w : (\forall w_i \in [w] : w_i^1 \notin \mathit{Sleep}(\mathit{last}(S))) \Rightarrow PC(S.w, |S|, p)$$

where $\forall w_i \in [w]$ denotes the set of sequences w_i of transitions equivalent to w (i.e., transition sequences that are part of the same Mazurkiewicz's trace – see [10] for details), and where w_i^1 denotes the first transition of w_i.

In the presence of sleep sets, we use the following definition (similar notions are used in [9], for instance in Theorem 5.2):

Definition 1. *A set $T \subseteq \mathcal{T}$ of transitions enabled in a state s is* partially persistent *in s iff, for all nonempty sequences w of transitions*

$$s_1 \xrightarrow{t_1} s_2 \xrightarrow{t_2} s_3 \ldots \xrightarrow{t_{n-1}} s_n \xrightarrow{t_n} s_{n+1}$$

from s in A_G and including only transitions $t_i \notin T$, $1 \le i \le n$, and such that $\forall w_i \in [w] : w_i^1 \notin \mathit{Sleep}(s)$, t_n is independent with all the transitions in T.

If $\mathit{Sleep}(s) = \emptyset$, this definition coincides with the definition of persistent sets. Note that if $T = \mathit{enabled}(s) \setminus \mathit{Sleep}(s)$, T is a partially persistent set in s.

With sleep sets, Lemma 1 and Theorem 1 in the appendix remains the same except that "is a persistent set in s" has to be replaced by "is a partially persistent set in s" in both. From this modified Theorem 1, it follows from the proof of Theorem 2 in [10] that all deadlocks (terminating states) are visited by the combined algorithm using sleep sets.

For clarity and completeness, we include below those modified versions of Lemma 1 and Theorem 1 extended with sleep sets, as well as their proof.

Lemma 1. *Whenever a state s reached after a transition sequence S is backtracked during the search performed by the algorithm of Figure 3, the set T of transitions that have been explored from s is a partially persistent set in s, provided the postcondition PC holds for every recursive call Explore($S.t$) for all $t \in T$.*

Proof. Let

$$\begin{array}{rcl} s & = & \mathit{last}(S) \\ T & = & \{\mathit{next}(s,p) \mid p \in \mathit{backtrack}(s)\} \end{array}$$

If T is not $\mathit{enabled}(s) \setminus \mathit{Sleep}(s)$, T is non-empty and we prove that T is a partially persistent set in s by contradiction: assume that there exist $t_1, \ldots, t_n \notin T$ such that

1. $S.t_1 \ldots t_n$ is a transition sequence from s_0 in A_G and
2. $\forall w_i \in [t_1 \ldots t_n] : w_i^1 \notin \mathit{Sleep}(s)$ and
3. $t_1, \ldots, t_{n-1}$ are all independent with T and
4. t_n is dependent with some $t \in T$.

Let $w = t_1 \ldots t_{n-1}$. By property of independence, this implies that t is enabled in the state $last(S.w)$ and hence co-enabled with t_n. Without loss of generality, assume that $t_1 \ldots t_n$ is the *shortest* such sequence. We thus have that

$$\forall 1 \leq i < n : i \rightarrow_{S.w} proc(t_n)$$

(If this was not true for some i, the same transition sequence without i would also satisfy our assumptions and be shorter.)

By definition, $S.w$ is itself a transition sequence from s_0 in A_G and we have

$$next(last(S.w), proc(t_n)) = t_n$$

If $proc(t) = proc(t_n)$ then

$$\begin{aligned} t &= next(last(S), proc(t)) \\ &= next(last(S.w), proc(t)) \\ &= t_n \end{aligned}$$

since t is independent with all the transitions in w, contradicting that $t_n \notin T$. Hence $proc(t) \neq proc(t_n)$.

Since t is in a different process than t_n and since t is independent with all the transitions in w, we have

$$\begin{aligned} t_n &= next(last(S.w), proc(t_n)) \\ &= next(last(S.w.t), proc(t_n)) \\ &= next(last(S.t.w), proc(t_n)) \end{aligned}$$

Since $t \in T$, t is executed from s. Since $\forall w_i \in [w] : w_i^1 \notin Sleep(s)$ and since $t_1, \ldots, t_n \notin T$ (i.e., none of those transitions are executed from s), none of the w_i^1 transitions are in $Sleep(last(S.t))$ (by construction – see the rules for defining sleep sets in [10]).

Let $i = |S| + 1$. Consider the postcondition

$$PC(S.t.w, i, proc(t_n))$$

for the recursive call Explore($S.t$). Clearly,

$$i \not\rightarrow_{S.t.w} proc(t_n)$$

(since t is in a different process than t_n and since t is independent with $t_1, \ldots, t_{n-1}$). In addition, we have (by definition of E):

$$\begin{aligned} &E(S.t.w, i, proc(t_n)) \subseteq \\ &\quad \{proc(t_1), \ldots, proc(t_{n-1}), proc(t_n)\} \cap enabled(s) \end{aligned}$$

Moreover, we have by construction:

$$\forall j \in dom(S.t.w) : j > i \Rightarrow j \rightarrow_{S.t.w} proc(t_n)$$

Hence, by the postcondition PC for the recursive call Explore($S.t$), either $E(S.t.w, i, proc(t_n))$ is nonempty and at least one process in $E(S.t.w, i, proc(t_n))$ is in $backtrack(s)$, or $E(S.t.w, i, proc(t_n))$ is empty and all the processes in $enabled(s) \setminus Sleep(s)$ are in $backtrack(s)$. In either case, at least one transition among $\{t_1, \ldots, t_n\}$ is in T. This contradicts the assumption that $t_1, \ldots, t_n \notin T$.

□

THEOREM 1. *Whenever a state s reached after a transition sequence S is backtracked during the search performed by the algorithm of Figure 3 in an acyclic state space, the postcondition PC for Explore(S) is satisfied, and the set T of transitions that have been explored from s is a partially persistent set in s.*

PROOF. Let

$$\begin{aligned} s &= last(S) \\ T &= \{next(s, p) \mid p \in backtrack(s)\} \end{aligned}$$

The proof is by induction on the order in which states are backtracked.

(Base case) Since the state space A_G is acyclic and since the search is performed in depth-first order, the first backtracked state must be either a deadlock where no transition is enabled, or a state s where $enabled(s) = Sleep(s)$ (i.e., all transitions enabled in s are in $Sleep(s)$). Therefore, in either case, the postcondition for that state becomes $\forall p :\ PC(S, |S|, p)$, and is directly established by lines 3–9 of the algorithm of Figure 3.

(Inductive case) We assume that each recursive call to Explore($S.t$) satisfies its postcondition. That T is a partially persistent set in s then follows by Lemma 1. We show that Explore(S) ensures its postcondition PC for any p and w such that $S.w$ is a transition sequence from s_0 in A_G and such that $\forall w_i \in [w] : w_i^1 \notin Sleep(last(S))$.

1. Suppose some transition in w is dependent with some transition in T. In this case, we split w into $X.t.Y$, where all the transitions in X are independent with all the transitions in T and t is the first transition in w that is dependent with some transition in T. Since T is a partially persistent set in s, t must be in T (otherwise, T would not be partially persistent in s). Thus, t is independent with all the transitions in X. By property of independence, this implies that the transition sequence $t.X.Y$ is executable from s. It also implies that t is one of the w_i^1 transitions.

   (Case 1.a) If t is the first transition of the w_i^1 transitions of w to be executed in s and since none of those are in $Sleep(last(S))$, then $Sleep(last(S.t))$ does not contain any of the w_i^1 transitions either (by the rules defining sleep sets in [10]). By applying the inductive hypothesis to the recursive call Explore($S.t$) for the sequence $X.Y$, we know

   $$\forall p :\ PC(S.t.X.Y, |S| + 1, p)$$

   which implies (by the definition of PC) that

   $$\forall p :\ PC(S.t.X.Y, |S|, p)$$

   Since t is independent with all the transitions in X, we also have that

   $$\forall i \in dom(S.t.X.Y) : i \rightarrow_{S.t.X.Y} p \text{ iff } \quad i \rightarrow_{S.X.t.Y} p$$

   Therefore, by definition,

   $$PC(S.t.X.Y, |S|, p) \text{ iff } PC(S.X.t.Y, |S|, p)$$

   We can thus conclude that

   $$\forall p :\ PC(S.X.t.Y, |S|, p)$$

   (Case 1.b) Otherwise, let t' be the first transition of the w_i^1 transitions of w which is executed in s before t. We thus have $w = X.t.W.t'.Z$. Since t' is one of the w_i^1 transitions, we know (by definition of w_i^1) that t' is independent of all transitions in $X.t.W$.

The same reasoning as in the previous case 1.a can be applied to Explore($S.t'$) and the sequence $X.t.W.Z$. We can thus prove that

$$PC(S.t'.X.t.W.Z, |S|, p) \text{ iff } PC(S.X.t.W.t'.Z, |S|, p)$$

and conclude again that

$$\forall p: \; PC(S.w, |S|, p)$$

2. Suppose that all the transitions in w are independent with all the transitions in T and $p \in \mathit{backtrack}(s)$. Then

   (a) $\mathit{next}(s, p) \in T$;

   (b) $\mathit{next}(s, p)$ is independent with w;

   (c) p is a different process from any transition in w;

   (d) $\mathit{next}(\mathit{last}(S.w), p) = \mathit{next}(\mathit{last}(S), p)$;

   (e) $\forall i \in \mathit{dom}(S): \; i \rightarrow_{S.w} p$ iff $i \rightarrow_S p$.

   Thus, we have $PC(S.w, |S|, p)$ iff $PC(S, |S|, p)$, and the latter is directly established by the lines 3–9 of the algorithm for all p.

3. Suppose that all the transitions in w are independent with all the transitions in T and $p \notin \mathit{backtrack}(s)$. Pick any $t \in T$. We then have that

   (a) $\mathit{proc}(t) \neq p$;

   (b) t independent with all the transitions in w;

   (c) $\mathit{next}(\mathit{last}(S.w), p) = \mathit{next}(\mathit{last}(S.t.w), p)$;

   (d) $\forall i \in \mathit{dom}(S): \; i \rightarrow_{S.w} p$ iff $i \rightarrow_{S.t.w} p$.

   Thus, we have $PC(S.w, |S|, p)$ iff $PC(S.t.w, |S|, p)$.

   Since none of the w_i^1 transitions are in $\mathit{Sleep}(\mathit{last}(S))$ and since none of those transitions are executed in s, $\mathit{Sleep}(\mathit{last}(S.t))$ does not contain any of the w_i^1 transitions either (by the rules defining sleep sets in [10]).

   By applying the inductive hypothesis to the recursive call Explore($S.t$), we know

   $$\forall p: \; PC(S.t.w, |S| + 1, p)$$

   which implies (by the definition of PC) that

   $$\forall p: \; PC(S.t.w, |S|, p)$$

   which in turn implies

   $$\forall p: \; PC(S.w, |S|, p)$$

   as required.

□

Acknowledgements: We thank Katie Coons for pointing out that combining sleep sets with our POPL'2005 algorithm is not as immediate as originally thought and for helpful comments.