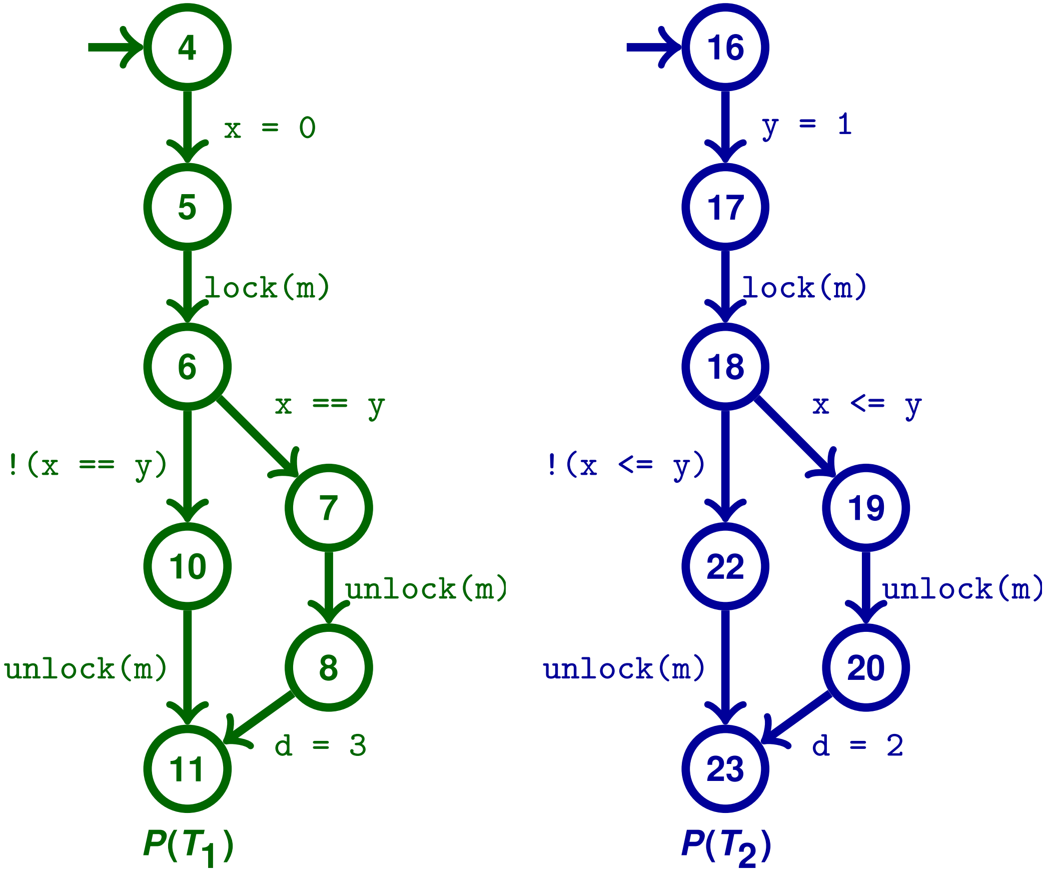
Verification of concurrent programs using trace abstraction refinement

Franck Cassez
Macquarie University
Sydney, Australia

Frowin Ziegler
University of Augsburg
Augsburg, Germany

Problem Statement

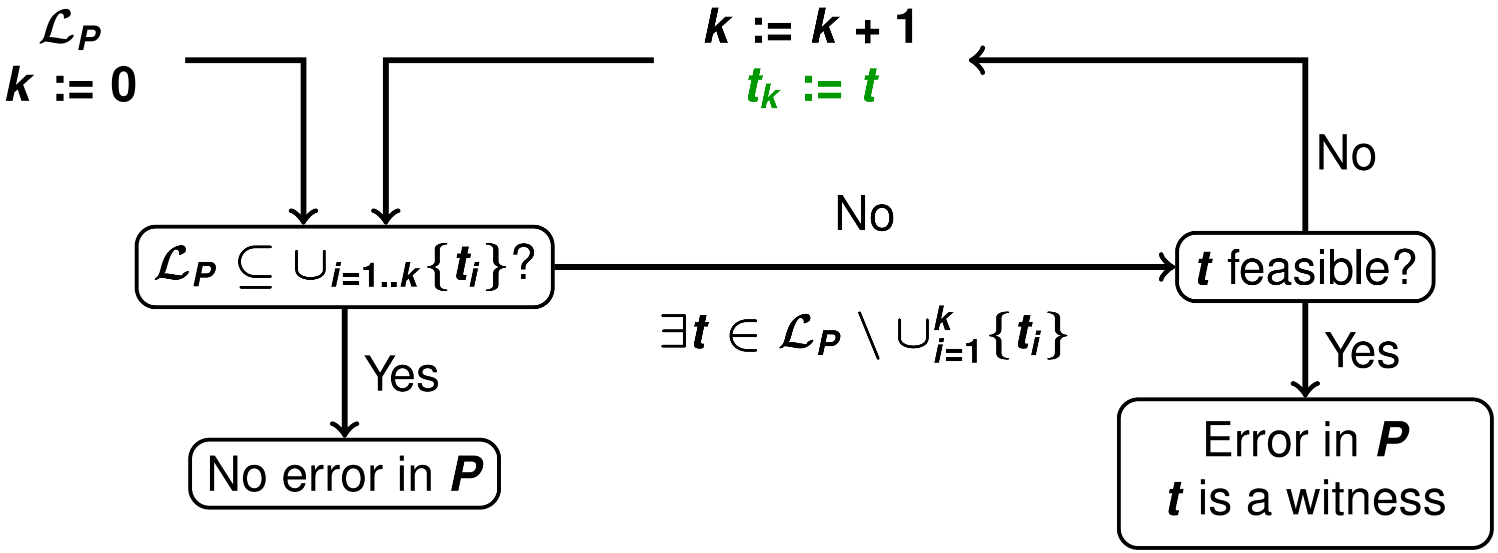
```
2   // thread T1
3   thread T1
4   x = 0;
5   lock(m);
6   if (x == y) {
7     unlock(m);
8     d = 3;
9   } else {
10    unlock(m);
11  }
12  /* end */
```



```
14  // Thread T2
15  thread T2
16  y = 1;
17  lock(m);
18  if (x <= y) {
19    unlock(m);
20    d = 2;
21  } else {
22    unlock(m);
23  }
```

$\mathcal{L}(P(T_1) \times P(T_2))$ **Is there a *feasible* error trace?**

Trace abstraction refinement (1)



$\mathcal{L}_P$

$k := 0$

$k := k + 1$

$t_k := t$

$\mathcal{L}_P \subseteq \cup_{i=1..k}\{t_i\}$?

No

$\exists t \in \mathcal{L}_P \setminus \cup_{i=1}^{k}\{t_i\}$

t feasible?

No

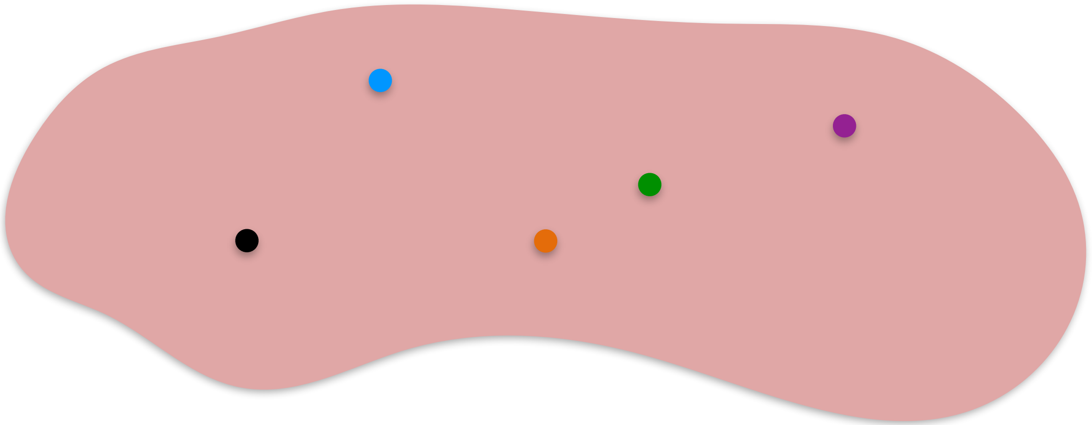
Yes

No error in P

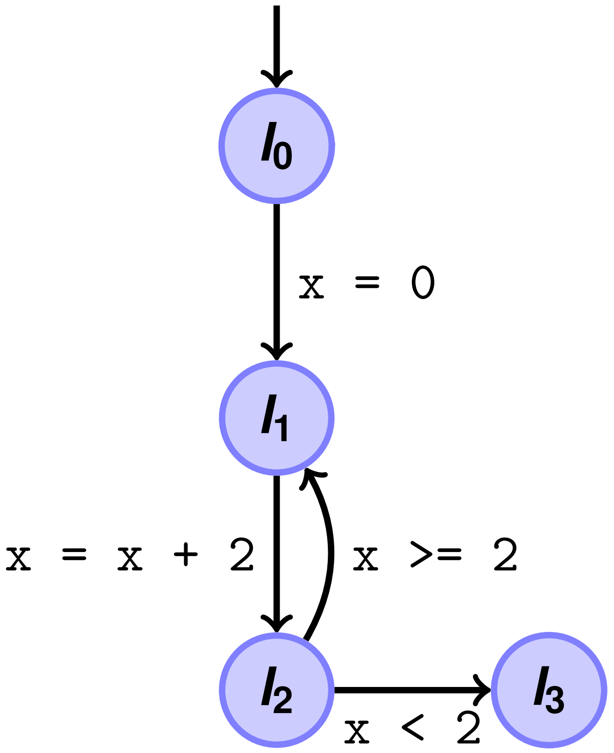
Yes

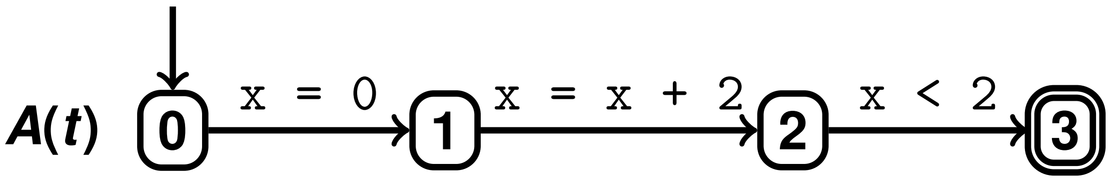
Error in P

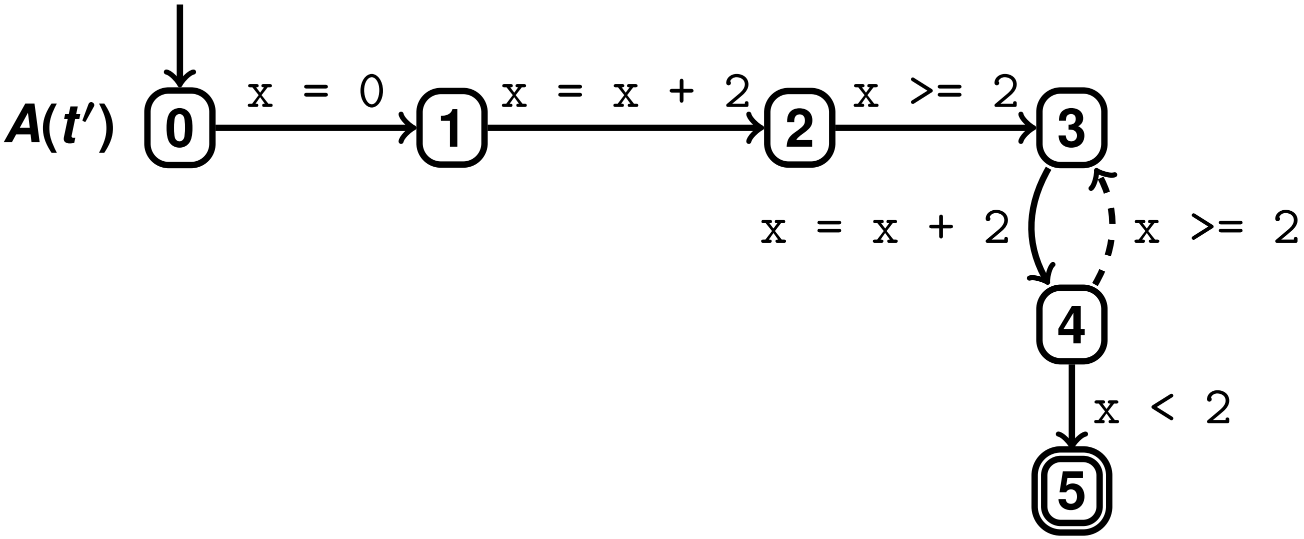
t is a witness



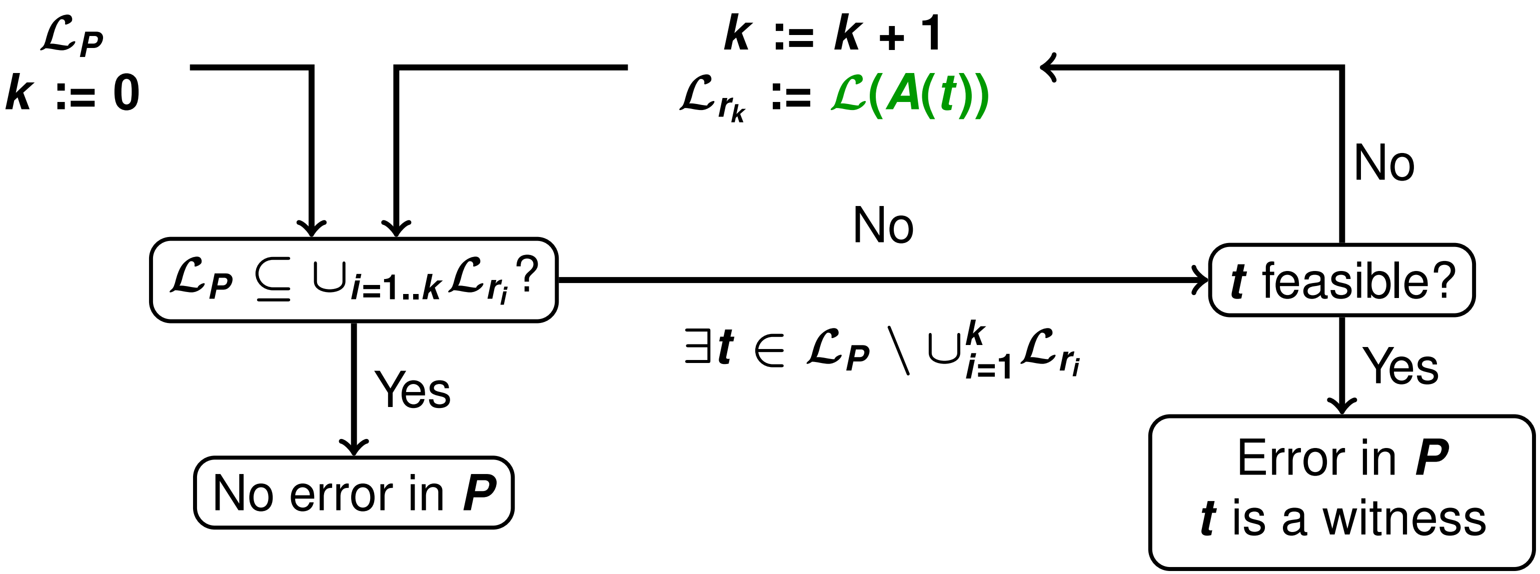
From one to many infeasible traces







Trace abstraction refinement (2)



$\mathcal{L}_P$
$k := 0$

$k := k + 1$
$\mathcal{L}_{r_k} := \mathcal{L}(A(t))$

$\mathcal{L}_P \subseteq \cup_{i=1..k} \mathcal{L}_{r_i}$?

No

$\exists t \in \mathcal{L}_P \setminus \cup_{i=1}^{k} \mathcal{L}_{r_i}$

t feasible?

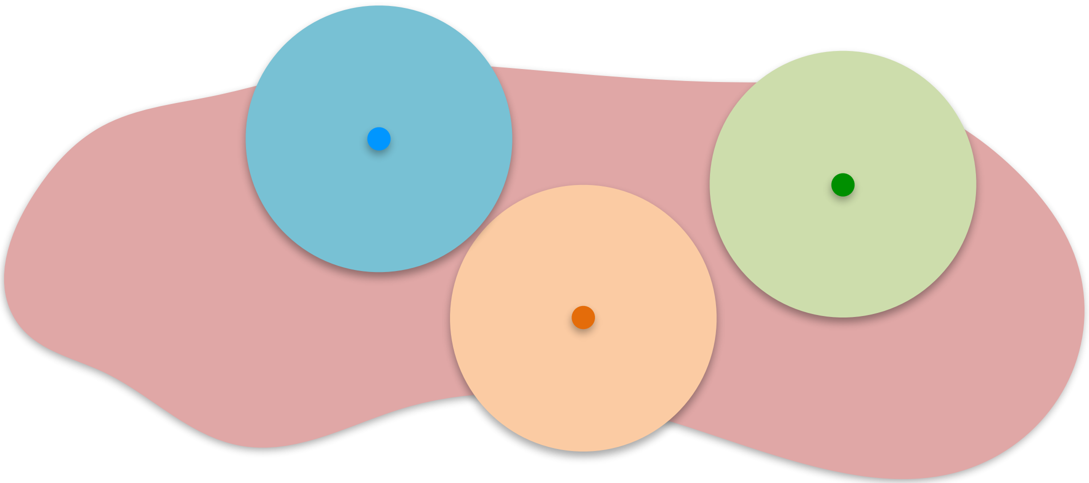
No

Yes

No error in P

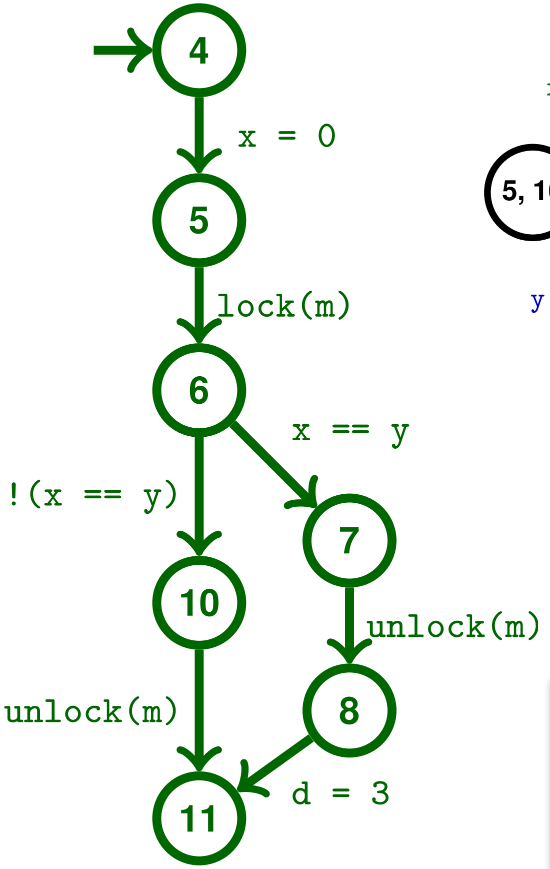
Yes

Error in P
t is a witness

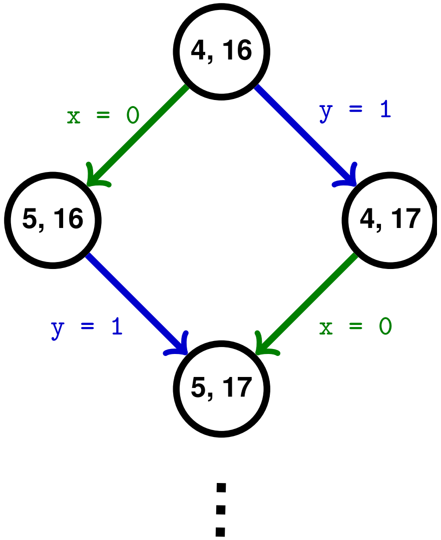


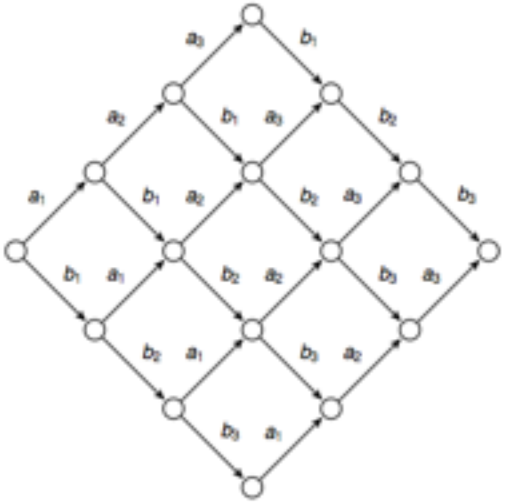
Heizmann, M., Hoenicke, J., Podelski, A.
Refinement of trace abstraction.
Static Analysis Symposium, 2009.

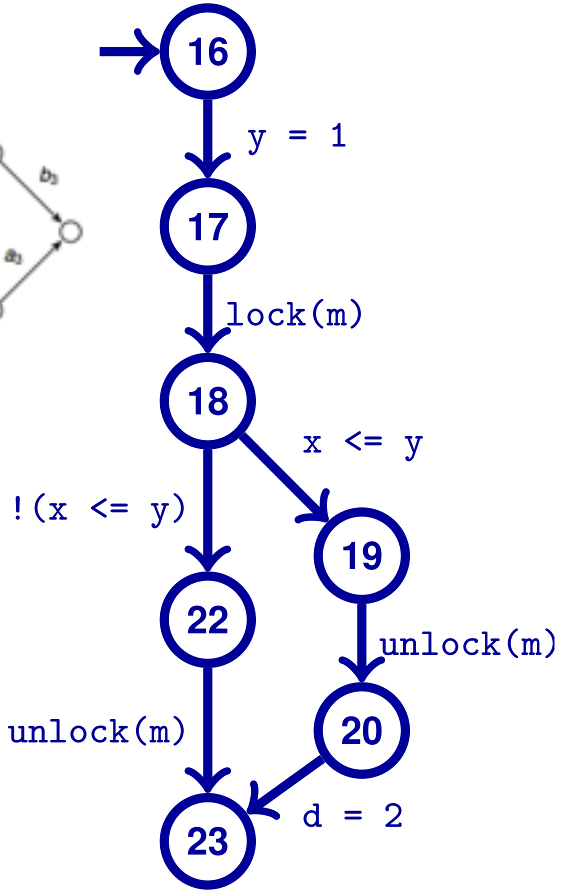
Partial order reductions



$P(T_1)$







$P(T_2)$

$\exists$an error trace $t \in \mathcal{L}(P)$

$$\Longleftrightarrow$$

$\exists$an error trace $t' \in \mathcal{L}_R(P)$

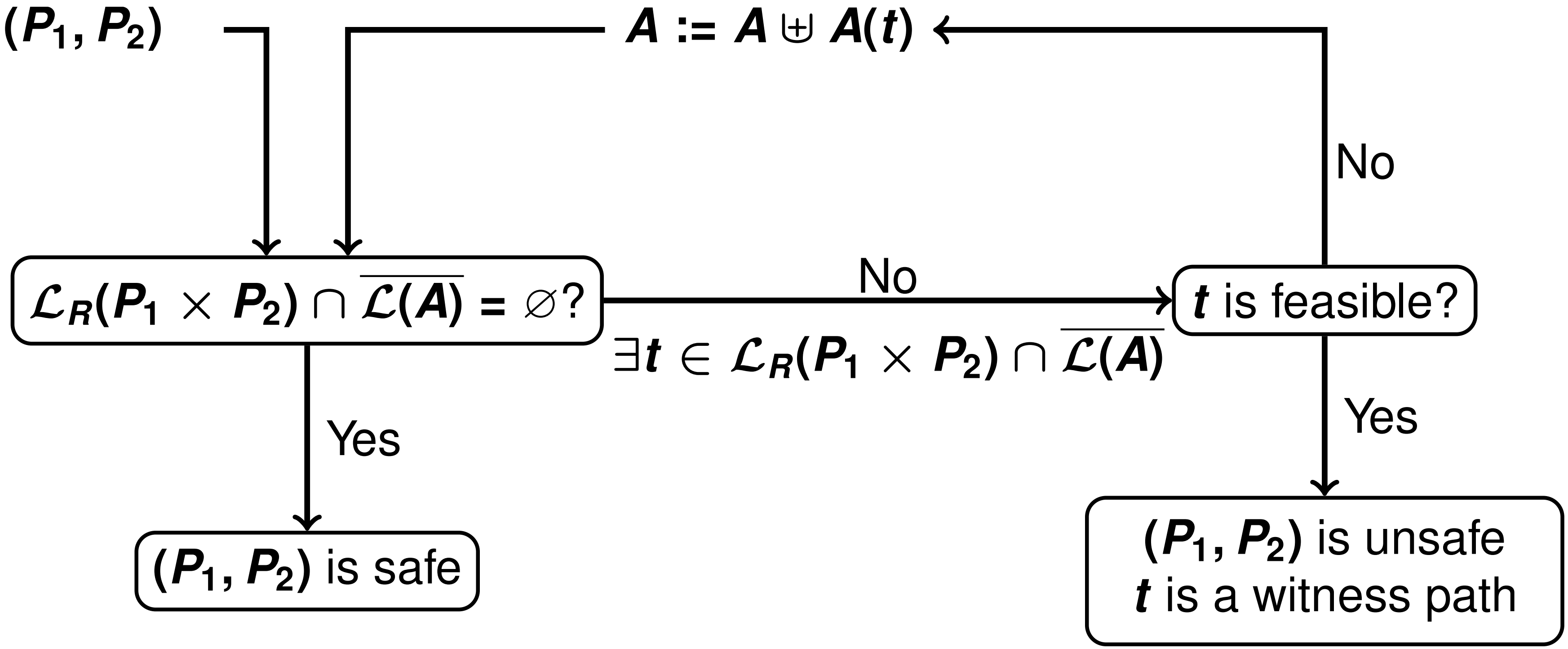
Main result

feasible

$\exists$an error trace $t \in \mathcal{L}(P)$

$\iff$

$\exists$an error trace $t' \in \mathcal{L}_R(P)$


(P_1, P_2)
$A := A \uplus A(t)$
No
$\mathcal{L}_R(P_1 \times P_2) \cap \overline{\mathcal{L}(A)} = \varnothing$?
No
$\exists t \in \mathcal{L}_R(P_1 \times P_2) \cap \overline{\mathcal{L}(A)}$
t is feasible?
Yes
Yes
(P_1, P_2) is safe
(P_1, P_2) is unsafe
t is a witness path

Experiments

Some SVCOMP-2015 benchmarks

Program	Safe	Ref	Sta.	Red.	LOC	#T	#V	Raptor	Mu-CSeq	Threader	Impara
stateful01	no	0	22	0%	34	3	6	1.1s/20	0.9s/1027	0.6s	N/A
stateful01	yes	10	1628	17%	34	3	6	6.1s	TO	2.6s	N/A
lazy01	no	1	11	0%	22	3	2	1.3s/9	0.6s/641	4.1s	0.16s
peterson	yes	29	1200	8%	31	2	4	5.7s	TO	4.6s	0.5s
dekker	yes	9	1276	7%	46	2	4	6.6s	TO	3.3s	0.7s
szymanski	yes	47	9811	13%	59	2	3	10s	TO	12s	1.43s
read_write_lock	no	11	2178	16%	65	4	5	6s/26	0.9s/992	55s	3.9s
read_write_lock	yes	38	10216	24%	63	4	5	9.5s	TO	57s	15s
time_var_mutex	yes	5	67	38%	33	2	5	0.69s	TO	4.9s	0.2s
fib_bench_false	no	284	10082	77%	25	3	2	29s/37	3.58s/949	TO	TO
ext-spin2003	yes	1	203	0%	44	4	2	3.4s	TO	176s	5.5s

Related work

Wachter, B., Kroening, D., Ouaknine, J.
Verifying multi-threaded software with impact.
In: FMCAD, IEEE (2013) 210–217

CEGAR + POR

Farzan, A., Kincaid, Z., Podelski, A.
Inductive data flow graphs.
POPL, ACM (2013) 129–142

TAR + iDFG

Gupta, A., Popeea, C., Rybalchenko, A.
Predicate abstraction and refinement for verifying multi-threaded programs.
POPL, ACM (2011), 331–344

Conclusion

Partial order reduction + trace abstraction refinement ☑

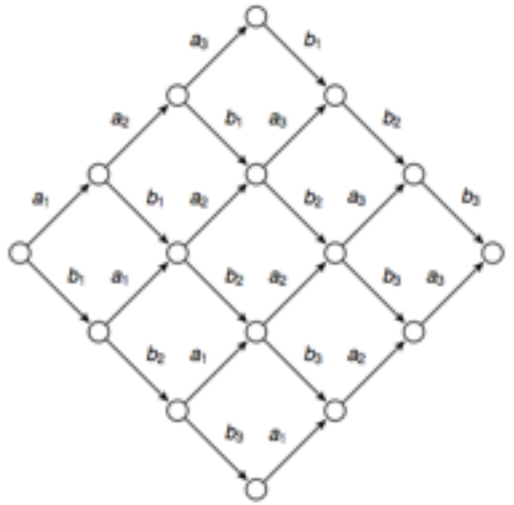
Avantages

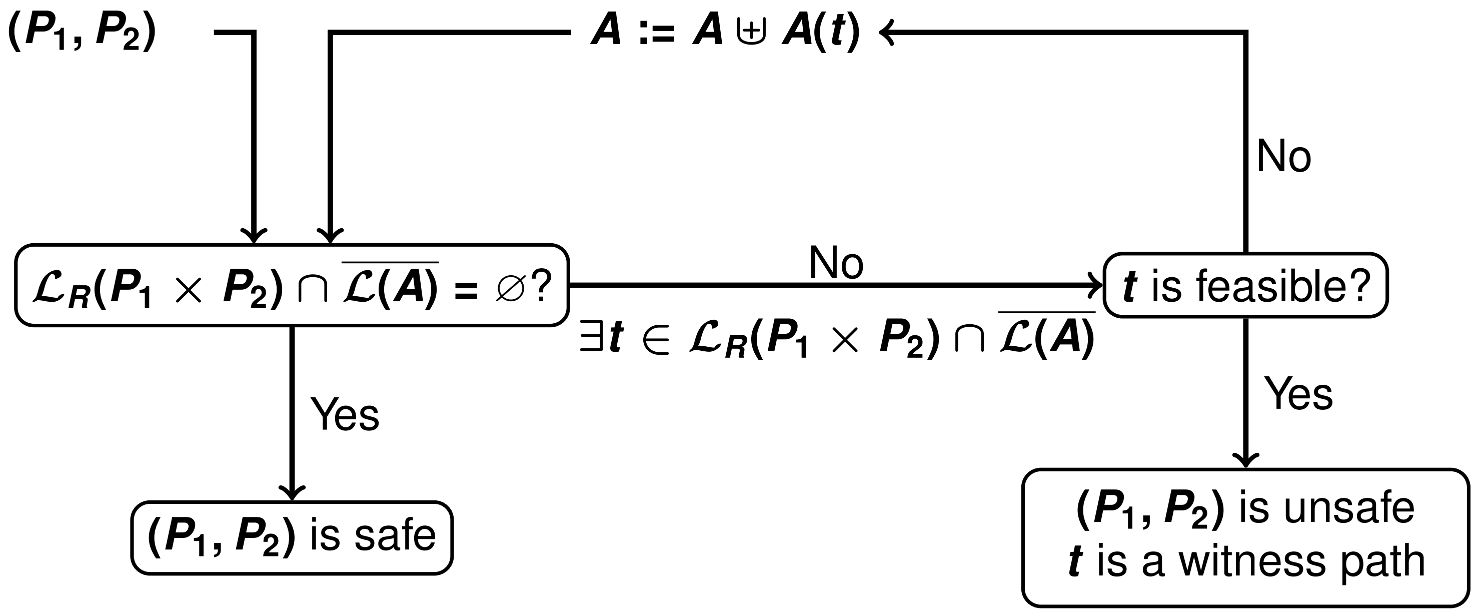
- absence/presence of bug vs. bug finding only
- modular, extends to other reduction (symmetry)
- simple

Ongoing

- even more infeasible traces ☑
- inter-procedural version □
- parse C programs (clang/LLVM) ☑

Appendix





$(\boldsymbol{P_1}, \boldsymbol{P_2})$

$\boldsymbol{A} := \boldsymbol{A} \uplus \boldsymbol{A}(\boldsymbol{t})$

No

$\mathcal{L}_R(\boldsymbol{P_1} \times \boldsymbol{P_2}) \cap \overline{\mathcal{L}(\boldsymbol{A})} = \varnothing$?

No

$\boldsymbol{t}$ is feasible?

$\exists \boldsymbol{t} \in \mathcal{L}_R(\boldsymbol{P_1} \times \boldsymbol{P_2}) \cap \overline{\mathcal{L}(\boldsymbol{A})}$

Yes

Yes

$(\boldsymbol{P_1}, \boldsymbol{P_2})$ is safe

$(\boldsymbol{P_1}, \boldsymbol{P_2})$ is unsafe
$\boldsymbol{t}$ is a witness path