HyperProb:
A Model Checker for Probabilistic Hyperproperties

24th International Symposium on Formal methods '21

Oyendrila Dobe†, Erika Ábrahám*, Ezio Bartocci**, Borzoo Bonakdarpour†

†Michigan State University (USA), *RWTH Aachen (Germany),

****TU-Wien (Austria)**

November 26, 2021

Hyperproperty [Clarkson, Schneider, 2010]

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, 2014]

- A trace property is a set of traces.

Hyperproperty [Clarkson, Schneider, 2010]

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, 2014]

- A trace property is a set of traces.
- Classical trace properties cannot express relations between traces!

Hyperproperty [Clarkson, Schneider, 2010]

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, 2014]

- A trace property is a set of traces.
- Classical trace properties cannot express relations between traces!
- A hyperproperty is a set of sets of traces.

Hyperproperty [Clarkson, Schneider, 2010]

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, 2014]

- A trace property is a set of traces.
- Classical trace properties cannot express relations between traces!
- A hyperproperty is a set of sets of traces.

HyperLTL property

I drink coffee every day at the same time.

$$\forall \pi.\ \forall \pi'.\quad (\square\ (\mathit{coffee}_{\pi} \Leftrightarrow \mathit{coffee}_{\pi'}))$$

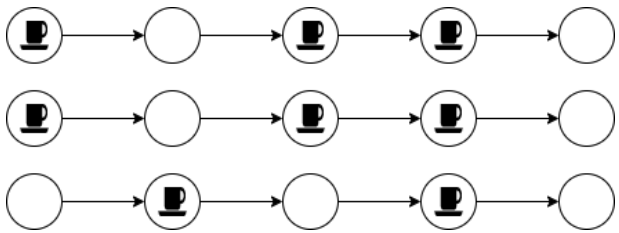


Figure: Example of set of traces

HyperPCTL [HyperPCTL for DTMCs by Ábrahám, Bonakdarpour, ’18]

[HyperPCTL for MDPs by Ábrahám, Bartocci, Bonakdarpour, Dobe, ’20]

- Express probabilistic relations between traces.

HyperPCTL [HyperPCTL for DTMCs by Ábrahám, Bonakdarpour, '18]

[HyperPCTL for MDPs by Ábrahám, Bartocci, Bonakdarpour, Dobe, '20]

- Express probabilistic relations between traces.
- Quantify ($Q_i \in \{\exists, \forall\}$) over schedulers and initial states:

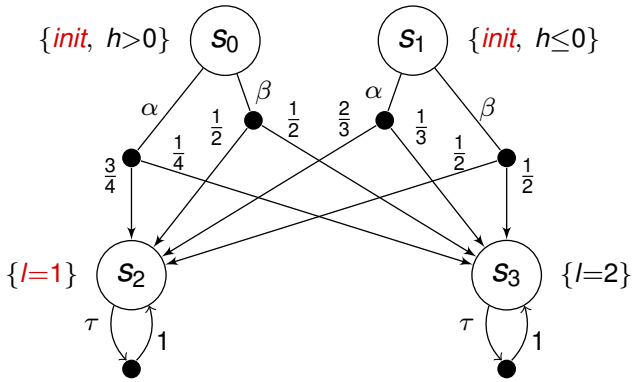
$$\underbrace{Q_{\hat{\sigma}_1}\hat{\sigma}_1 \ldots Q_{\hat{\sigma}_m}\hat{\sigma}_m.}_{\textit{scheduler quantification}} \quad \underbrace{Q_{\hat{s}_1}\hat{s}_1(\hat{\sigma}_1)\ldots Q_{\hat{s}_n}\hat{s}_n(\hat{\sigma}_n)}_{\textit{state quantification}}. \quad \underbrace{\psi}_{\textit{non-quantified HyperPCTL}}$$

HyperPCTL [HyperPCTL for DTMCs by Ábrahám, Bonakdarpour, '18]

[HyperPCTL for MDPs by Ábrahám, Bartocci, Bonakdarpour, Dobe, '20]

- Express probabilistic relations between traces.
- Quantify ($Q_i \in \{\exists, \forall\}$) over schedulers and initial states:

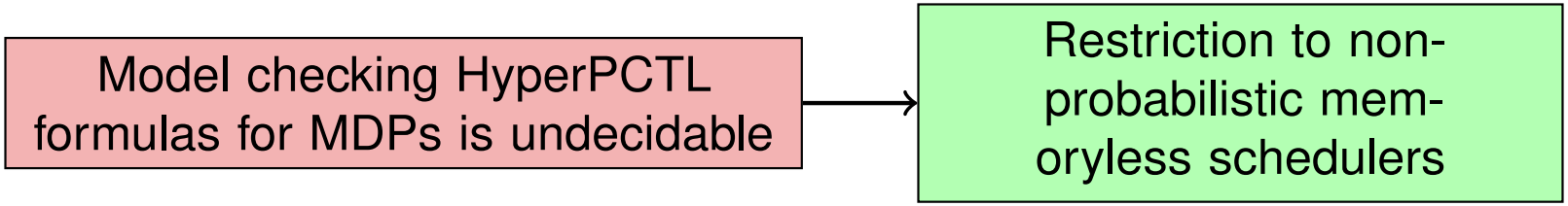
$$\underbrace{Q_{\hat{\sigma}_1}\hat{\sigma}_1 \ldots Q_{\hat{\sigma}_m}\hat{\sigma}_m.}_{\textit{scheduler quantification}} \quad \underbrace{Q_{\hat{s}_1}\hat{s}_1(\hat{\sigma}_1)\ldots Q_{\hat{s}_n}\hat{s}_n(\hat{\sigma}_n)}_{\textit{state quantification}}. \quad \underbrace{\psi}_{\textit{non-quantified HyperPCTL}}$$



$$\forall\hat{\sigma}.\forall\hat{s}(\hat{\sigma}).\forall\hat{s}'(\hat{\sigma}).\left(init_{\hat{s}} \wedge init_{\hat{s}'} \wedge h_{\hat{s}} \neq h_{\hat{s}'}\right) \Rightarrow \left(\mathbb{P}\left(\Diamond(l{=}1)_{\hat{s}}\right) = \mathbb{P}\left(\Diamond(l{=}1)_{\hat{s}'}\right)\right)$$

Figure: Example of an MDP.

Restricted HyperPCTL model checking



Restricted HyperPCTL model checking

Model checking HyperPCTL formulas for MDPs is undecidable → Restriction to non-probabilistic memoryless schedulers

In HyperProb,

$\mathcal{M} = (S, Act, Pr, AP, L)$
satisfies
$Q\hat{\sigma}.Q_1\hat{s}_1(\hat{\sigma}).\ldots.Q_n\hat{s}_n(\hat{\sigma}).\varphi^{nq}$

iff

SMT encoding is satisfied

Currently, HyperProb allows one scheduler quantifier.

Overview of the tool

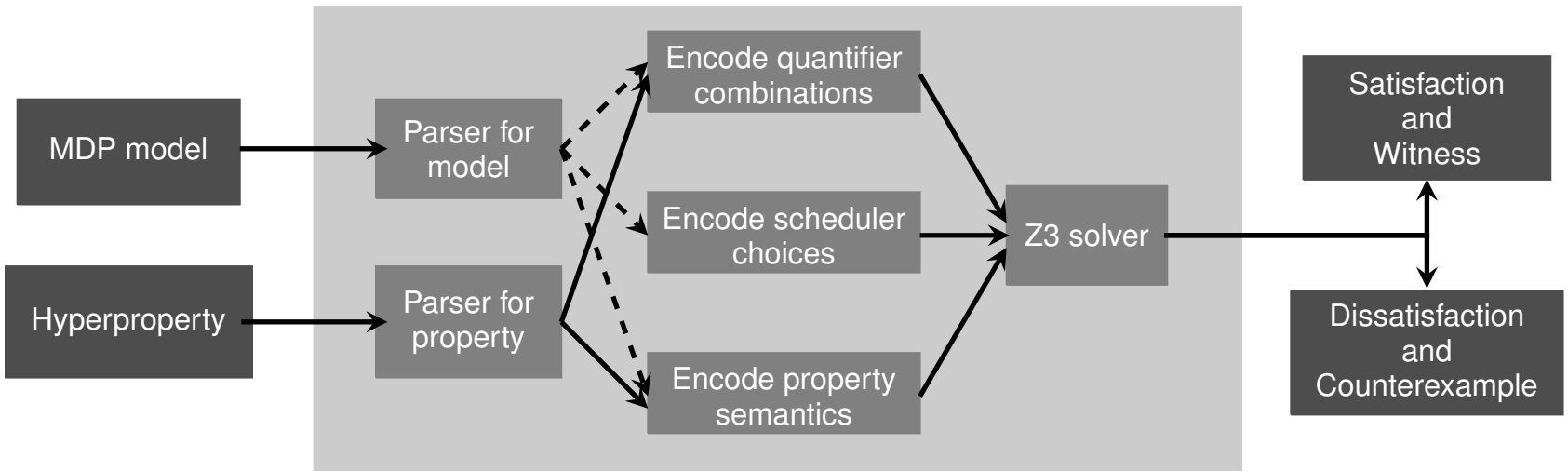


Figure: Dataflow inside the tool

- Input: Prism model (.nm file) and HyperPCTL formula (String).
- Output: Verification Result (Boolean) & Witness / Counterexample (sequence of actions).

Example output

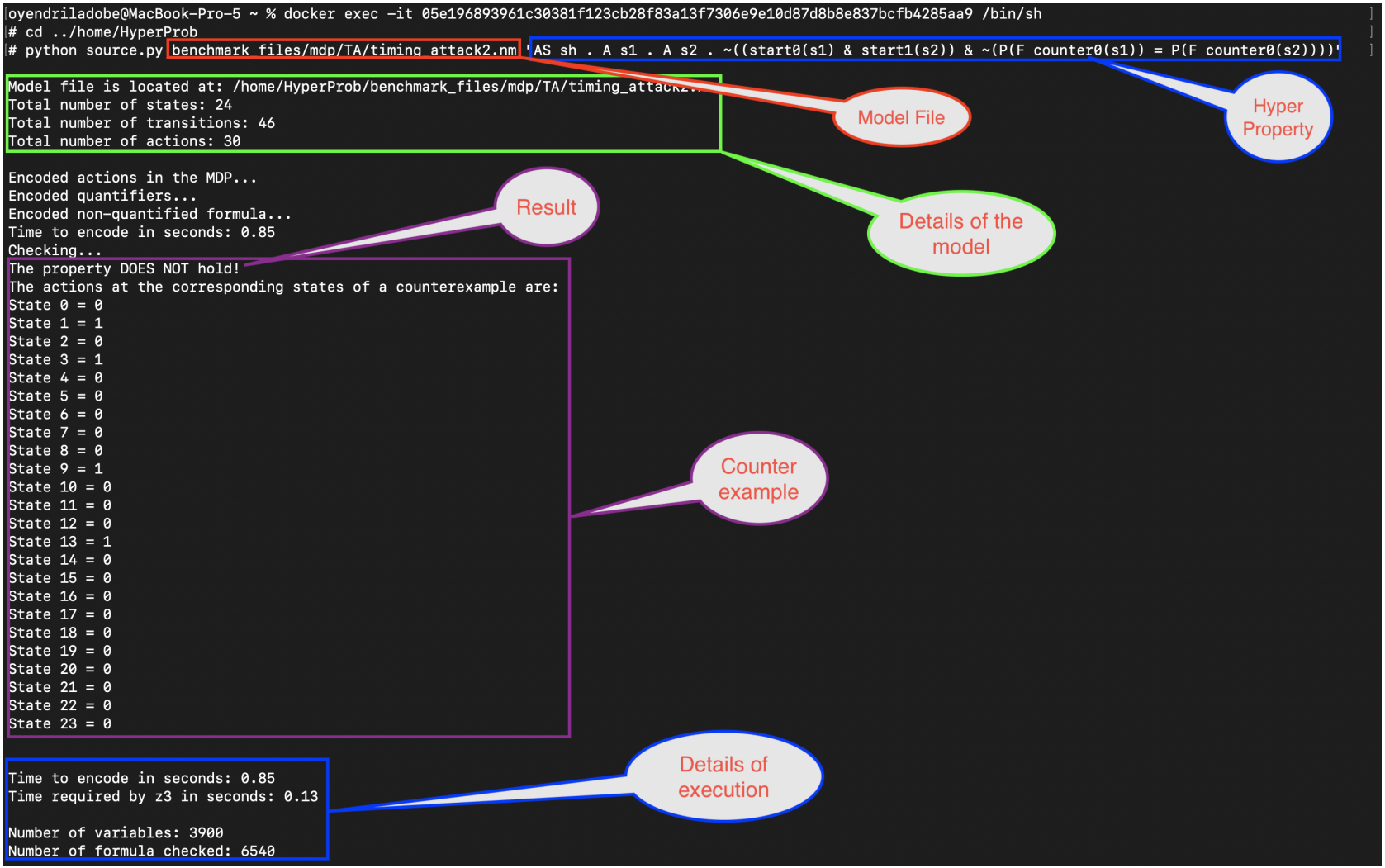


Figure: Sample execution

Evaluation

Case Study		Running time(s)						#SMT variables		#op	#st	#tr
		SE		SS		Total						
		N	O	N	O	N	O	N	O			
TA	$m=2$	5	2	<1	<1	5	2	8088	2520	14	24	46
	$m=4$	114	18	20	1	134	19	50460	14940		60	136
	$m=6$	1721	140	865	45	2586	185	175728	51184		112	274
	$m=8$	12585	952	**TO**	426	**TO**	1378	388980	131220		180	460
PW	$m=2$	5	2	<1	<1	6	3	8088	2520	14	24	46
	$m=4$	207	26	40	1	247	27	68670	20230		70	146
	$m=6$	3980	331	1099	41	5079	372	274540	79660		140	302
	$m=8$	26885	2636	**TO**	364	**TO**	3000	657306	221130		234	514
TS	$h=(0,1)$	<1	<1	<1	<1	1	1	1379	441	28	7	13
	$h=(0,15)$	60	8	1607	<1	1667	8	34335	8085		35	83
	$h=(4,8)$	12	3	17	<1	29	3	12369	3087		21	48
	$h=(8,15)$	60	8	1606	<1	1666	8	34335	8085		35	83
	$h=(10,20)$	186	19	13707	1	13893	20	52695	13095		45	108
PC	s=(0)	277	10	1996	5	2273	15	21220	6780	44	20	188
	s=(0,1)	822	13	5808	5	6630	18	21220	6780		20	340
	s=(0..2)	1690	15	**TO**	5	**TO**	20	21220	6780		20	494
	s=(0..3)	4631	16	**TO**	7	**TO**	23	21220	6780		20	648
	s=(0..4)	7353	22	**TO**	21	**TO**	43	21220	6780		20	802
	s=(0..5)	10661	19	**TO**	61	**TO**	80	21220	6780		20	956
	s=(0..6)	13320	18	**TO**	41	**TO**	59	21220	6780		20	1110

Table: **TA:** Timing attack. **PW:** Password leakage. **TS:** Thread scheduling. **PC:** Probabilistic conformance. **TO:** Timeout. **N:** Prototype presented in previous work. **O:** HyperProb,. **SE:** SMT encoding. **SS:** SMT solving. #op: Formula size (number of operators). #st: Number of states. #tr: Number of transitions.

Summary

1. Presented a SMT-based model checking algorithm which is NP-complete (coNP-complete for universal quantifier) in the state set size of the input MDP.
2. Provided a docker container with the pre-installed dependencies, to run the tool.

`https://github.com/TART-MSU/HyperProb`