On-The-Fly Partial Order Reduction

Lecture #11 of Advanced Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

E-mail: `katoen@cs.rwth-aachen.de`

May, 2012

Outline of partial-order reduction

- During state space generation obtain $\widehat{TS}$
  - a *reduced version* of transition system *TS* such that $\widehat{TS} \triangleq TS$
  - $\Rightarrow$ this preserves all stutter sensitive LT properties, such as $\text{LTL}_{\setminus\bigcirc}$
  - at state s select a (small) subset of enabled actions in s
  - different approaches on how to select such set: consider Peled's *ample sets*

- *Static* partial-order reduction
  - obtain a high-level description of $\widehat{TS}$ (without generating *TS*)
  - $\Rightarrow$ POR is preprocessing phase of model checking

- *Dynamic (or: on-the-fly)* partial-order reduction
  - construct $\widehat{TS}$ during $\text{LTL}_{\setminus\bigcirc}$ model checking
  - if accept cycle is found, there is no need to generate entire $\widehat{TS}$

Ample-set conditions for LTL

(A1) **Nonemptiness condition**

$\varnothing \neq \textit{ample}(s) \subseteq \textit{Act}(s)$

(A2) **Dependency condition**

Let $s \xrightarrow{\beta_1} \ldots \xrightarrow{\beta_n} s_n \xrightarrow{\alpha} t$ be a finite execution fragment in *TS* such that α depends on $\textit{ample}(s)$. Then: $\beta_i \in \textit{ample}(s)$ for some $0 < i \leqslant n$.

(A3) **Stutter condition**

If $\textit{ample}(s) \neq \textit{Act}(s)$ then any $\alpha \in \textit{ample}(s)$ is a stutter action.

(A4) **Cycle condition**

For any cycle $s_0\, s_1\, \ldots\, s_n$ in $\widehat{\textit{TS}}$ and $\alpha \in \textit{Act}(s_i)$, for any $0 < i \leqslant n$, there exists $j \in \{ 1, \ldots, n \}$ such that $\alpha \in \textit{ample}(s_j)$.

Correctness theorem

For action-deterministic, finite *TS* without terminal states:

if conditions (A1) through (A4) are satisfied, then $\widehat{TS} \triangleq TS$.

Strong cycle condition

(A4') Strong cycle condition

On any cycle $s_0 \, s_1 \, \ldots \, s_n$ in $\widehat{\mathit{TS}}$,

there exists $j \in \{ 1, \ldots, n \}$ such that $\mathit{ample}(s_j) = \mathit{Act}(s_j)$.

- If (A1) through (A3) hold: (A4') implies the cycle condition (A4)
- (A4') can be checked easily in DFS when backward edge is found

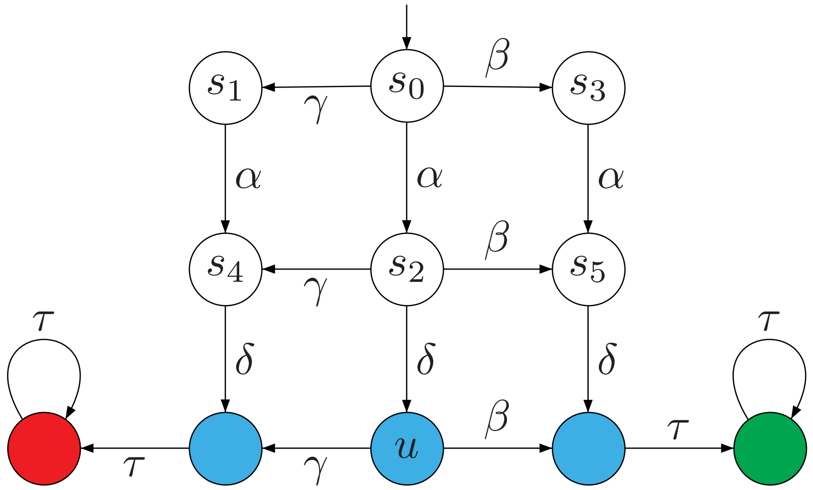
The branching-time ample approach

- Linear-time ample approach:
  - during state space generation obtain $\widehat{TS}$ such that $\widehat{TS} \triangleq TS$
  - $\Rightarrow$ this preserves all stutter sensitive LT properties, such as $\text{LTL}_{\setminus \bigcirc}$
  - static partial order reduction: generate $\widehat{TS}$ prior to verification
  - on-the-fly partial order reduction: generate $\widehat{TS}$ during the verification
  - generation of $\widehat{TS}$ by means of static analysis of program graphs

- Branching-time ample approach
  - during state space generation obtain $\widehat{TS}$ such that $\widehat{TS} \approx^{div} TS$
  - $\Rightarrow$ this preserves all $\text{CTL}_{\setminus \bigcirc}$ and $\text{CTL}^{*}_{\setminus \bigcirc}$ formulas
  - static partial order reduction only

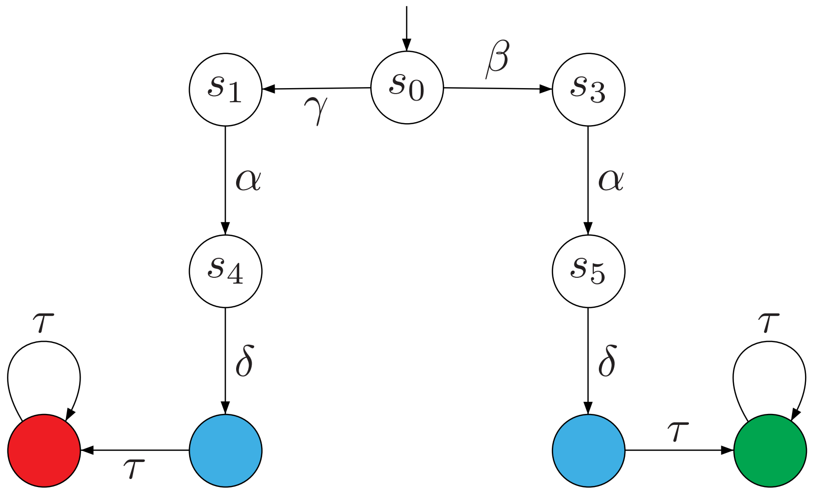
as $\approx^{div}$ is strictly finer than $\triangleq$, try (A1) through (A4)

Example



transition system *TS*

Conditions (A1)-(A4) are insufficient



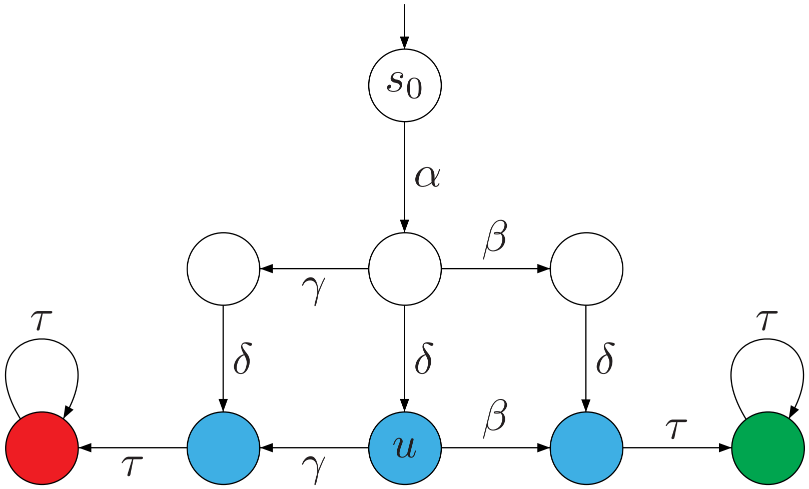
$\widehat{TS} \models \forall\Box\Big(a \rightarrow (\forall\Diamond b \vee \forall\Diamond c)\Big)$ but TS does not and thus $\widehat{TS} \not\approx^{div} TS$

Branching condition

(A5)

If $\mathit{ample}(s) \neq \mathit{Act}(s)$ then $|\mathit{ample}(s)| \; = \; 1$

A sound reduction for $\text{CTL}^*_{\setminus\bigcirc}$



$\widehat{TS} \not\models \forall\Box\Big(a \rightarrow (\forall\Diamond b \vee \forall\Diamond c)\Big)$ and *TS* does not ;in fact $\widehat{TS} \approx^{div} TS$

Correctness theorem

For action-deterministic, finite *TS* without terminal states:

if conditions (A1) through (A5) are satisfied, then $\widehat{TS} \approx^{div} TS$.

recall that this implies that $\widehat{TS}$ and *TS* are $\text{CTL}^*_{\setminus \bigcirc}$-equivalent

Ample-set conditions for CTL*

(A1) **Nonemptiness condition**

$\varnothing \neq \mathit{ample}(s) \subseteq \mathit{Act}(s)$

(A2) **Dependency condition**

Let $s \xrightarrow{\beta_1} \ldots \xrightarrow{\beta_n} s_n \xrightarrow{\alpha} t$ be a finite execution fragment in *TS* such that α depends on *ample*(s). Then: $\beta_i \in \mathit{ample}(s)$ for some $0 < i \leqslant n$.

(A3) **Stutter condition**

If *ample*$(s) \neq$ *Act*(s) then any $\alpha \in \mathit{ample}(s)$ is a stutter action.

(A4) **Cycle condition**

For any cycle $s_0\, s_1\, \ldots\, s_n$ in $\widehat{TS}$ and $\alpha \in \mathit{Act}(s_i)$, for some $0 < i \leqslant n$, there exists $j \in \{ 1, \ldots, n \}$ such that $\alpha \in \mathit{ample}(s_j)$.

(A5) **Branching condition**

If *ample*$(s) \neq$ *Act*(s) then $|\mathit{ample}(s)| \; = \; 1$