Heuristics for Checking Liveness Properties with Partial Order Reductions

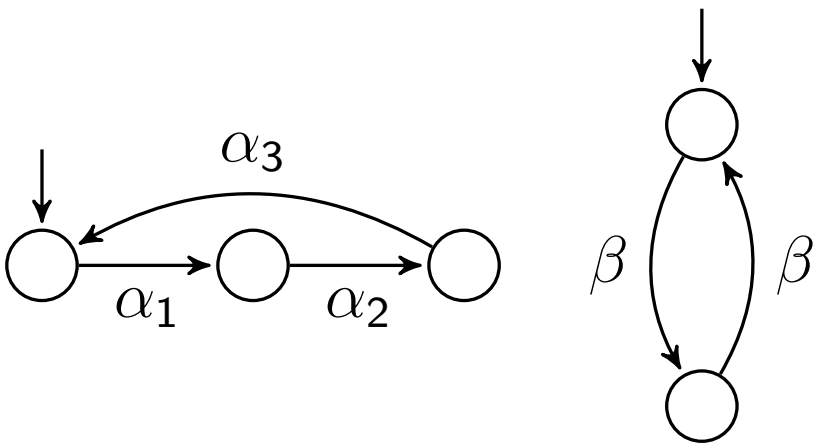
A. Duret-Lutz, F. Kordon, D. Poitrenaud, E. Renault

Tuesday, October 18th

State Space Explosion

- Two concurrent processes
- β independent of α_1, α_2, and α_3

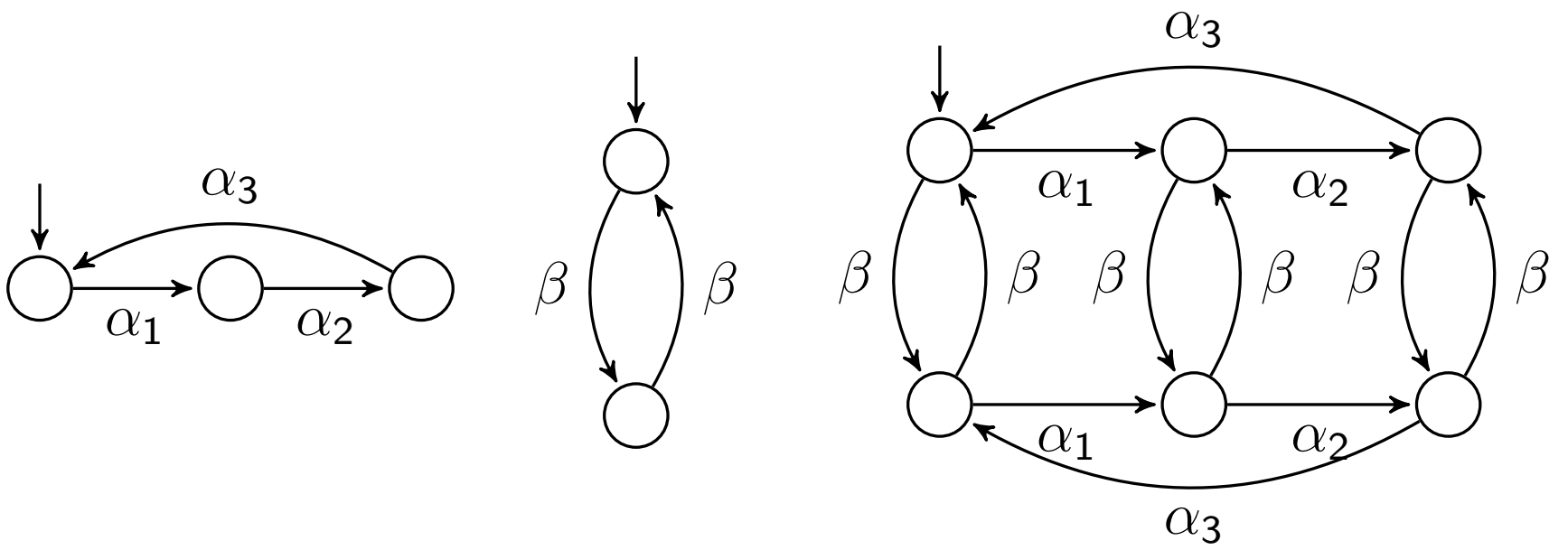
Process 1	Process 2



State Space Explosion

- Two concurrent processes
- β independent of α_1, α_2, and α_3

Process 1 | Process 2 | State Space

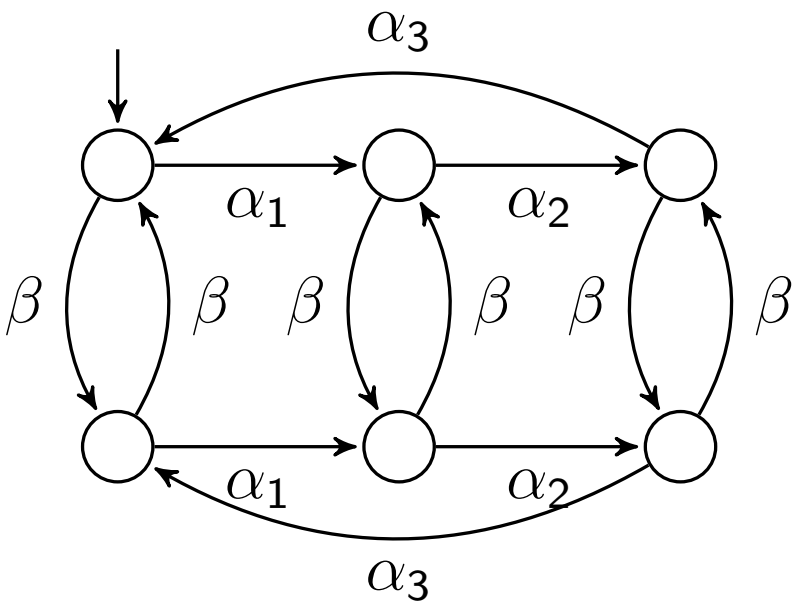


Process interleavings are one of the main sources of state-space explosion for explicit model checkers

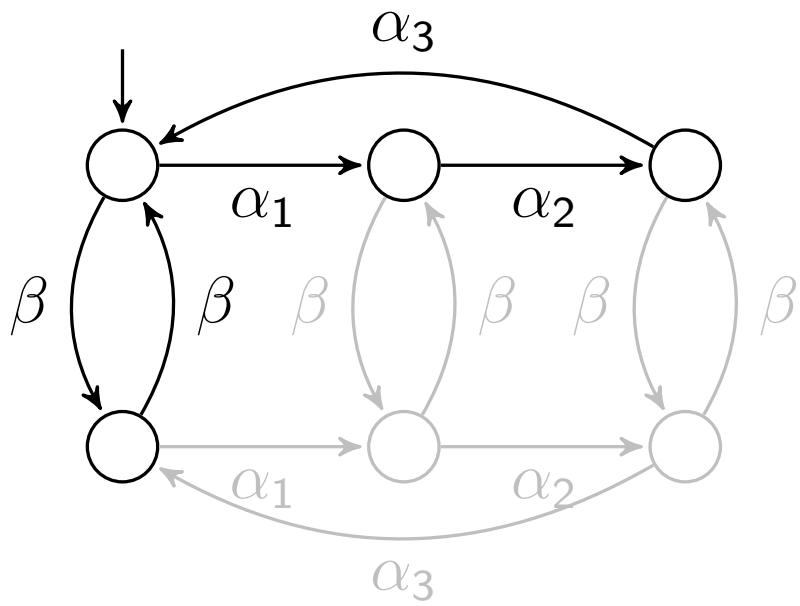
Partial Order Reductions (POR)

- Build a reduced state space
- For each state only consider a **reduced** subset of actions

State Space



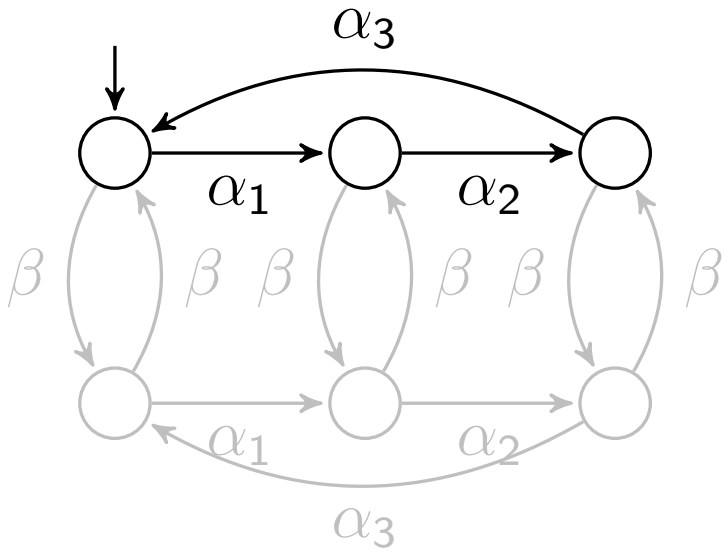
Possible Reduced State Space



POR work only iff the property to check belongs to LTL\X

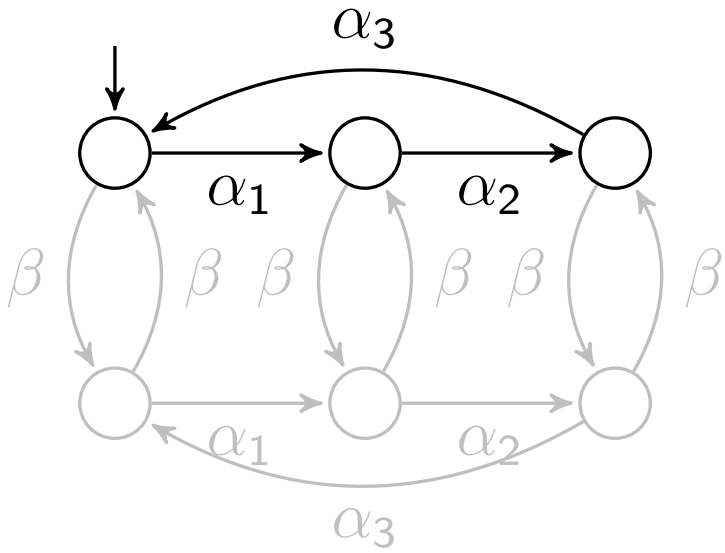
The Ignoring Problem for Liveness Properties

- If the same actions are consistently ignored along a cycle, they may never be executed (below β is never executed)



The Ignoring Problem for Liveness Properties

- If the same actions are consistently ignored along a cycle, they may never be executed (below β is never executed)



Requires an extra condition: the proviso

A proviso[a] ensures that every cycle in the reduced graph contains at least one **expanded state**, i.e, a state where all actions are considered.

[a] More simpler provisos can be applied for safety properties Evangelista and Pajault [2010]

Model Checking LTL\X with POR

Use classical DFS-based emptiness checks

During DFS:

- how to detect cycles without expanded states?
- which state to expand in a cycle?

Objectives:

- Choose states to expand states in order to have the smallest reduced state space

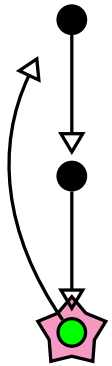
Variations on SPIN's proviso

Source [Peled, 1994]

Expanded state  Not expanded state ● Already visited edge →▷

Variations on SPIN's proviso

Source [Peled, 1994]

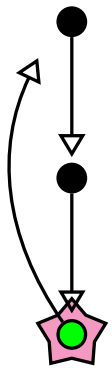


Systematically expands the source of a backedge

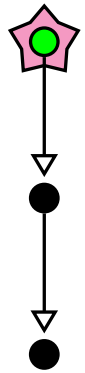
Expanded state  Not expanded state ● Already visited edge →▷

Variations on SPIN's proviso

Source [Peled, 1994]



CondSource

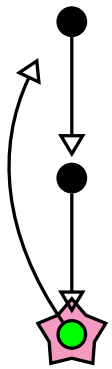


Systematically expands the source of a backedge

Expanded state  Not expanded state ● Already visited edge →▷

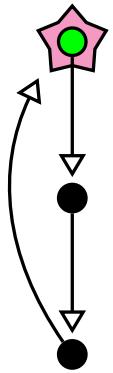
Variations on SPIN's proviso

Source [Peled, 1994]



Systematically expands the source of a backedge

CondSource



Expands the source of backedge iff destination is not expanded

Expanded state ⬟ Not expanded state ● Already visited edge →

Evaluation

- 38 models from the BEEM benchmark
- *reduced* implements the stubborn-set method from Valmari
- Each model is run 100 times with different transition order

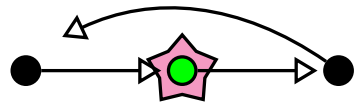
	states (10^6)		transitions (10^6)		st/ms
Full	784.45	100.00%	2,677.73	100.00%	17.90
SOURCE [Peled, 1994]	303.21	38.65%	679.16	25.36%	12.33
CONDSOURCE	252.83	32.23%	518.80	19.37%	11.85
None	57.58	7.34%	97.65	3.65%	22.65

Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource

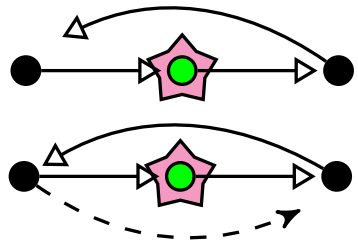
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:



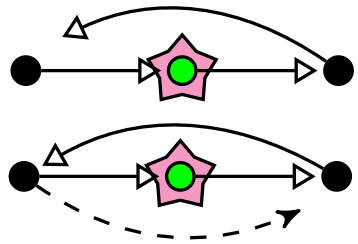
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



Deconstructing Evangelista and Pajault [2010] proviso

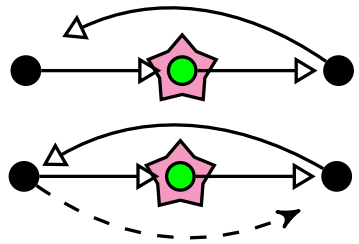
- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:

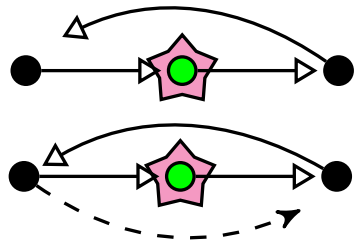


- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known

Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

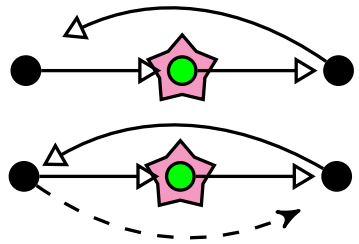
Weighted	Scan	Known

weight: 0

Keep track of exp-
-anded states on DFS

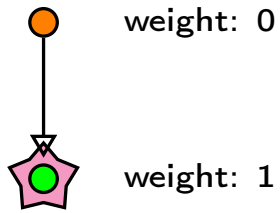
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

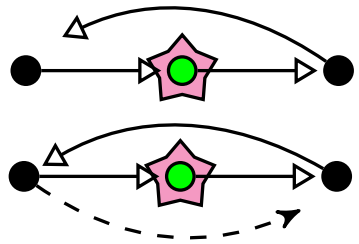
Weighted	Scan	Known



Keep track of exp-
-anded states on DFS

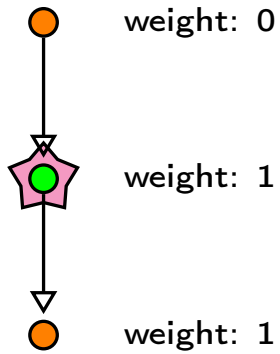
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

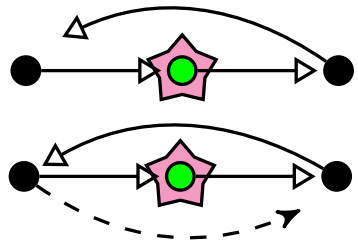
Weighted	Scan	Known



Keep track of exp-
-anded states on DFS

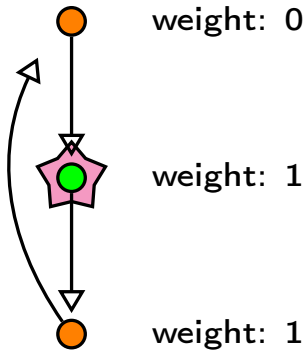
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

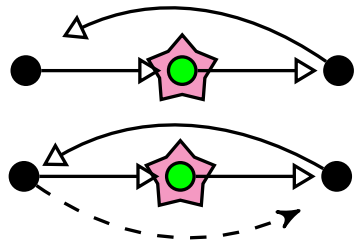
Weighted	Scan	Known



Keep track of exp-
-anded states on DFS

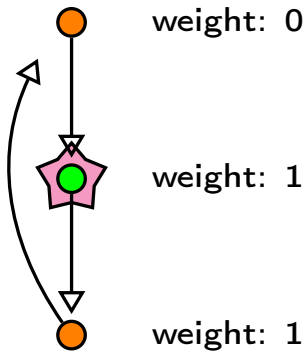
Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known

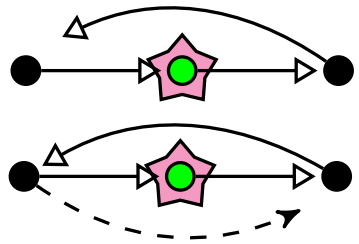


Keep track of exp--anded states on DFS

Early tag "safe" states

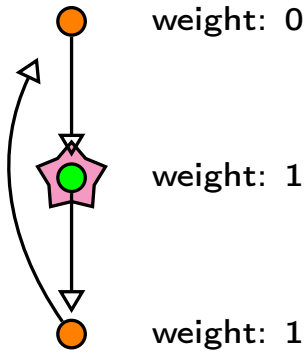
Deconstructing Evangelista and Pajault [2010] proviso

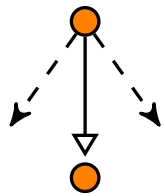
- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known



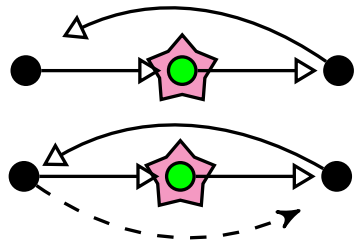


Keep track of exp--anded states on DFS

Early tag "safe" states

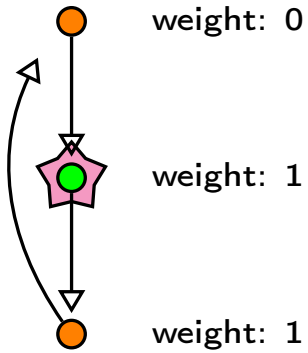
Deconstructing Evangelista and Pajault [2010] proviso

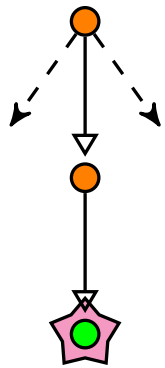
- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known



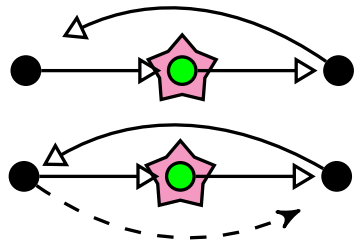


Keep track of exp--anded states on DFS

Early tag "safe" states

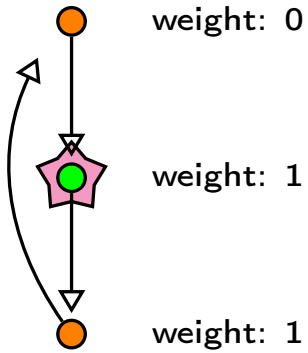
Deconstructing Evangelista and Pajault [2010] proviso

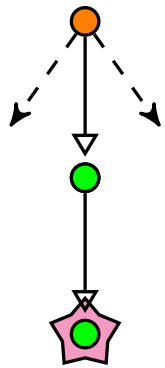
- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known



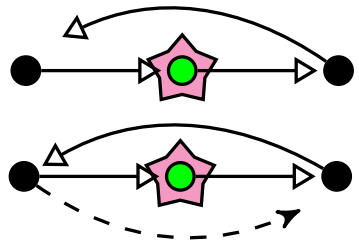


Keep track of exp--anded states on DFS

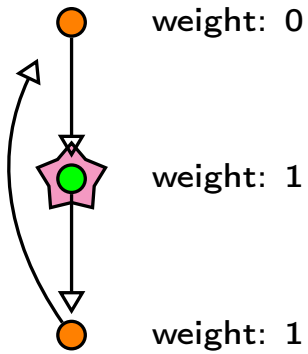
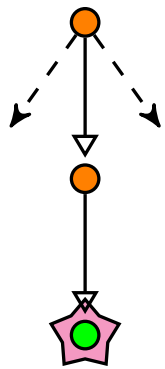
Early tag "safe" states

Deconstructing Evangelista and Pajault [2010] proviso

- Based on CondSource
- Try to reduce useless expansions:
- Must consider all closing-edges:



- Colors: safe, dangerous, on-dfs & not expanded

Weighted	Scan	Known
		
Keep track of exp--anded states on DFS	Early tag "safe" states	Prioritizing known successors

Evaluation of each optimization

	states (10^6)		transitions (10^6)		st/ms
Full	784.45	100.00%	2,677.73	100.00%	17.90
Source [Peled, 1994]	303.21	38.65%	679.16	25.36%	12.33
WeightedSource	263.43	33.58%	537.56	20.08%	11.68
WeightedSourceKnown[1]	262.63	33.48%	534.35	19.96%	11.77
CondSource	252.83	32.23%	518.80	19.37%	11.85
CondSourceKnown	251.05	32.00%	510.91	19.08%	11.89
WeightedSourceScan	250.49	31.93%	505.98	18.90%	11.67
WeightedSourceKnownScan[1]	248.11	31.63%	498.68	18.62%	11.70
None	57.58	7.34%	97.65	3.65%	22.65

- SOURCE have the best throughput
- Most of the improvement comes from COND
- Evangelista's provisos outperforms SOURCE

[1] [Evangelista and Pajault, 2010]

Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	

Systematically expands the source of a backegde

Provisos Based on Destination Expansion

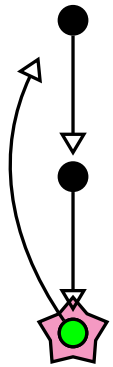
- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest

Systematically expands the source of a backegde

Provisos Based on Destination Expansion

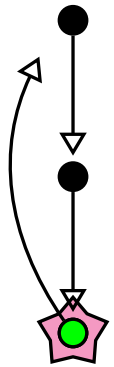
- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	

Systematically expands the source of a backegde

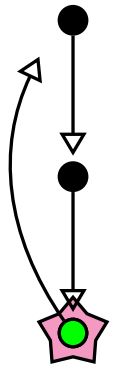
Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	
Systematically expands the source of a backegde	Systematically expands the destination of a backedge

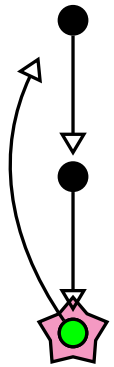
Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	
Systematically expands the source of a backegde	Systematically expands the destination of a backedge

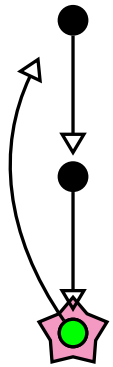
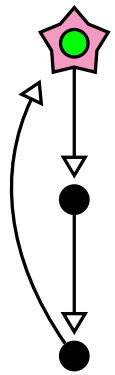
Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	
Systematically expands the source of a backegde	Systematically expands the destination of a backedge

Provisos Based on Destination Expansion

- Proposed by Nalumasu and Gopalakrishnan [2002] in a narrower context

Source	Dest
	
Systematically expands the source of a backegde	Systematically expands the destination of a backedge

Optimizations for these new provisos

- Compatible with: Cond, Weighted, Known

Mark for expansion ■ Already visited edge ⇾ Not yet visited edge ⇢

Optimizations for these new provisos

- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest

Mark for expansion ■ Already visited edge ⇾ Not yet visited edge ⇢

Optimizations for these new provisos

- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest
		

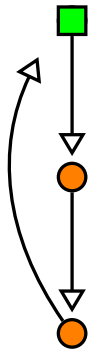
Reuse colors
Mark for expansion
Expand iff necessary

Mark for expansion ■ Already visited edge ⇾ Not yet visited edge ⇢

Optimizations for these new provisos

- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest

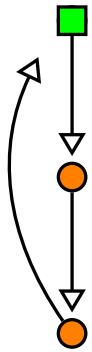


Reuse colors
Mark for expansion
Expand iff necessary

Mark for expansion ■ Already visited edge —▷ Not yet visited edge - ➤

Optimizations for these new provisos

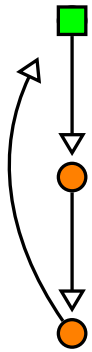
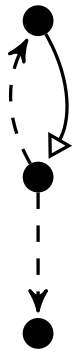
- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest
		
Reuse colors Mark for expansion Expand iff necessary	Prioritizing unknown successsors	

Mark for expansion ■ Already visited edge —▷ Not yet visited edge - ➤

Optimizations for these new provisos

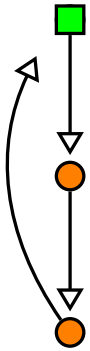
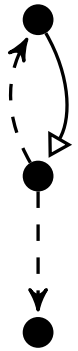
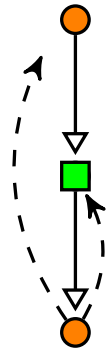
- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest
		
Reuse colors Mark for expansion Expand iff necessary	Prioritizing unknown successsors	Only mark the deepest dest. for expansion

Mark for expansion ■ Already visited edge —▷ Not yet visited edge - ➤

Optimizations for these new provisos

- Compatible with: Cond, Weighted, Known

Colored	Unknown	Deepest
		
Reuse colors Mark for expansion Expand iff necessary	Prioritizing unknown successsors	Only mark the deepest dest. for expansion

Mark for expansion ■ Already visited edge —▷ Not yet visited edge - ➤

Evaluation

	states (10^6)		transitions (10^6)		st/ms
DeepestDestUnknown	276.51	35.25%	570.52	21.31%	11.81
DeepestDest	275.31	35.10%	566.63	21.16%	11.87
WeightedDestUnknown	273.94	34.92%	563.61	21.05%	11.83
Dest	272.79	34.77%	508.17	18.98%	14.48
WeightedDest	272.68	34.76%	559.73	20.90%	11.80
WeightedSourceKnownScan	248.11	31.63%	498.68	18.62%	11.70
CondDest	213.98	27.28%	413.15	15.43%	12.57
CondDestUnknown	213.92	27.27%	412.75	15.41%	12.52
ColoredDest	213.92	27.27%	412.93	15.42%	12.54
ColoredDestUnknown	213.83	27.26%	412.27	15.40%	12.46

- CondDest outperforms state-of-the-art provisos
- Weighted and Deepest variants are disappointing

Improving Provisos With SCCs information

- When destination is red, an expansion is required:
  - Until now, the source was expanded

Improving Provisos With SCCs information

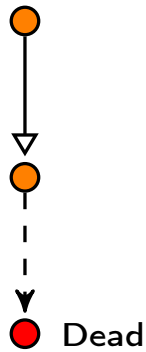
- When destination is red, an expansion is required:
  - Until now, the source was expanded

Dead	Highlinks

Improving Provisos With SCCs information

- When destination is red, an expansion is required:
  - Until now, the source was expanded

Dead	Highlinks

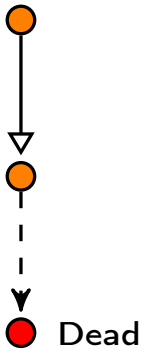


Avoid expansions when dest. is dead, i.e. in a fully visited SCC

Improving Provisos With SCCs information

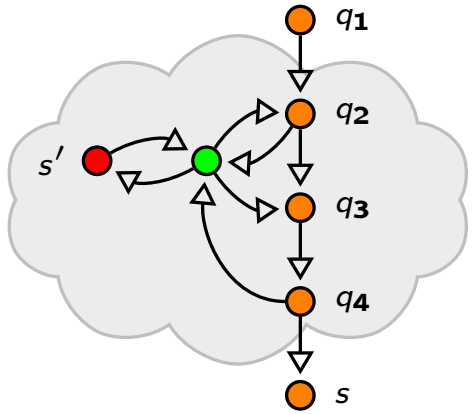
- When destination is red, an expansion is required:
  - Until now, the source was expanded

Dead



Avoid expansions when dest. is dead, i.e. in a fully visited SCC

Highlinks

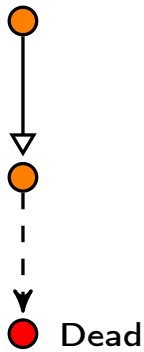


Adaptation of Deepest when dest. is not on the DFS and not dead

Improving Provisos With SCCs information

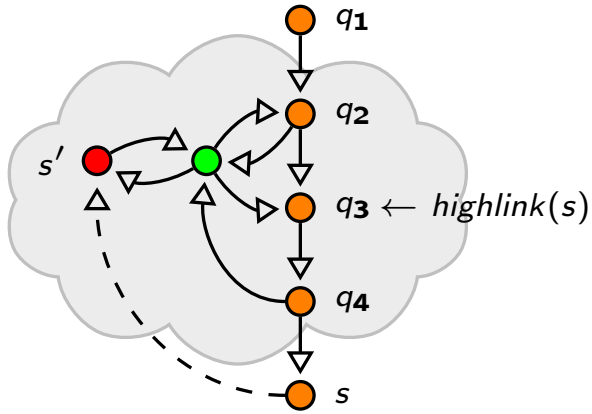
- When destination is red, an expansion is required:
  - Until now, the source was expanded

Dead



Avoid expansions when dest. is dead, i.e. in a fully visited SCC

Highlinks



Adaptation of Deepest when dest. is not on the DFS and not dead

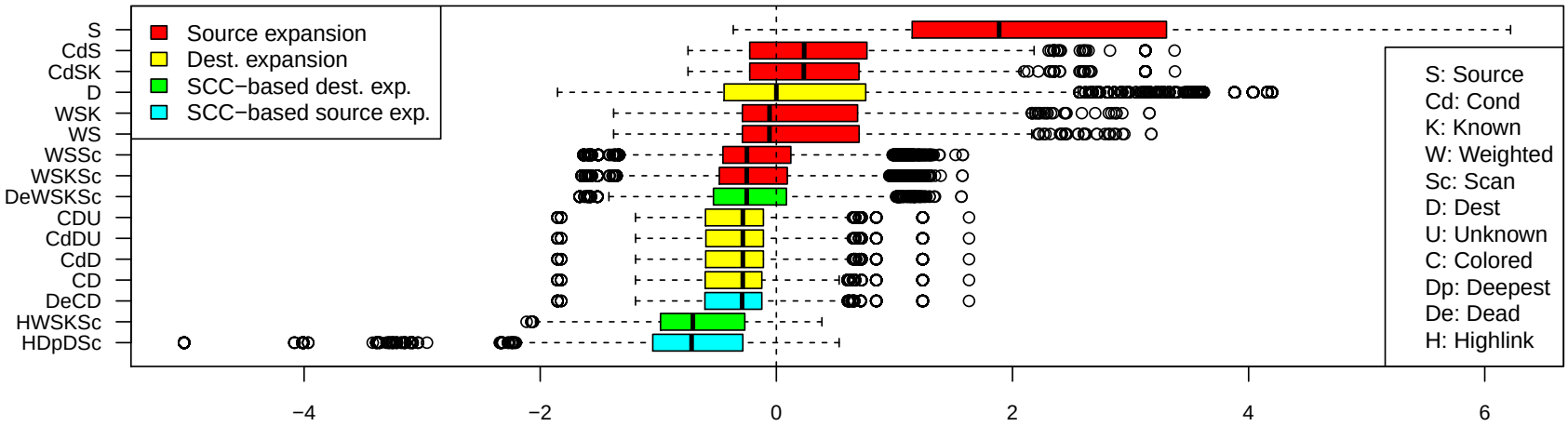
Dead and Highlinks are compatibles with both source and destination expansion-based provisos.

Evaluation 1/2

	states (10^6)		transitions (10^6)	
DeepestDest	275.31	35.10%	566.63	21.16%
DeadDeepestDest	269.10	34.30%	543.64	20.30%
WeightedDest	272.68	34.76%	559.73	20.90%
DeadWeightedDest	270.62	34.50%	554.91	20.72%
DeadWeightedSourceKnownScan	247.68	31.57%	497.79	18.59%
ColoredDest	213.92	27.27%	412.93	15.42%
DeadColoredDest	213.87	27.26%	412.80	15.42%
HighlinkWeightedDest	207.41	26.44%	393.22	14.68%
HighlinkWeightedDestScan	206.23	26.29%	391.05	14.60%
HighlinkWeightedSourceKnown	203.20	25.90%	386.84	14.45%
HighlinkWeightedSourceKnownScan	203.08	25.89%	386.60	14.44%
HighlinkDeepestDest	192.84	24.58%	349.89	13.07%
HighlinkDeepestDestScan	191.78	24.45%	347.95	12.99%

Evaluation 2/2

- Standard score for selected provisos
  - take the set of 1600 runs generated
  - compute a mean number μ_M for each model M
  - compute a standard deviation σ_M for each model M
  - standard score for a run r is then $\frac{states(r)-\mu_M}{\sigma_M}$
- Boxplot standard score



Conclusion

- Overview of state-of-the-art provisos for checking liveness properties
- New heuristics: Colored, Deepest, Dead, Highlink
- Combination with existing heuristics
- Intensive evaluation
- Independant of the reduction technique: ample set, sttuborn set, etc. (see [Laarman et al., 2014] for survey)

Our recommended provisos:

- CondDest in NDFS-based emptiness-checks
- HighlinkWeightedSourceKnown in SCC-based emptiness checks (no scan required)

Bibliography I

Evangelista, S. and Pajault, C. (2010). Solving the ignoring problem for partial order reduction. STTT, 12(2):155–170.

Laarman, A., Pater, E., Pol, J., and Hansen, H. (2014). Guard-based partial-order reduction. STTT, pages 1–22.

Nalumasu, R. and Gopalakrishnan, G. (2002). An efficient partial order reduction algorithm with an alternative proviso implementation. FMSD, 20(1):231–247.

Peled, D. (1994). Combining partial order reductions with on-the-fly model-checking. In Proceedings of the 6th International Conference on Computer Aided Verification (CAV'94), volume 818 of Lecture Notes in Computer Science, pages 377–390. Springer-Verlag.