Comparison of Symbolic Maximal End Component Decomposition Algorithms

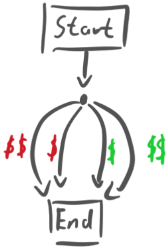
Felix Faber

RWTH Aachen University
Lehrstuhl für Informatik 2

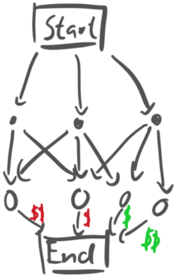
12.09.2023

(https://FelixFaber.dev/ for Thesis / Slides)

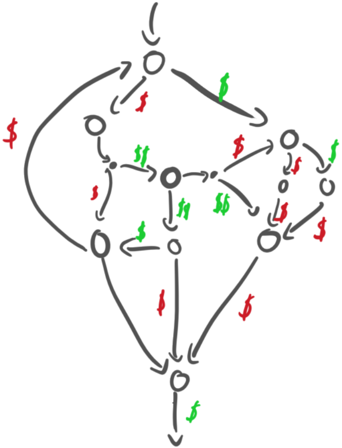
Motivation



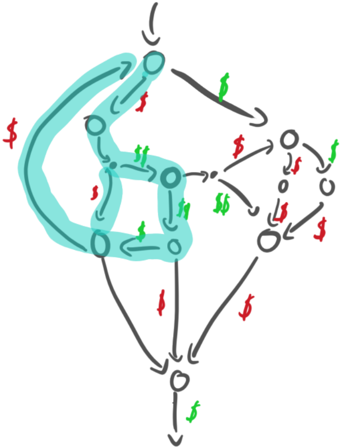
Motivation



Motivation



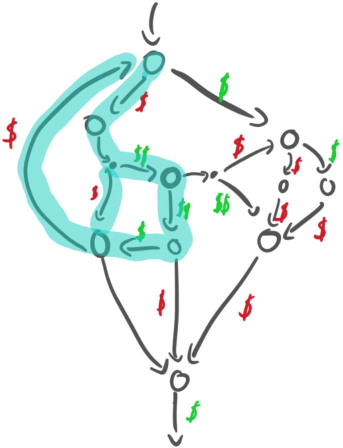
Motivation



Motivation

- Represent slot machine as model
- Find all "loops"
- Check for exploits

⇒ (Symbolic) model checking



Symbolic MEC Decomposition

$n := |V|, \;\; m := |E|$

Algorithm	Worst-Case Sym. Ops.	Worst-Case Sym. Space	Applicability
NAIVE [DA98, CHL+18]	$O(n^2)$	$O(\log n)$	G_{EBA}, G_{VBA}
LOCKSTEP [CHL+18]	$O(n\sqrt{m})$	$O(\sqrt{m})$	G_{EBA}, G_{VBA}
COLLAPSING [CDHS21]	$O(n^{2-\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	$O(n^{\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	G_{VBA}

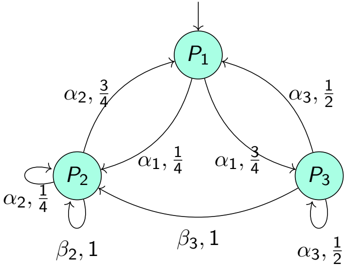
$\Rightarrow$ Performance in practice?

Structure

- MDPs: MECs on graph-like structures G_{EBA} and G_{VBA}
- Symbolic MDP representation using BDDs
- Symbolic MEC algorithms: Naive, LockStep, Collapsing
- Empirical evaluation of algorithms
- Summary and Future Work

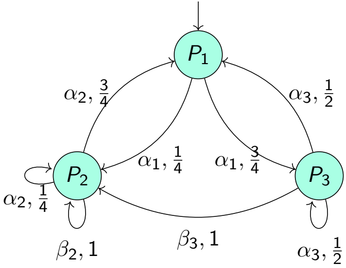
MDP

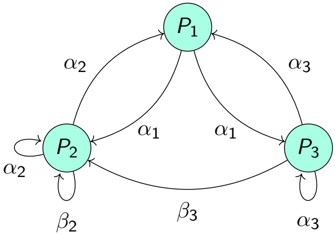
$$\mathcal{M} = (S, A, d_{\mathsf{init}}, \delta, r)$$



MDP

Edge-based actions G_{EBA}



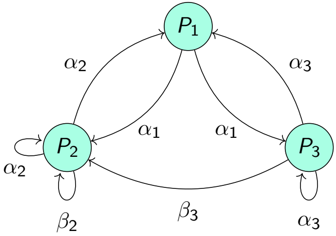


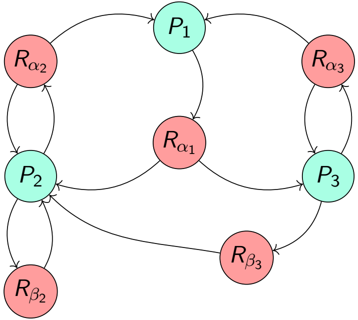
MDP

Edge-based actions G_{EBA}

vs

Vertex-based actions G_{VBA}





ECs

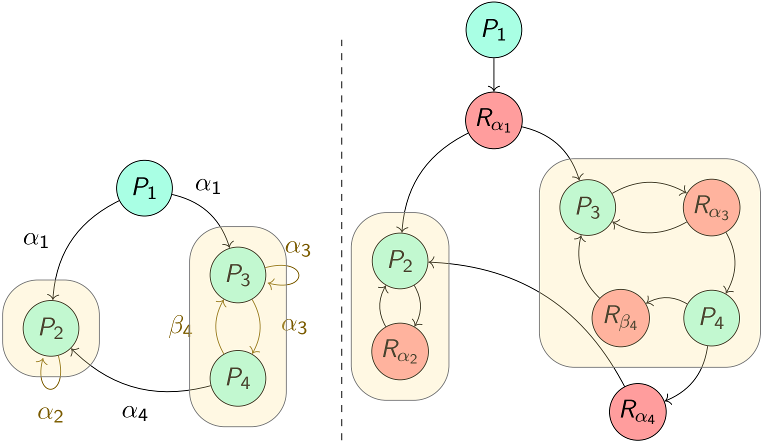
End Component:

- Set of states and actions
- Each state has at least one action
- All pairs of states of EC are reachable from another (using actions of EC)
- No outgoing actions
  - $\Rightarrow$ Player can stay within EC indefinitely

MECs

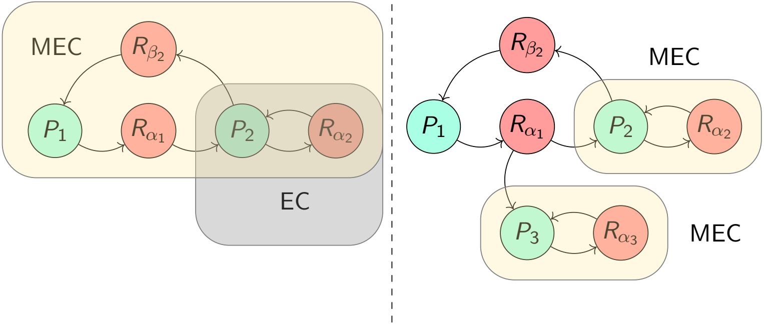
- MEC: Maximal EC (WRT set inclusion)
  $\Rightarrow$ Each state/action belongs to at most one MEC
- MEC decomposition: Compute all MECs of MDP $\mathcal{M}$

MECs



P_1
α_1
α_1
α_3
P_3
P_2
β_4
α_3
P_4
α_2
α_4
P_1
R_{α_1}
P_3
R_{α_3}
P_2
R_{β_4}
P_4
R_{α_2}
R_{α_4}

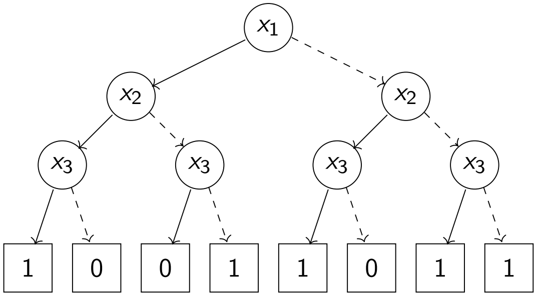
MECs


MEC
R_{β_2}
P_1
R_{α_1}
P_2
R_{α_2}
EC
R_{β_2}
MEC
P_1
R_{α_1}
P_2
R_{α_2}
P_3
R_{α_3}
MEC

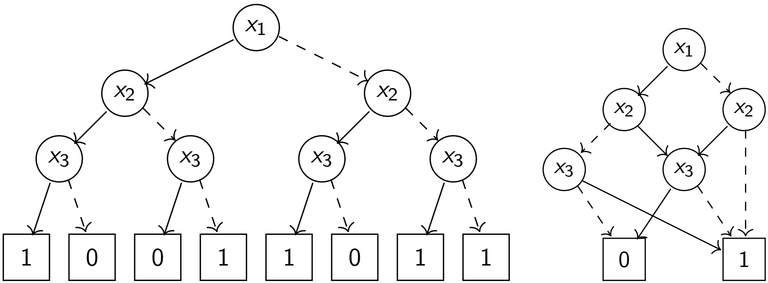
BDDs

Binary Decision Diagrams (BDDs)
In practice: Reduced Ordered BDDs (ROBDDs)

x_1	x_2	x_3	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



BDDs



MDP Representation: States

$$f_{V'} : \{0,1\}^t \to \{0,1\}, \quad f_{V'}(\underbrace{x_1, ..., x_t}_{\text{encodes } s \in V}) = \begin{cases} 1, & s \in V' \\ 0, & s \notin V' \end{cases}$$

G_{VBA}:

- f_{V_P} for Player Vertices V_P
- f_{V_R} for Random Vertices V_R

MDP Representation: Transitions

Transition BDD G_{VBA}:

$$t_{VBA}(\underbrace{x_1, \ldots, x_t}_{\substack{s \in V \\ \text{(Row)}}}, \underbrace{x'_1, \ldots, x'_t}_{\substack{s' \in V \\ \text{(Column)}}}) = \begin{cases} 1, & (s, s') \in E \\ 0, & (s, s') \notin E \end{cases}$$

Transition BDD G_{EBA}:

$$t_{EBA}(\underbrace{x_1, \ldots, x_t}_{\substack{s \in V \\ \text{(Row-Group)}}}, \underbrace{y_1, \ldots, y_u}_{\alpha \in A[s]}, \underbrace{x'_1, \ldots, x'_t}_{\substack{s' \in V \\ \text{(Column)}}}) = \begin{cases} 1, & (s, \alpha, s') \in E \\ 0, & (s, \alpha, s') \notin E \end{cases}$$

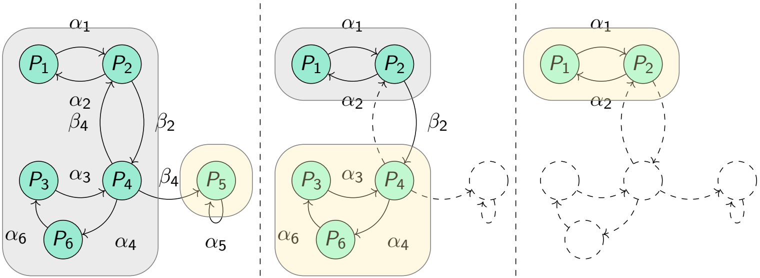
Symbolic Algorithms

- Need to utilize symbolic operations
- Runtime: symbolic operations ($Pre/Post$)
- Space: symbolic space (BDD)
- Assumes $O(n)$ symbolic SCC decomposition [GPP03]

$n := |V|, \quad m := |E|$

Algorithm	Worst-Case Sym. Ops.	Worst-Case Sym. Space	Applicability
Naive [DA98, CHL+18]	$O(n^2)$	$O(\log n)$	G_{EBA}, G_{VBA}
LockStep [CHL+18]	$O(n\sqrt{m})$	$O(\sqrt{m})$	G_{EBA}, G_{VBA}
Collapsing [CDHS21]	$O(n^{2-\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	$O(n^{\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	G_{VBA}

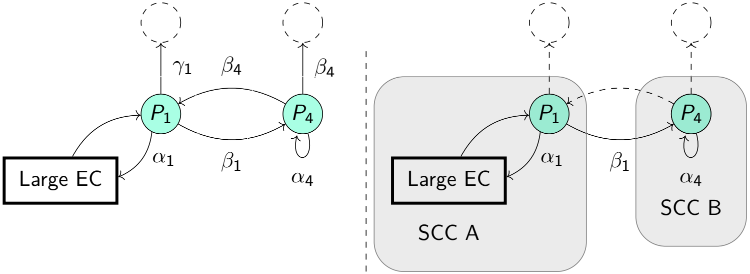
MEC Decomposition: Naive



Symbolic implementation of basic MEC decomposition algorithm

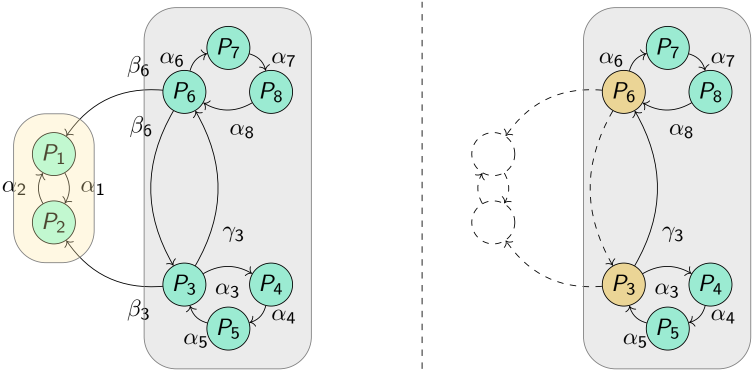
$\Rightarrow$ SCC decompositions and removal of outgoing actions

MEC Decomposition: LockStep



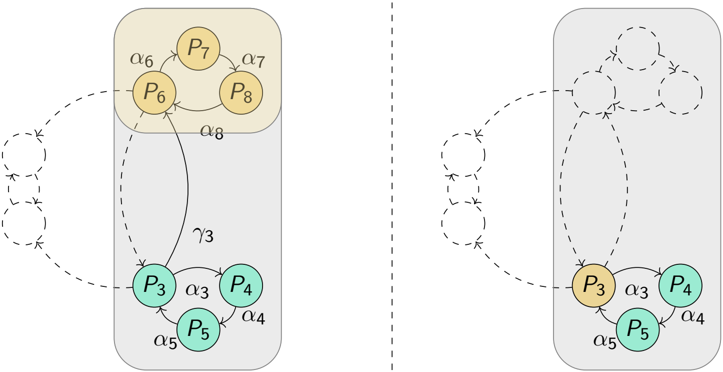
Idea: identify bottom SCCs to avoid duplicate vertex traversals

MEC Decomposition: LockStep

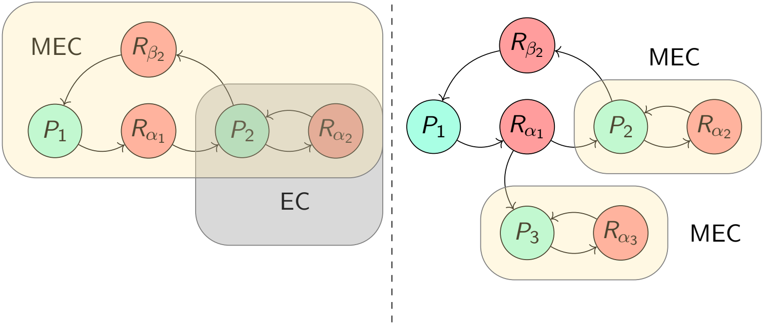


- Start lockstep search from each vertex which lost an edge.
- *Post* operations yield bottom SCC

MEC Decomposition: LockStep



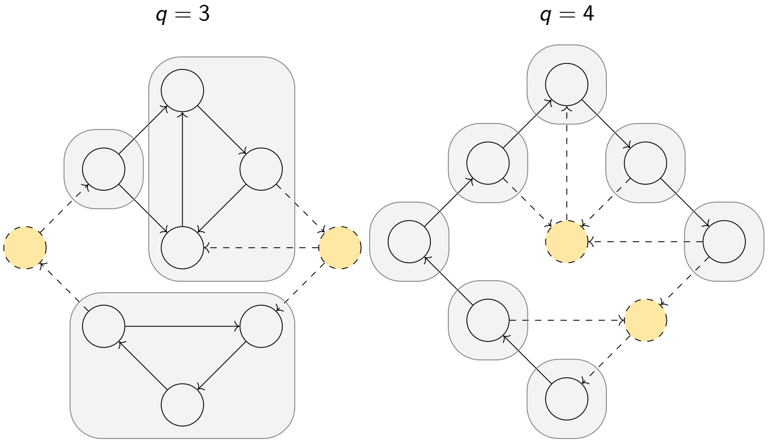
MEC Decomposition: Collapsing



Two passes:

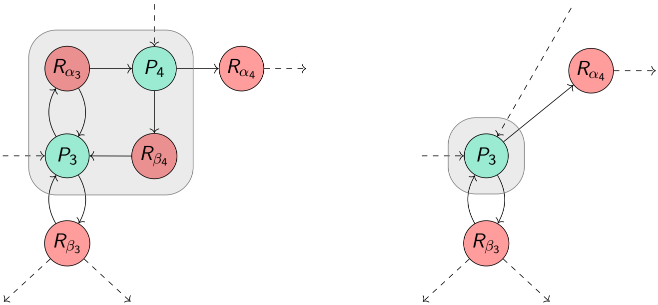
- Detect vertices of ECs quickly
- SCC decomposition on ECs $\Rightarrow$ MEC decomposition

MEC Decomposition: Collapsing



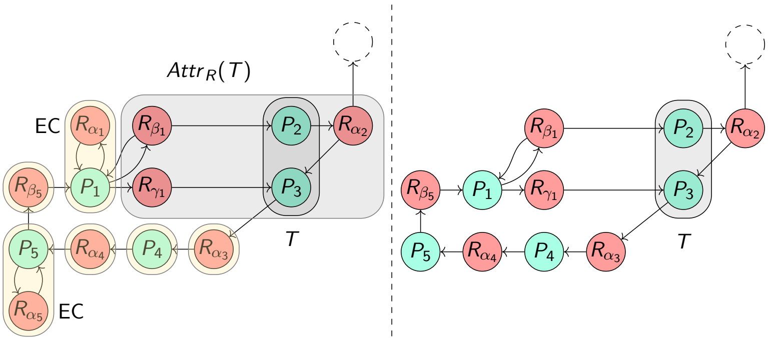
- Split up large SCCs using separator.
- Process smaller SCCs
- Re-add separator

MEC Decomposition: Collapsing

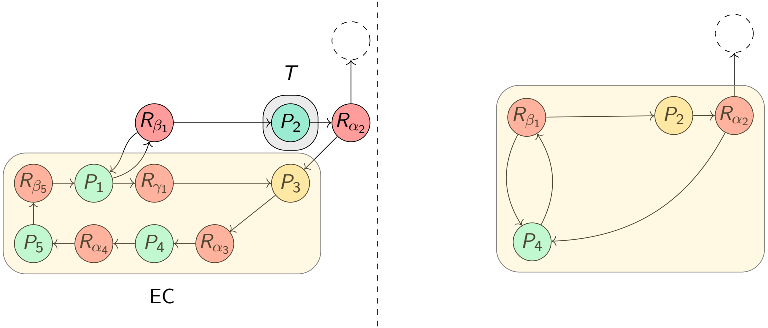


Retraversal of ECs cheaper after collapsing

MEC Decomposition: Collapsing

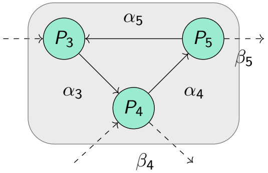


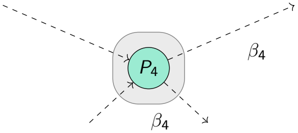
MEC Decomposition: Collapsing



MEC Decomposition: Collapsing

$$t_{EBA}(\underbrace{x_1, \ldots, x_t}_{\substack{\text{Source Vertex} \\ s \in V}}, \underbrace{y_1, \ldots, y_u}_{\substack{\text{Action} \\ \alpha \in A[s]}}, \underbrace{x'_1, \ldots, x'_t}_{\substack{\text{Destination Vertex} \\ s' \in V}}) = \begin{cases} 1, & (s, \alpha, s') \in E \\ 0, & (s, \alpha, s') \notin E \end{cases}$$





Evaluation

$n := |V|, \quad m := |E|$

Algorithm	Worst-Case Sym. Ops.	Worst-Case Sym. Space	Applicability
NAIVE [DA98, CHL+18]	$O(n^2)$	$O(\log n)$	G_{EBA}, G_{VBA}
LOCKSTEP [CHL+18]	$O(n\sqrt{m})$	$O(\sqrt{m})$	G_{EBA}, G_{VBA}
COLLAPSING [CDHS21]	$O(n^{2-\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	$O(n^{\epsilon} \log n)$ $(0 < \epsilon \leq 0.5)$	G_{VBA}

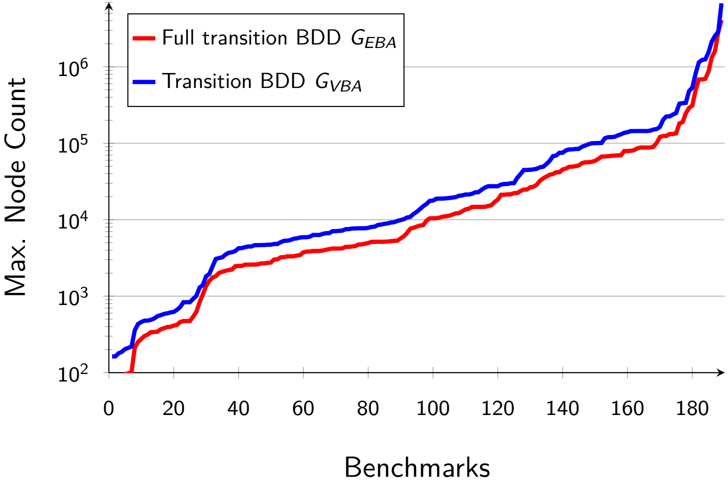
Performance in practice?

$\Rightarrow$ Empirical evaluation using

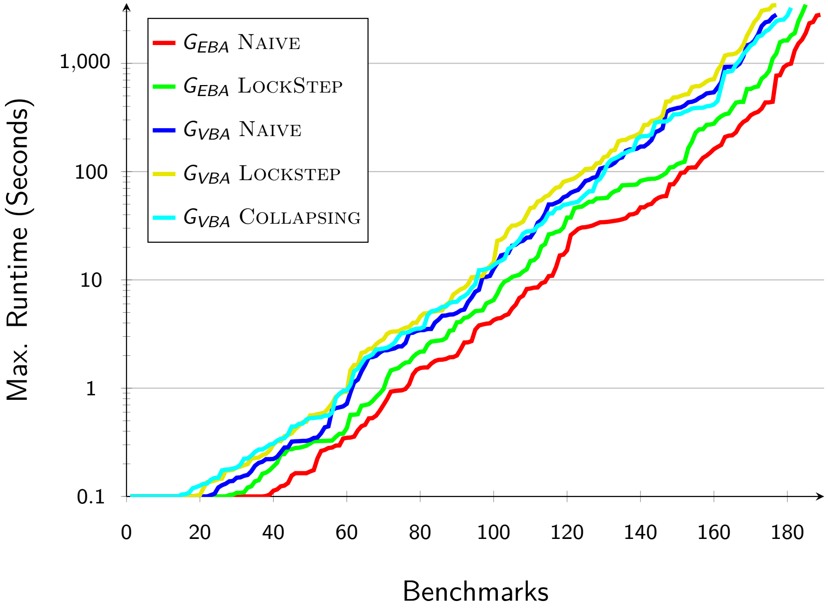
- STORM
- quantitative verification benchmark set (QVBS)
- 1 Hour time limit

Evaluation

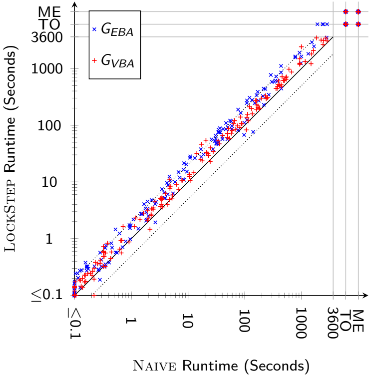
- STORM assumes $G_{EBA} \Rightarrow$ convert to G_{VBA} if desired
- 189 Benchmarks for runtimes
- 177 Benchmarks for symbolic operations

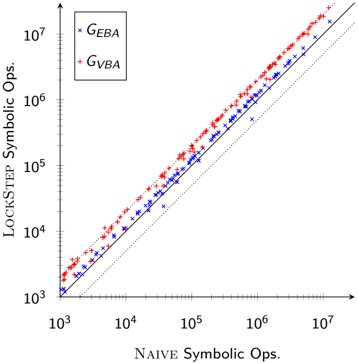


Evaluation: Runtime Overview

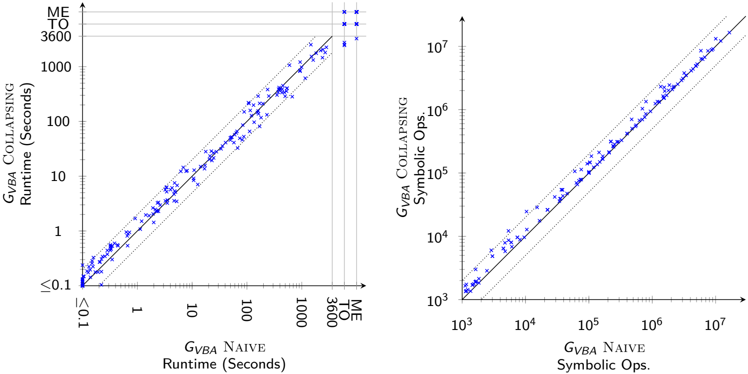


Evaluation: LockStep

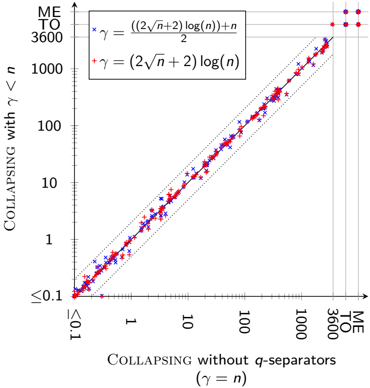


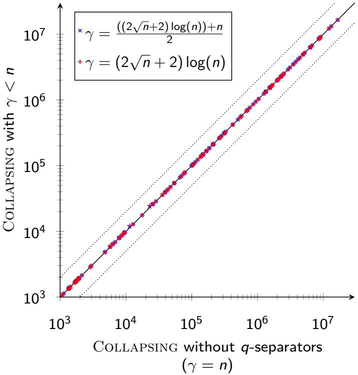


Evaluation: Collapsing



Evaluation: Collapsing





Summary

- Theoretical worst-case improvements are not reflected in empirical results
- Least amount of Sym. Ops.: NAIVE
- Best Runtime: NAIVE or COLLAPSING
  - But COLLAPSING is (at the moment) only applicable to G_{VBA}

Future Work

- "Native" G_{VBA} implementation
- COLLAPSING implementation on G_{EBA}
- Improvements in SCC decomposition [LSS+23]
- Comparison of *explicit* MEC decomposition algorithms

References I

[CDHS21] Krishnendu Chatterjee, Wolfgang Dvořák, Monika Henzinger, and Alexander Svozil. Symbolic time and space tradeoffs for probabilistic verification. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13. IEEE, 2021.

[CHL+18] Krishnendu Chatterjee, Monika Henzinger, Veronika Loitzenbauer, Simin Oraee, and Viktor Toman. Symbolic algorithms for graphs and Markov decision processes with fairness objectives. In *International Conference on Computer Aided Verification*, pages 178–197. Springer, 2018.

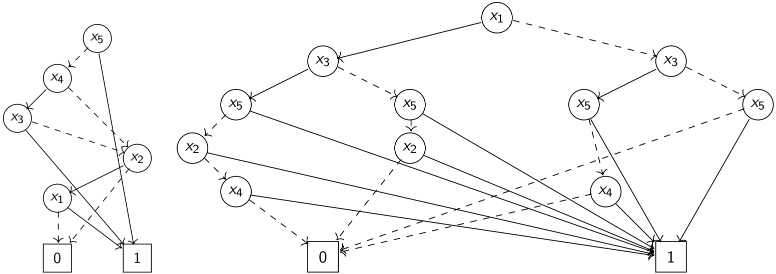
[DA98] Luca De Alfaro. *Formal verification of probabilistic systems*. stanford university, 1998.

References II

[GPP03] Raffaella Gentilini, Carla Piazza, and Alberto Policriti. Computing strongly connected components in a linear number of symbolic steps. In *SODA*, volume 3, pages 573–582, 2003.

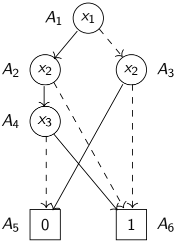
[LSS+23] Casper Abild Larsen, Simon Meldahl Schmidt, Jesper Steensgaard, Anna Blume Jakobsen, Jaco van de Pol, and Andreas Pavlogiannis. A Truly Symbolic Linear-Time Algorithm for SCC Decomposition. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 353–371. Springer, 2023.

ROBDDs: Variable Ordering

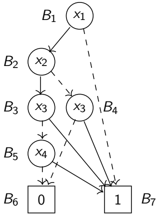


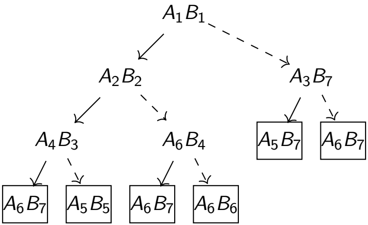
ROBDDs: AND Operation

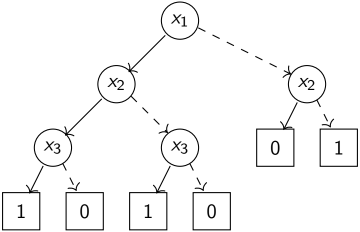
$f(x_1, ..., x_4) = x_1 x_3 + \overline{x_2}$



$g(x_1, ..., x_4) = x_2 x_4 + x_3 + \overline{x_1}$

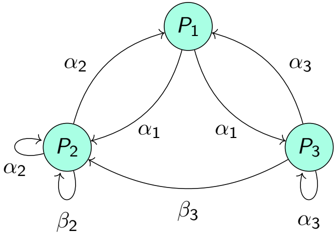


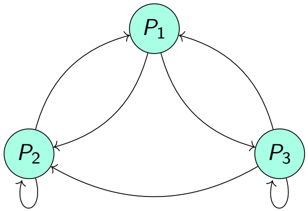




MDP: Directed Graph of G_{EBA}

Directed graph of G_{EBA}





MDP Representation: Explicit Transitions

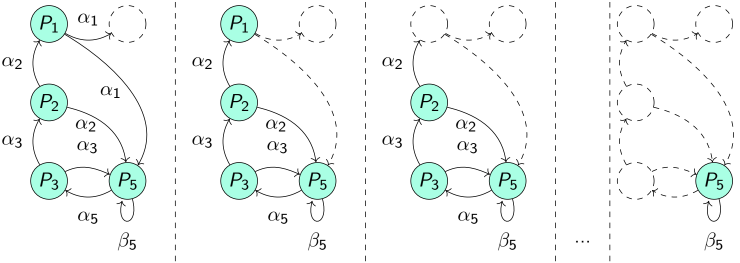
$$
\begin{array}{cc}
\begin{bmatrix}
t_{11} & \cdots & t_{1n} \\
t_{21} & \cdots & t_{2n} \\
\vdots & \ddots & \vdots \\
t_{n1} & \cdots & t_{nn}
\end{bmatrix}
&
\begin{array}{l}
s_1 \\
s_2 \\
\\
s_n
\end{array}
\end{array}
$$

$$G_{VBA}$$

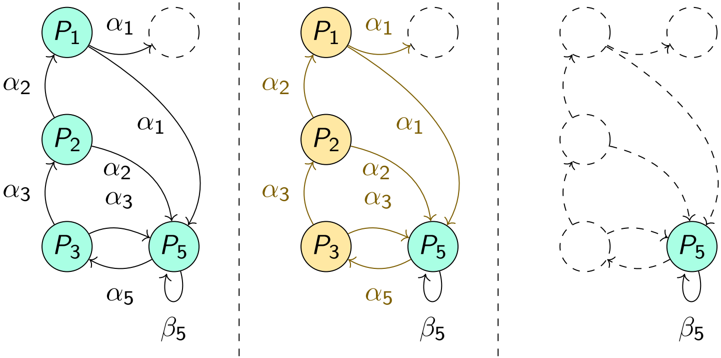
$$
\begin{array}{cc}
\left[\begin{array}{ccc}
t_{11} & \cdots & t_{1n} \\
t_{21} & \cdots & t_{2n} \\
\hline
t_{31} & \cdots & t_{3n} \\
\vdots & \ddots & \vdots \\
\hline
\vdots & \ddots & \vdots \\
\hline
\vdots & \ddots & \vdots \\
t_{m1} & \cdots & t_{mn}
\end{array}\right]
&
\begin{array}{l}
\left.\begin{array}{c} \\ \\ \end{array}\right\} \text{Row Group of } s_1 \\
\left.\begin{array}{c} \\ \\ \end{array}\right\} \text{Row Group of } s_2 \\
\\
\left.\begin{array}{c} \\ \\ \end{array}\right\} \text{Row Group of } s_n
\end{array}
\end{array}
$$

$$G_{EBA}$$

NAIVE Worst-Case



Random Attractor Removal Optimization



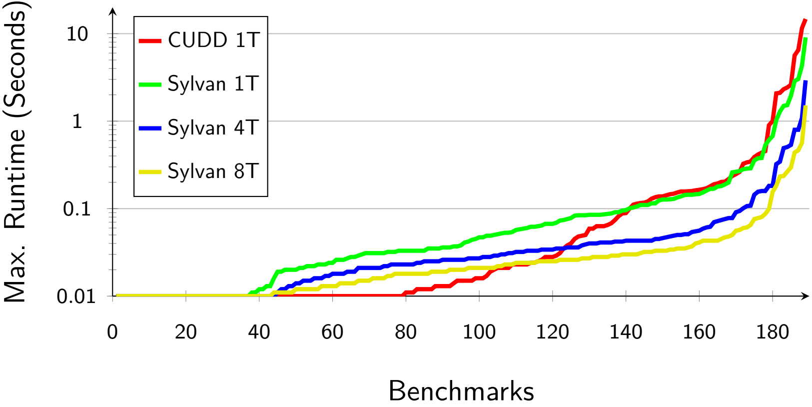
G_{EBA} to G_{VBA} Conversion

$$\text{Have:}\quad t_{EBA}(\underbrace{x_1, \ldots, x_t}_{\text{Source Vertex}}, \underbrace{y_1, \ldots, y_u}_{\text{Action}}, \underbrace{x'_1, \ldots, x'_t}_{\text{Target Vertex}})$$

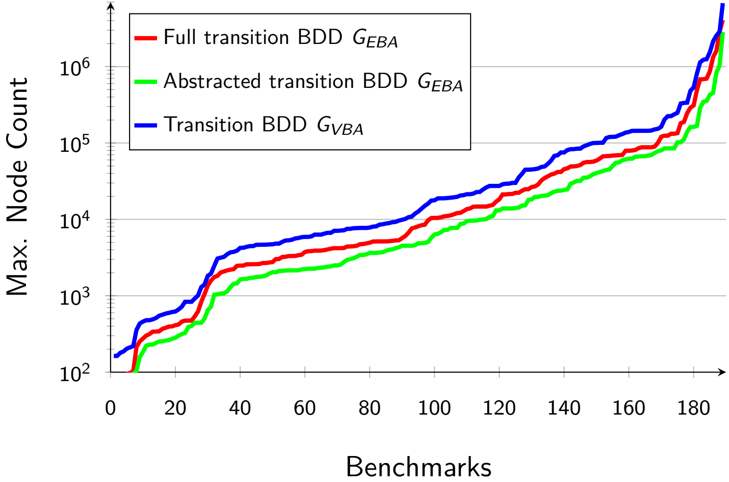
$$\text{Want:}\quad t_{VBA}(\underbrace{x_1, \ldots, \ldots, \ldots, \ldots, x_s}_{\text{Source Vertex}}, \underbrace{x'_1, \ldots, \ldots, \ldots, \ldots, x'_s}_{\text{Target Vertex}})$$

$$\text{Conversion:}\quad t_{VBA}(\underbrace{x_1, \ldots, x_t, y_1, \ldots, y_u, {\color{red}z}}_{\text{Source Vertex}}, \underbrace{x'_1, \ldots, x'_t, {\color{red}y'_1}, {\color{red}\ldots}, {\color{red}y'_u}, {\color{red}z'}}_{\text{Target Vertex}})$$

$G_{EBA} \rightarrow G_{VBA}$ Conversion Runtimes



$G_{EBA} \rightarrow G_{VBA}$ Conversion Full Quantile Plot



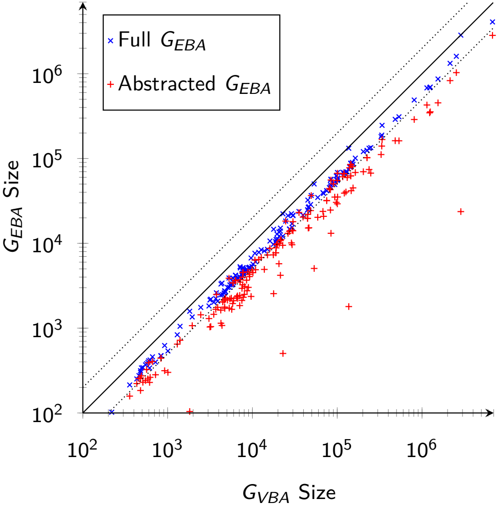
All Transition BDDs

$$\text{Full:}\quad t_{EBA}(\underbrace{x_1,\ldots,x_t}_{\text{Source Vertex}}, \underbrace{y_1,\ldots,y_u}_{\text{Action}}, \underbrace{x'_1,\ldots,x'_t}_{\text{Target Vertex}})$$

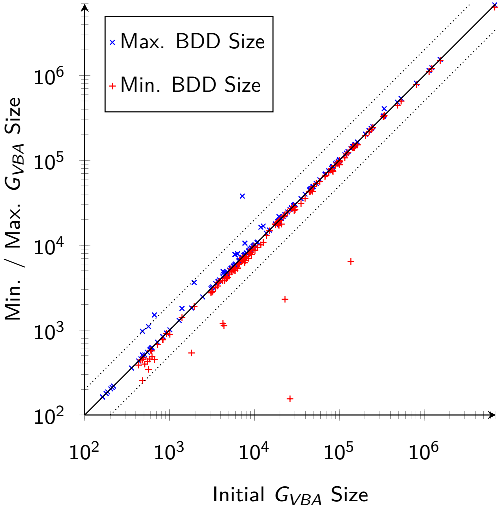
$$\text{Abstracted:}\quad t_{EBA}(\underbrace{x_1,\ldots,x_t}_{\text{Source Vertex}}, \underbrace{x'_1,\ldots,x'_t}_{\text{Target Vertex}})$$

$$\text{Converted:}\quad t_{VBA}(\underbrace{x_1,\ldots,x_t,y_1,\ldots,y_u,z}_{\text{Source Vertex}}, \underbrace{x'_1,\ldots,x'_t,y'_1,\ldots,y'_u,z'}_{\text{Target Vertex}})$$

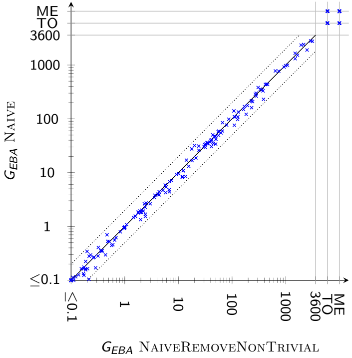
$G_{EBA} \rightarrow G_{VBA}$ Conversion BDD Sizes

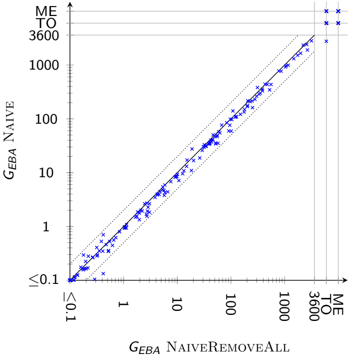


Collapsing BDD Sizes

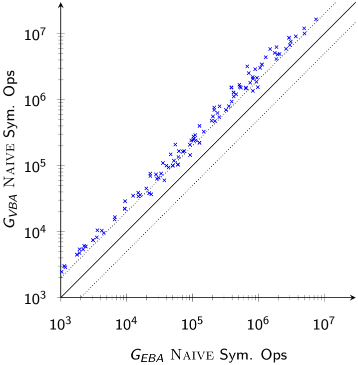


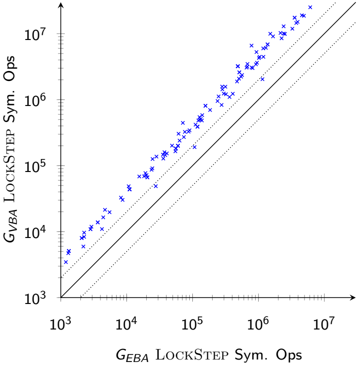
NAIVE Variants Runtime



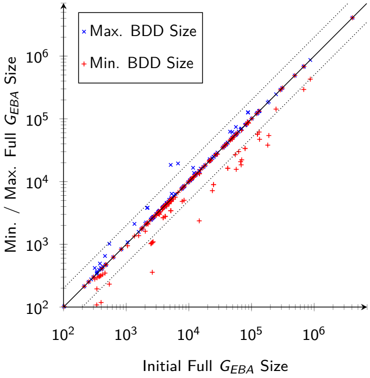


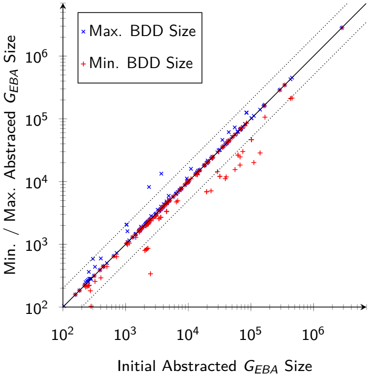
G_{EBA} vs G_{VBA} Sym. Ops



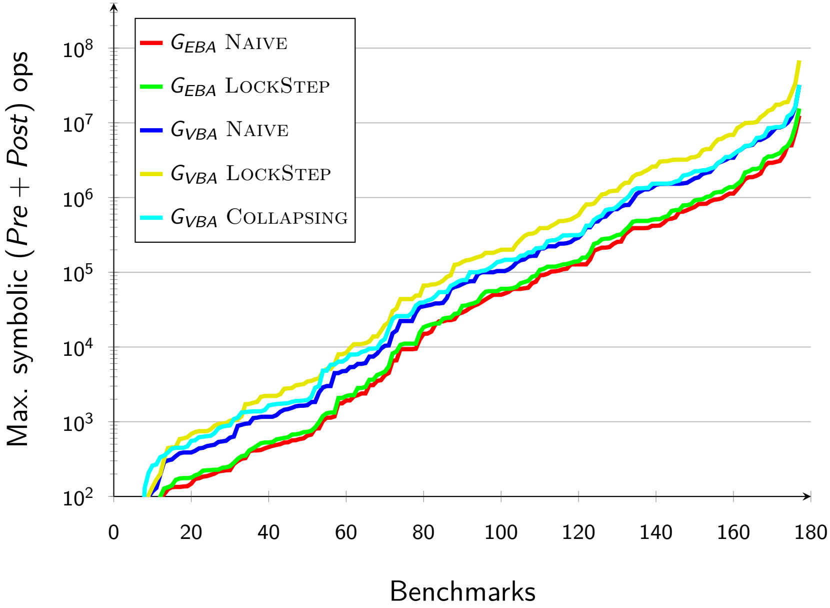


G_{EBA} BDD Sizes





Quantile Plot of Sym. Ops (*Pre* + *Post*)



Quantile Plot of Sym. Ops (All)