BACH： Path-oriented Reachability Checker of Linear Hybrid Automata

Xuandong Li

Department of Computer Science and Technology,

Nanjing University, P.R.China

Outline

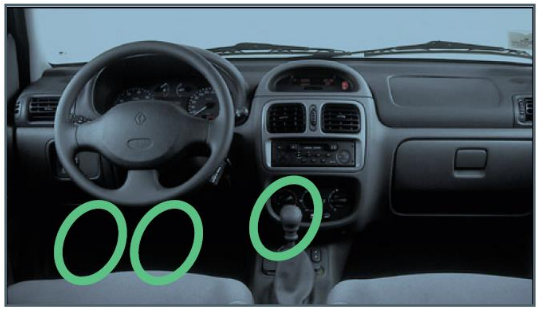
- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

Outline

- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

Hybrid System

- Systems containing both discrete and continuous components
- Practical Examples:
  - Embedded System Controller
  - VLSI circuits
  - System Biology
- Safety Critical Area
- Formal Verification
  - Formal Model : Hybrid Automata

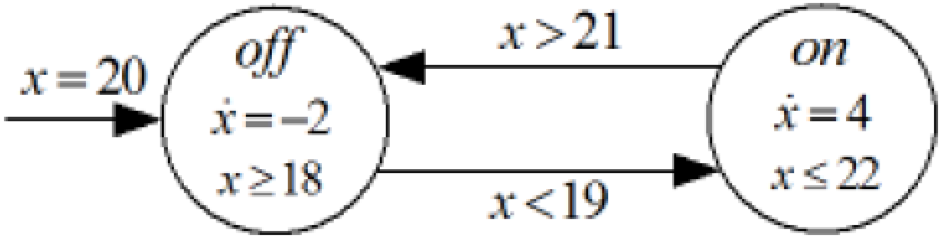


Hybrid Automata

- Widely studied formal models for hybrid systems

$$H = (X, \Sigma, V, E, V^0, \alpha, \beta, \gamma)$$

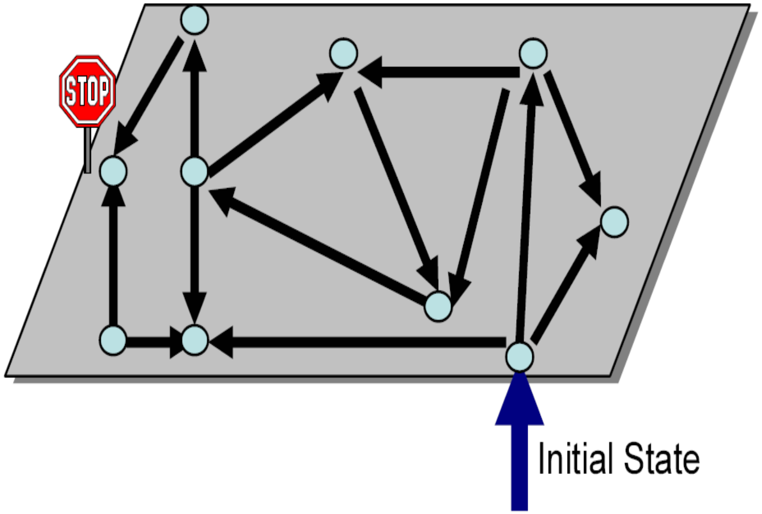
- They consist of
  - A finite state transition system
  - Differential equations in each location
- Example

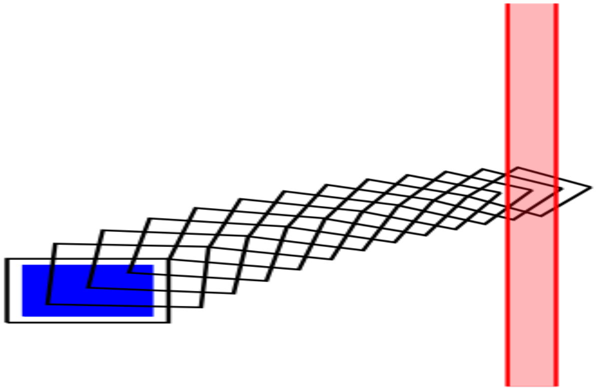


- Linear Hybrid Automata

Reachability Analysis

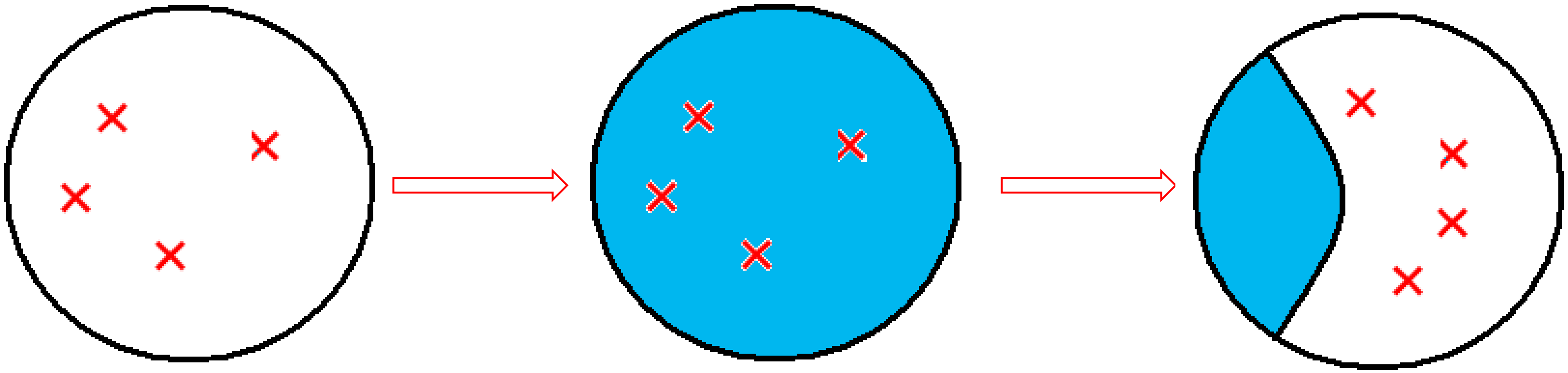
- Approach
  - Over-approximation
  - Geometric Computation
- Performance
  - Undecidable
  - Imprecise
  - Low dimension





Reachability Analysis

- Bounded Model Checking
  - Search for a potential behavior within *k* step
  - Usually solved by SMT techniques
    - SMT： satisfiability modulo theories
  - Need to encode all the potential bounded behavior firstly
  - Medium bound —> Large SMT problem



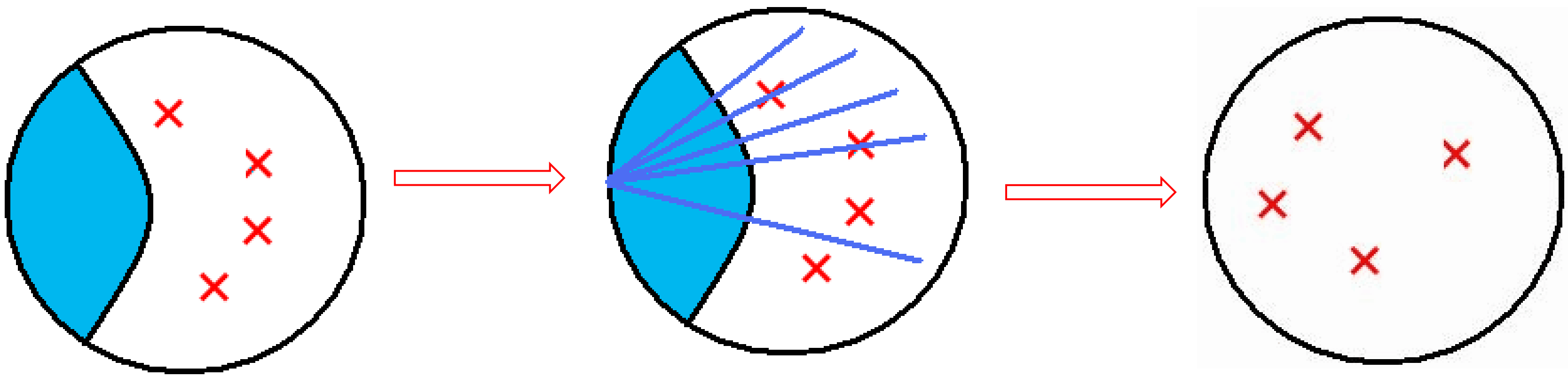
Control The Complexity!

Outline

- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

Reachability Analysis

- Path-oriented Based Bounded Model Checking
  - Check the reachability of one abstract path using Linear Programming （LP）
  - Enumerate all the candidate paths in bound by Depth First Search （DFS）

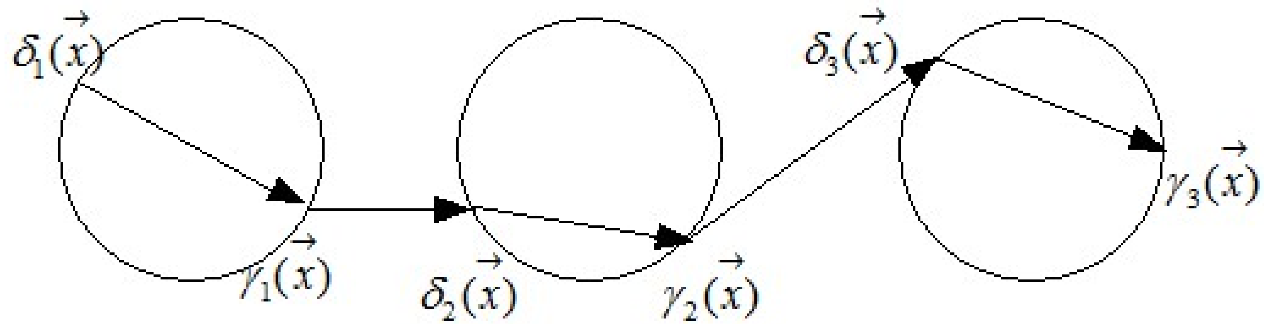


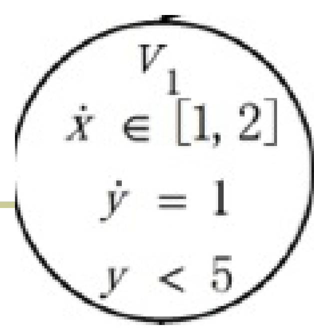
Path, Behavior, Encoding

- Path: $\langle v_0 \rangle \xrightarrow[\sigma_0]{(\varphi_0,\psi_0)} \langle v_1 \rangle \xrightarrow[\sigma_1]{(\varphi_1,\psi_1)} \cdots \xrightarrow[\sigma_{n-1}]{(\varphi_{n-1},\psi_{n-1})} \langle v_n \rangle$

- Behavior: $\left\langle \begin{matrix} v_0 \\ t_0 \end{matrix} \right\rangle \xrightarrow[\sigma_0]{(\varphi_0,\psi_0)} \left\langle \begin{matrix} v_1 \\ t_1 \end{matrix} \right\rangle \xrightarrow[\sigma_1]{(\varphi_1,\psi_1)} \cdots \xrightarrow[\sigma_{n-1}]{(\varphi_{n-1},\psi_{n-1})} \left\langle \begin{matrix} v_n \\ t_n \end{matrix} \right\rangle$

- Encoding

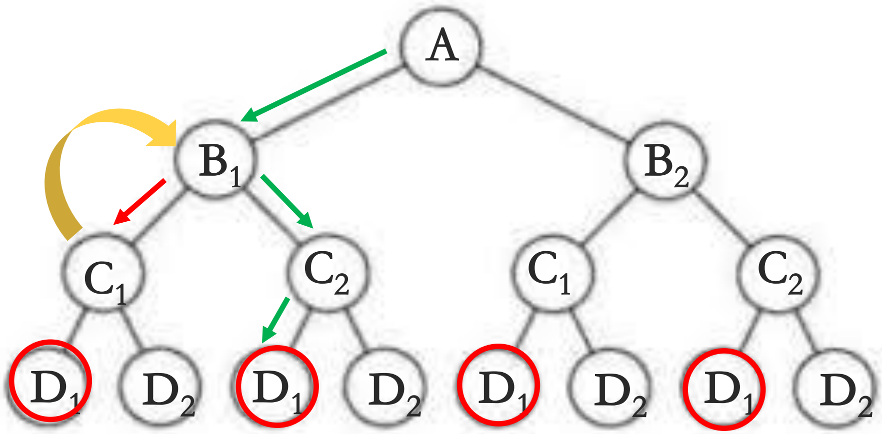




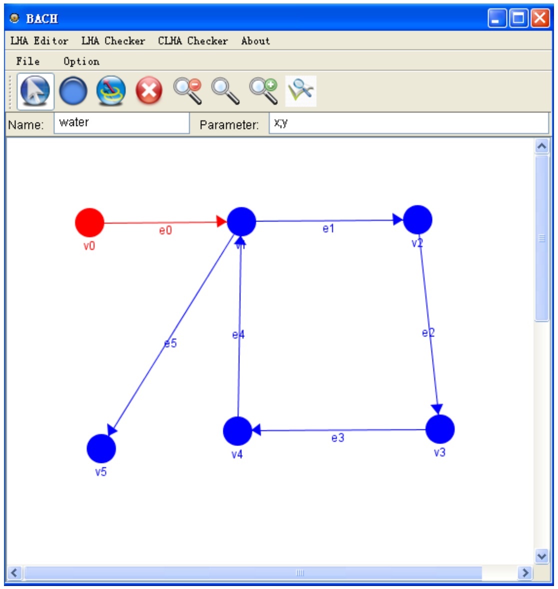
$$g_1(y) = d_1(y) + t_1$$
$$d_1(x) + t_1 \leq g_1(x) \leq d_1(x) + 2t_1$$
$$t_1 \geq 0; d_1(y) < 5; g_1(y) < 5$$

DFS-Based Bounded Model Checking

- Eager-DFS-BMC
  - check each path ρ in the given bound
  - If ρ is infeasible, backtrack to the last location



- BACH: ***B***ounded re***A***chability ***CH***ecker



- http://seg.nju.edu.cn/BACH/

Outline

- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

Eager - DFS - BMC

- Eager - DFS - BMC
  - Check each path ρ in the given bound
  - Lots of redundant work

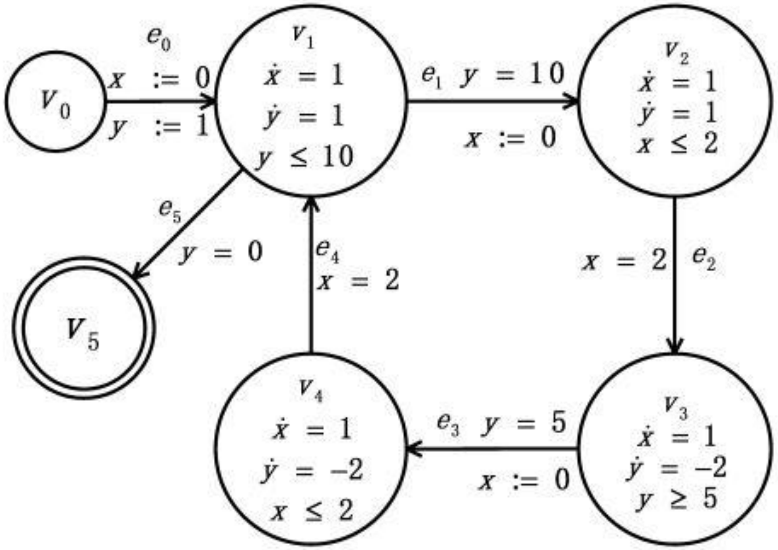
- Example
  - Target v_5

$v_0 \rightarrow v_1$

$v_0 \rightarrow v_1 \rightarrow v_2$

$v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow v_3$

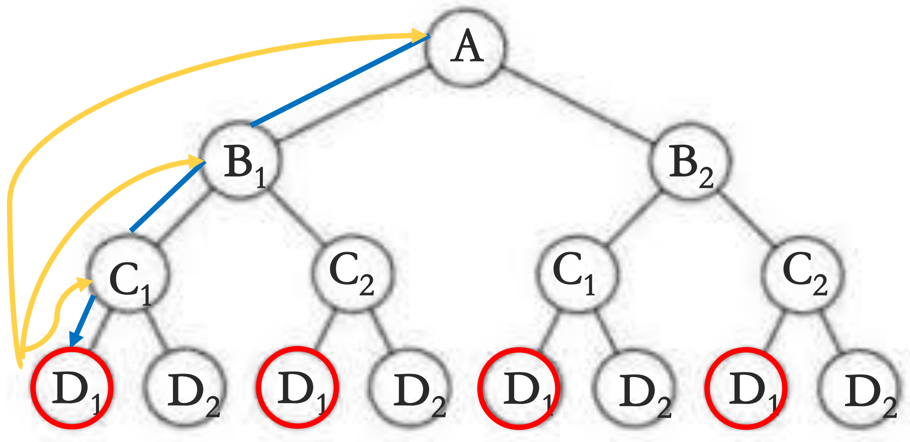
∘ ∘ ∘



Most of the time are spent in LP solving

Lazy DFS + LP

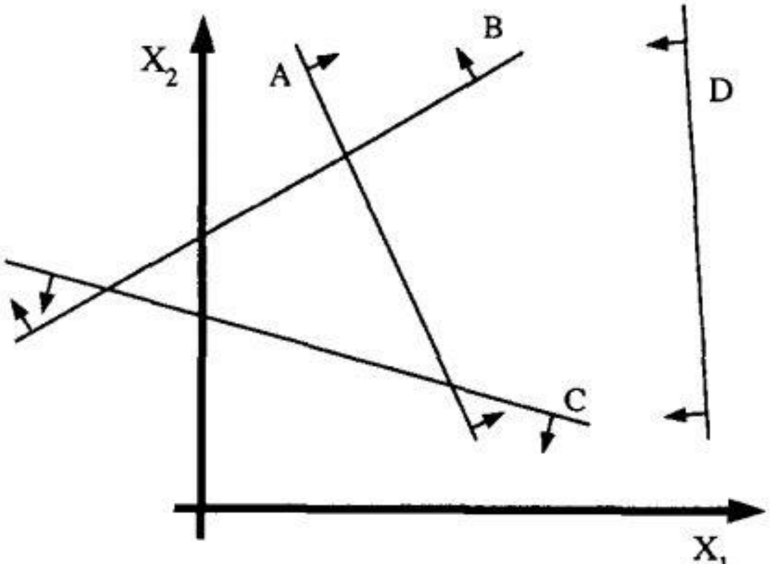
- Lazy DFS + LP
  - Only check the path ρ when it reaches the target



Where to backtrack？

IIS

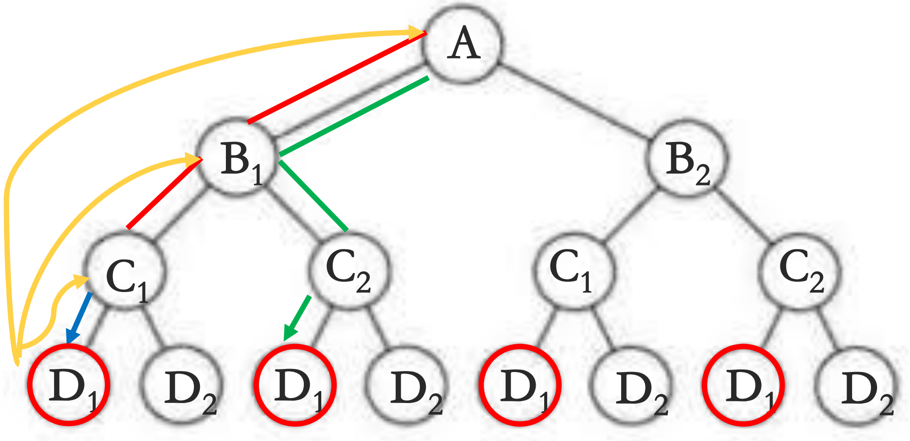
- Using IIS to locate infeasible path segment core to accelerate the backtracking
- An irreducible infeasible set (IIS) of an infeasible linear constraint set is an unsatisfiable set of constraints that becomes satisfiable if any constraint is removed



{A,B,C} is an IIS

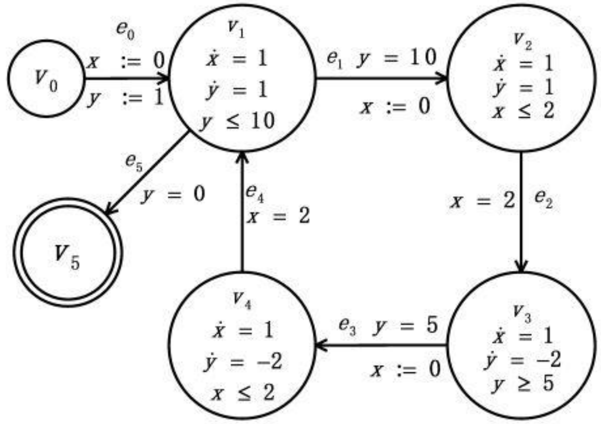
Extract the infeasible path segment

- ***Recall:*** We use an LP based approach to check the feasibility of a path ρ
- IIS technique can be used to locate the minimal inconsistent set
- Such inconsistent set can be mapped back to an path segment. All the paths containing such path segments are not feasible for sure.



Example

- Example
  - $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$
  - $v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$ is the IIS path segme
  - Backtrack to v_1
  - Once DFS found a new path containing $v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$ it will backtrack to v_1 directly without call LP solver



Bound 100, Lazy DFS+IIS ->25 paths only call LP solver 2 times

Problem:

These paths containing the IIS are not feasible for sure.

Can we don't waste time in enumerating such paths?

SAT-LP-IIS

- **The transition relation graph can be encoded as propositional formulas**
  - Encode the bounded graph structure of an LHA into a propositional formula set
  - Find a truth assignment using a SAT solver
    - SAT： Boolean satisfiability problem
  - Decode the truth assignment to get a path in the graph

SAT Encoding of the Bounded Graph

- Consist of four clauses

$$NEXT := \bigwedge_{q \in V} (loc = q \rightarrow \bigvee_{(q,q') \in N} loc' = q')$$

$$EXCLUDE := \bigwedge_{q \in V} (loc = q \rightarrow \bigwedge_{q' \in V \wedge q' \neq q} loc \neq q')$$

$$INIT := (loc = v_I) \wedge EXCLUDE$$

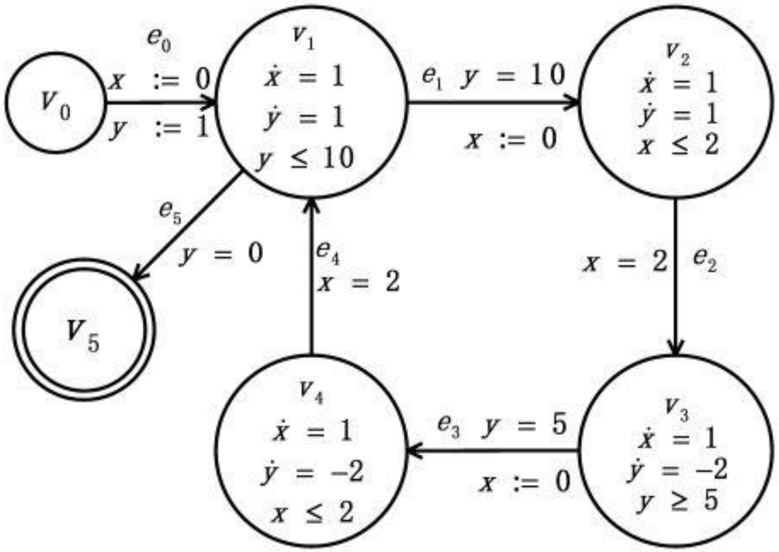
$$TARGET := (loc = v_T)$$

- The bounded graph formula set with bound k

$$BG^k := INIT^0 \wedge \bigwedge_{0 \leq i \leq k-1} NEXT^i \wedge \bigwedge_{1 \leq i \leq k} EXCLUDE^i \wedge (\bigvee_{0 \leq i \leq k} TARGET^i)$$

Decode From The Truth Assignment

- The superscript of the name of variables represents the order of the nodes in the path
- Suppose we get a truth valuation: v_0^0, v_1^1, v_5^2 from the SAT encoding, the corresponding path in the graph is $\langle v_0 \rangle \xrightarrow[e_0]{} \langle v_1 \rangle \xrightarrow[e_5]{} \langle v_5 \rangle$



Accelerating SAT-Based Enumeration by IIS

- Include a *IIS* clause to prevent the SAT from enumerating paths which contain an infeasible path segment.

$$IIS := \bigwedge_{\rho' \in IIS\ Path} IIS^k(\rho')$$

$$BG^k := BG^k \wedge IIS$$

Example

- The previous checked path

$$\rho = \langle v_0 \rangle \xrightarrow[e_0]{} \langle v_1 \rangle \xrightarrow[e_1]{} \langle v_2 \rangle \xrightarrow[e_2]{} \langle v_3 \rangle \xrightarrow[e_3]{} \langle v_4 \rangle \xrightarrow[e_4]{} \langle v_1 \rangle \xrightarrow[e_5]{} \langle v_5 \rangle$$

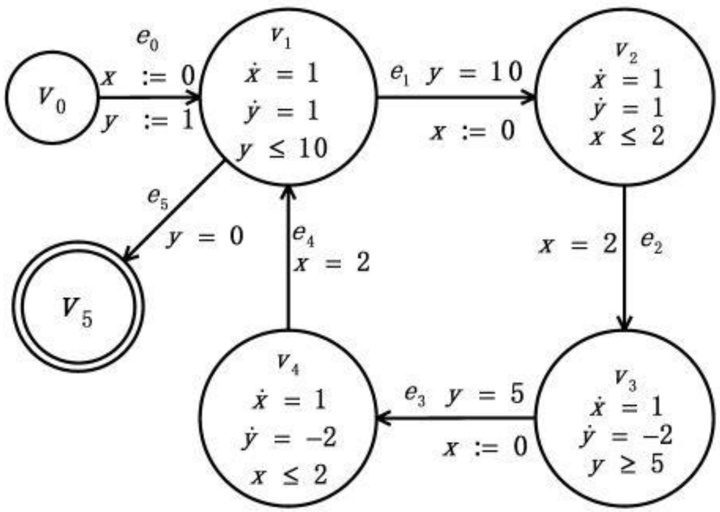
- The infeasible path segment

$$\rho' = \langle v_3 \rangle \xrightarrow[e_3]{} \langle v_4 \rangle \xrightarrow[e_4]{} \langle v_1 \rangle \xrightarrow[e_5]{} \langle v_5 \rangle$$

- The *IIS* clause

$$IIS^k(\rho') := \bigwedge_{0 \le i \le k - leng + 1} \left(v_3^i \wedge v_4^{i+1} \wedge v_1^{i+2} \rightarrow \neg v_5^{i+3} \right)$$

Example



Bound 100, v_5

DFS+IIS ->25 paths

(call LP 2 times)

SAT+IIS -> 2 paths

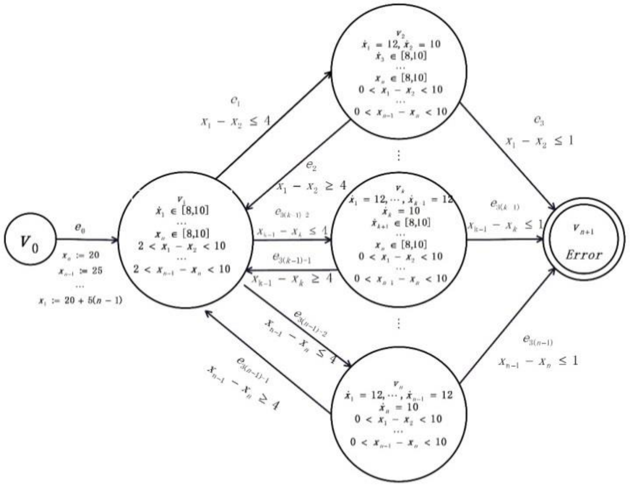
Performance

Performance Data On The Highway System With 500 Vehicles

System Size: 502 locations, 500 variables

Tech. / Bound	BACH-SAT		BACH-DFS		MathSAT		Z3	
	Time	Memory	Time	Memory	Time	Memory	Time	Memory
3	53.2s	<1000m	12.3s	<600m	OOM	>4096m	542.1s	2967m
100	62.2s	<2500m	OOT	<4096m	N/A	N/A	OOM	>4096m
200	74.2s	<4096m	N/A	N/A	N/A	N/A	N/A	N/A

- Large Scale System 500 locations， 500 variables
- Classical SMT-style BMC， OOM（Out of Memory） with bound 3
- BACH：
    - Path-oriented, complexity well controlled
    - With the help of IIS， 200 steps in only 74 seconds！



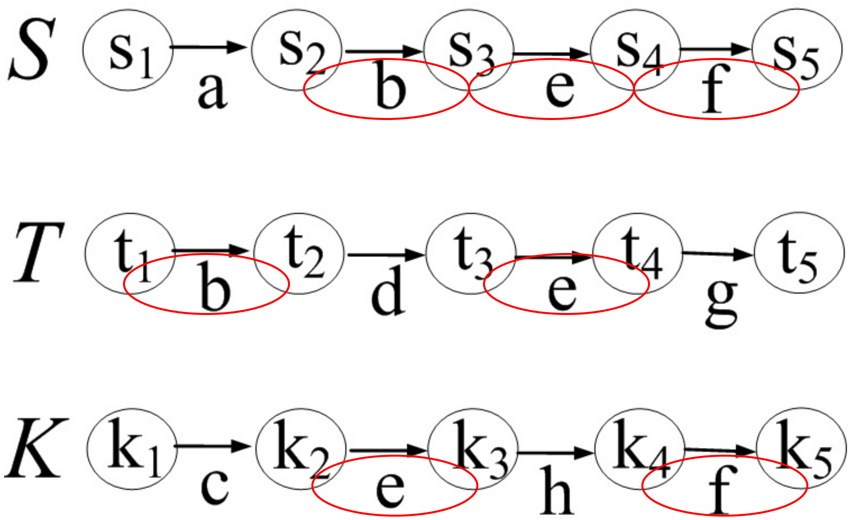
Scalable Highway System

Outline

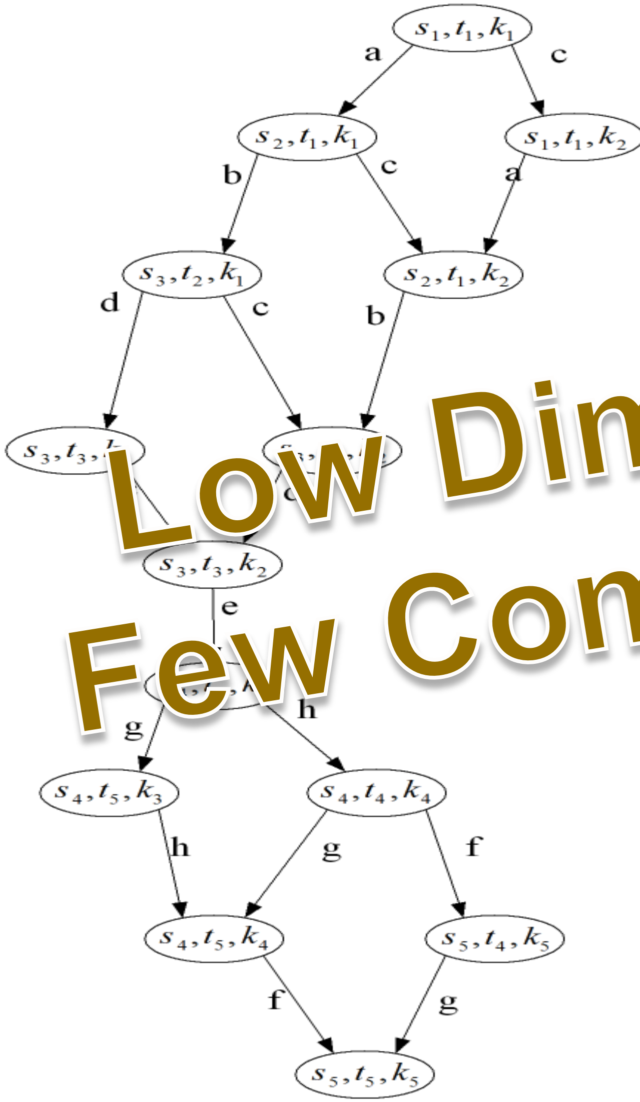
- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

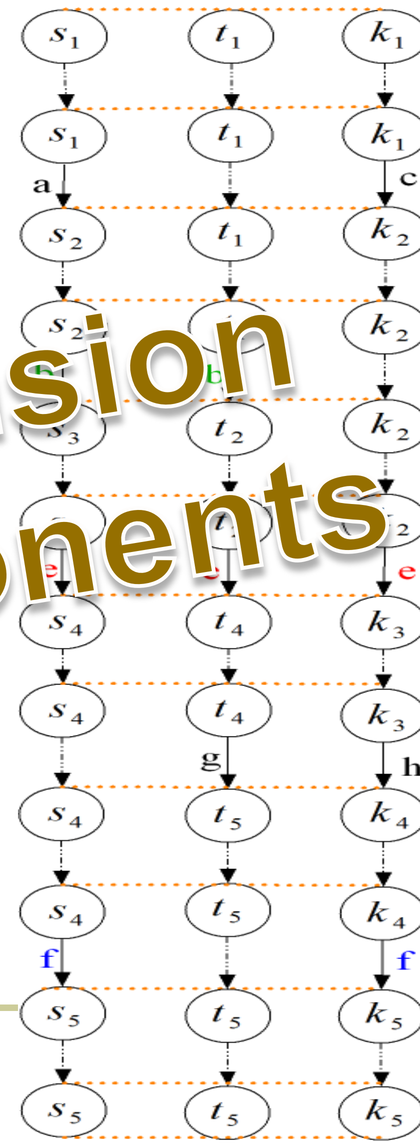
Compositional LHA System

- Compositional LHA Systems



Current Status

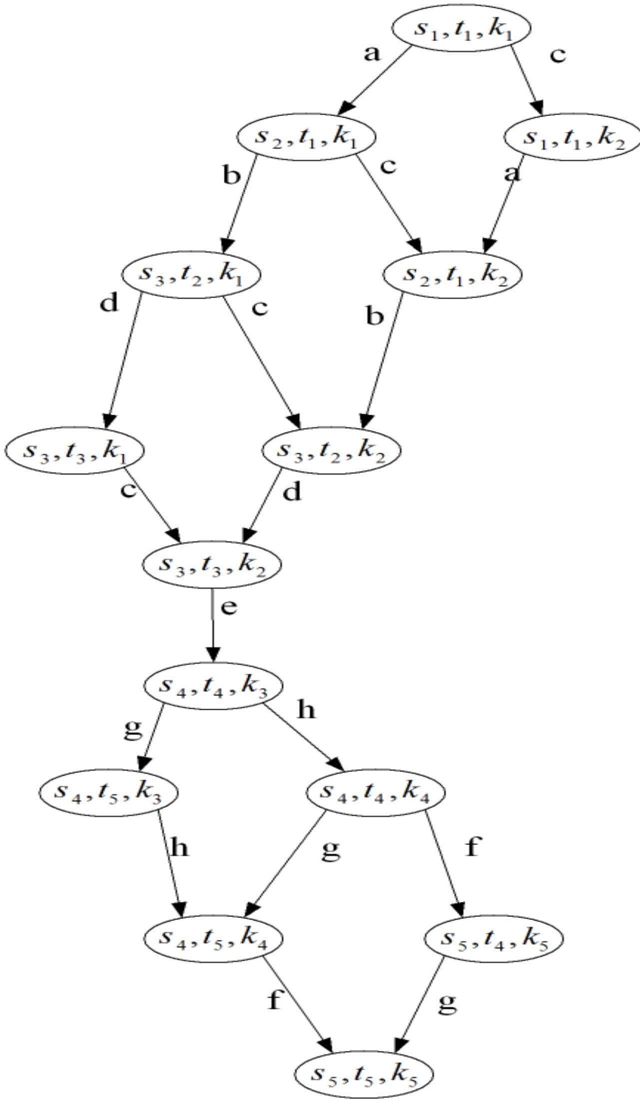


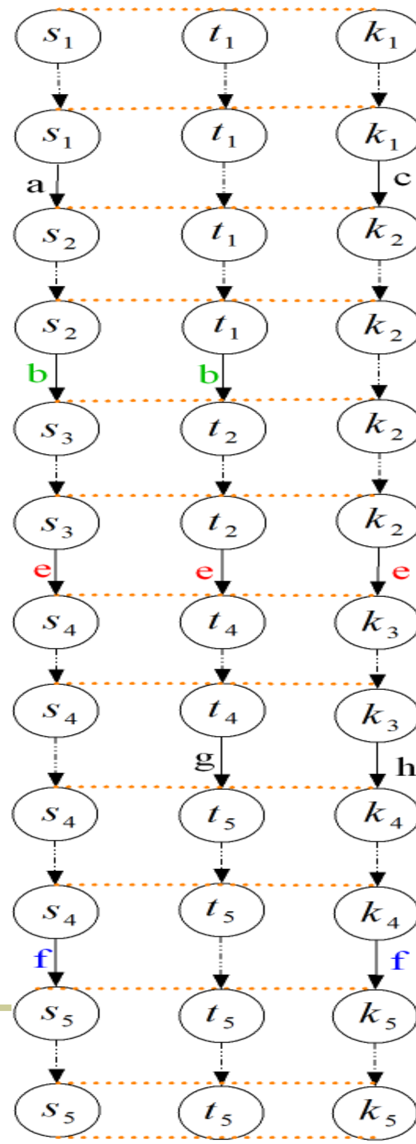


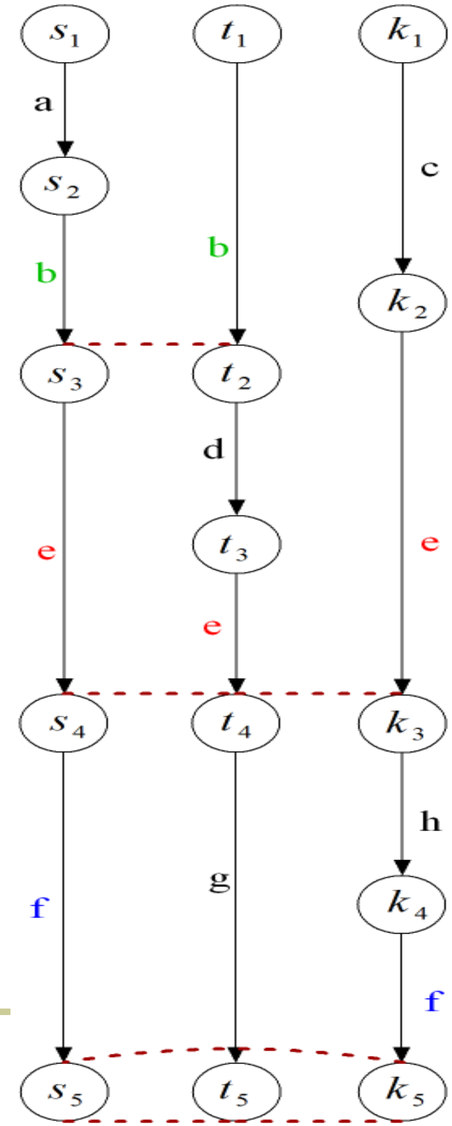
Low Dimension

Few Components

Shallow Synchronization Semantic







Bounded Reachability

- Find and verify all the path sets in the given bound limit

- Reduce the number of potential path sets which needs to be verified.

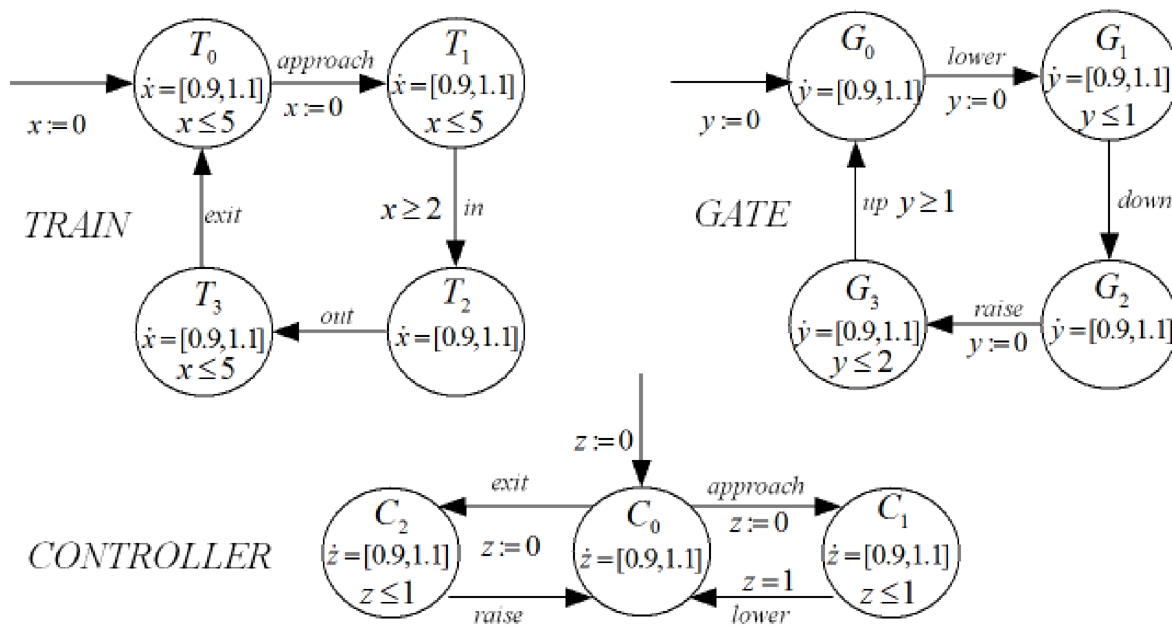
- Share label sequence guided DFS

Share Label Sequence Guided DFS

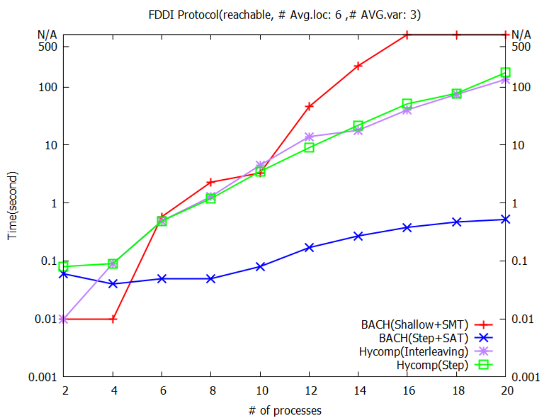
$$T_0 \xrightarrow{approach} T_1 \xrightarrow{in} T_2$$

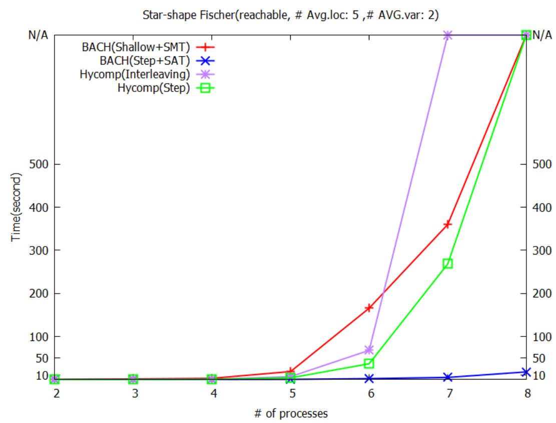
$$G_0 \xrightarrow{lower} G_1 \xrightarrow{down} G_2$$

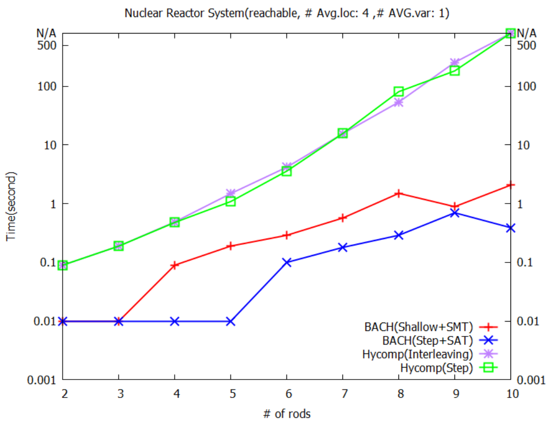
$$C_0 \xrightarrow{approach} C_1 \xrightarrow{lower} C_0 \xrightarrow{exit} C_2$$

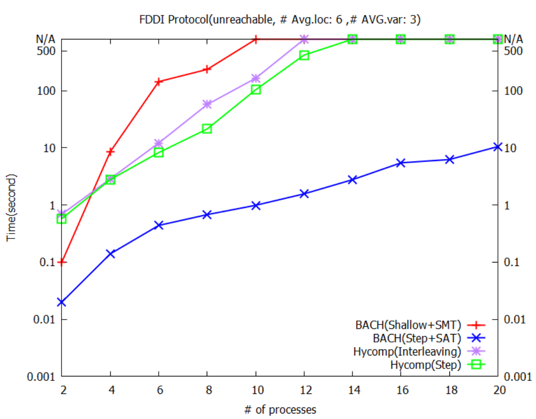


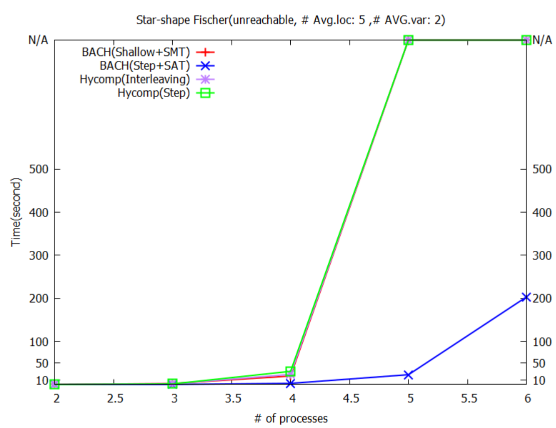
Performance

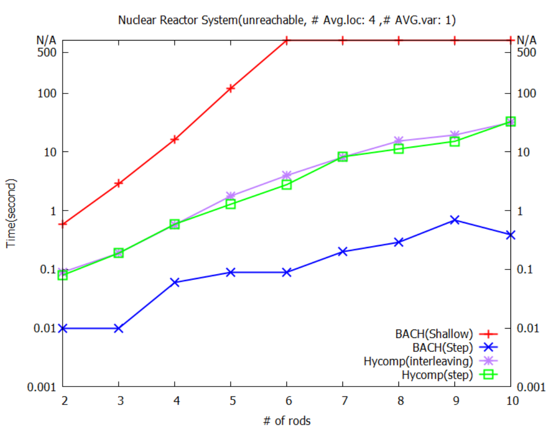








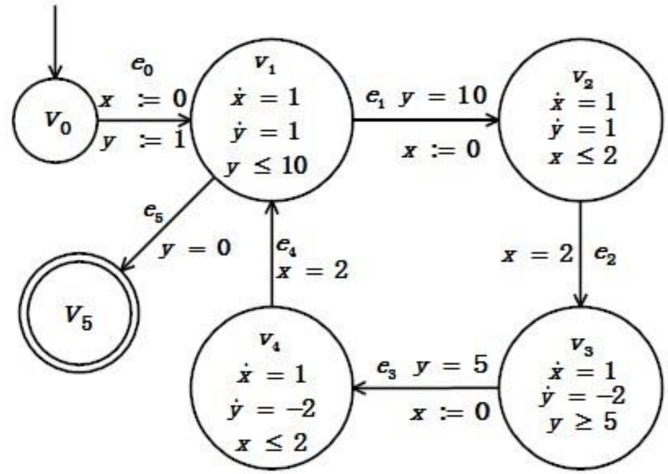




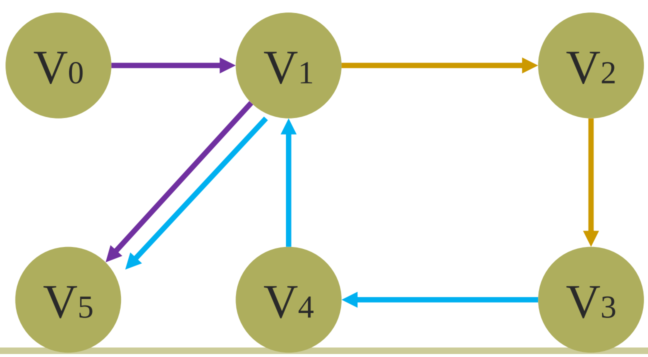
Outline

- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

Previous Example



Water-Level Monitor System



- Is v_5 reachable within 10 steps?

- Path: $v_0 \rightarrow v_1 \rightarrow v_5$
- IIS: $v_0 \rightarrow v_1 \rightarrow v_5$

- Path: $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$
- IIS: $v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$

Potential path can not contain

$v_3 \rightarrow v_4 \rightarrow v_1 \rightarrow v_5$

$v_0 \rightarrow v_1 \rightarrow v_5$

No more potential paths, not reachable!

Key insights

- Avoiding IIS path segments may make the target location not reachable in the unbounded state space

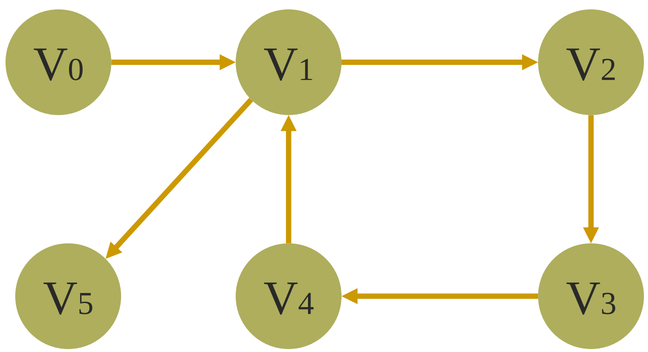
- Goal
  - Prove whether there exists a path which can reach the target location without touching certain path segments

- Solution
  - LTL model checking
    - LTL： linear temporal logic

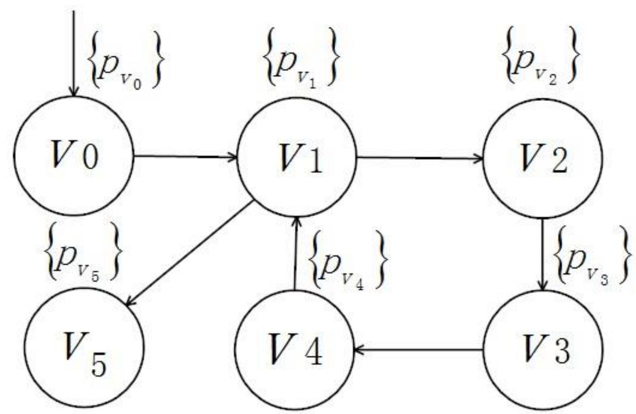
automatic and efficient

Model graph structure of LHA with TS

- We propose to model the graph structure of an LHA with a finite-state transition system (TS)



(A) Graph structure of Water-Level Monitor System



(B) TS Model for Water-Level Monitor System

Avoid containing an IIS path segment

- Suppose there is an IIS path segment:

$$\rho' = v_i \rightarrow v_{i+1} \rightarrow \ldots \rightarrow v_j$$

$$p_{v_i} p_{v_{i+1}} \cdots p_{v_j}$$

- The LTL formula which can represent ρ':

$$IIS_{\rho'} = p_{v_i} \& X\ p_{v_{i+1}} \& .. \& \underbrace{X\ X\ \ldots X}_{j-i} p_{v_j}$$

- A path which does not contain ρ':

$$G(\neg IIS_{\rho'})$$

Reach target without any IIS path segment

- The target location q_{bad} is finally reached:

$$v_i v_{i+1} \dots q_{bad}$$

$$p_{v_i} p_{v_{i+1}} \cdots p_{q_{bad}} \longrightarrow F\ p_{q_{bad}}$$

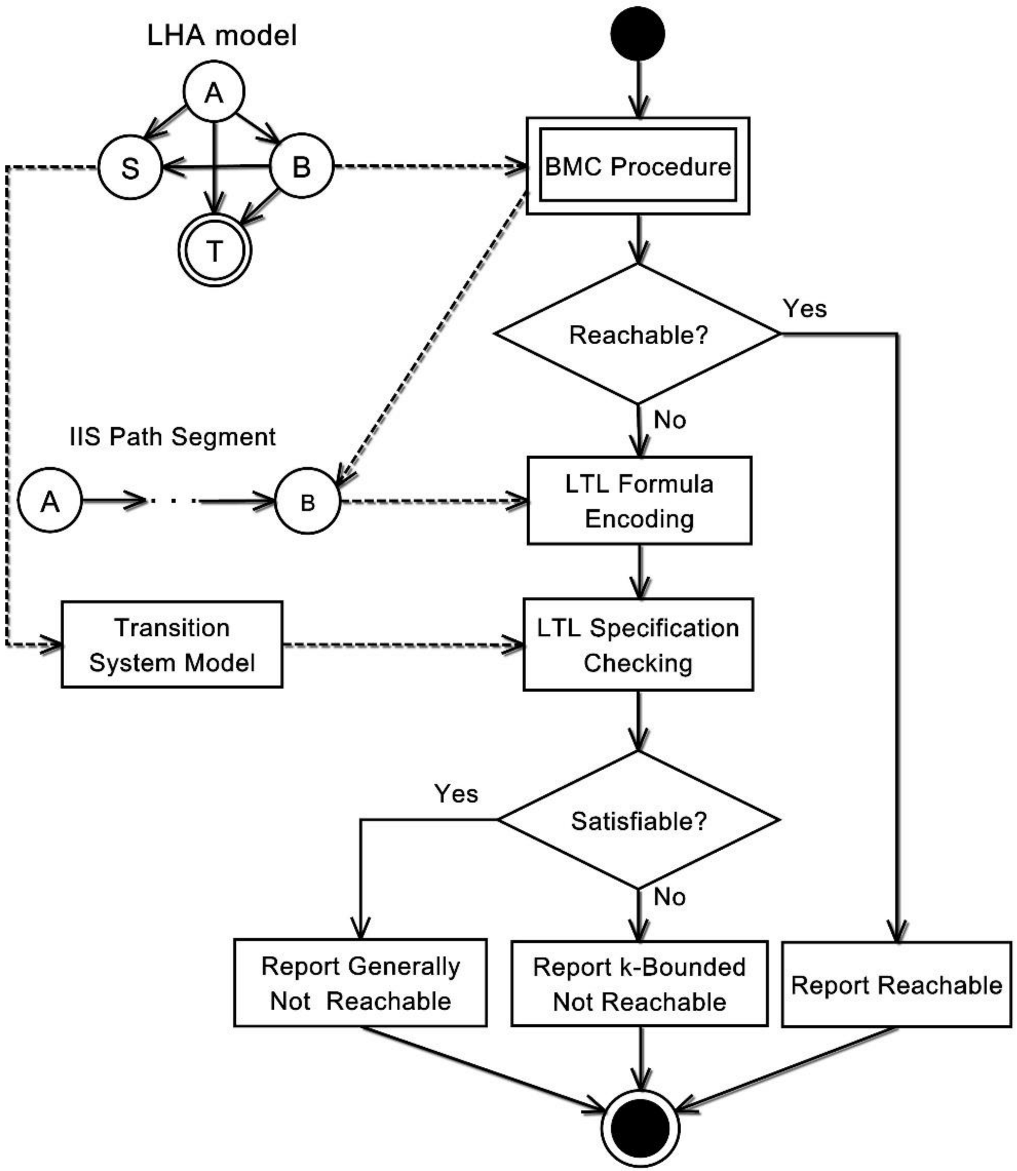
- The LTL formula which is true for path reaching the target without containing any IIS path segment

$$\{\rho_1, \rho_2, \dots \rho_n\} : (G(\bigwedge_{1 \le i \le n} \neg IIS_{\rho_i})) \wedge F\ p_{q_{bad}}$$

- As our target is to prove the nonexistence of such a path, the final LTL specification :

$$\neg((G(\bigwedge_{1 \le i \le n} \neg IIS_{\rho_i})) \wedge F\ p_{q_{bad}})$$

Workflow of Unbounded Proof Derivation



Experiment

System	#locs	#vars	BACH (NuSMV)			BACH (IC3)		SpaceEx (PHA.)		SpaceEx (Supp.)	
			#IIS	Time (s)	Mem. (MB)	Time (s)	Mem. (MB)	Time (s)	Mem. (MB)	Time (s)	Mem. (MB)
water	6	2	2	0.94_U	<1	0.87_U	30.4	**0.07**$_U$	<1	0.22_U	7.9
tcs	5	3	4	**0.97**$_U$	<1	0.98_U	16.4	T.O.	-	0.36_U	9.4
sample	8	2	9	0.96_B	26.8	0.41_B	21.2	**0.93**$_U$	<1	EXC	-
train	8	2	2	1.02_U	<1	**0.3**$_U$	<1	0.62_U	<1	1.24_U	24.8
motorcade_5	7	5	4	**0.05**$_U$	<1	0.4_U	<1	4.94_U	16	T.O.	-
motorcade_10	12	10	9	**0.12**$_U$	<1	0.6_U	16.9	T.O.	-	T.O.	-
motorcade_20	22	20	19	**0.53**$_U$	60.8	1.1_U	25.4	T.O.	-	T.O.	-
motorcade_100	102	100	99	**6.66**$_U$	163.9	15.7_U	389	T.O.	-	T.O.	-
motorcade_200	202	200	199	**61.8**$_U$	652.7	115.3_U	3299	T.O.	-	T.O.	-

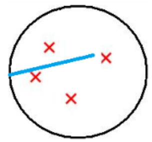
Try the task of unbounded proof by the cost of BMC!

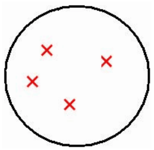
Outline

- Preliminary Knowledge
- Path-oriented Reachability Checking
- IIS-Based Bounded Checking
- Shallow Semantic Based Compositional Checking
- Unbounded Proof Derivation
- Conclusion

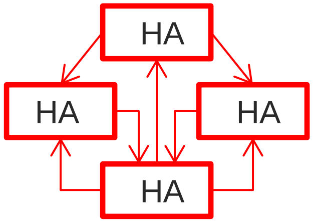
Framework

- Path oriented ⟹ Bounded

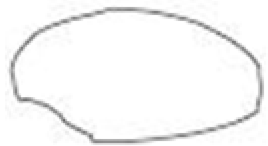




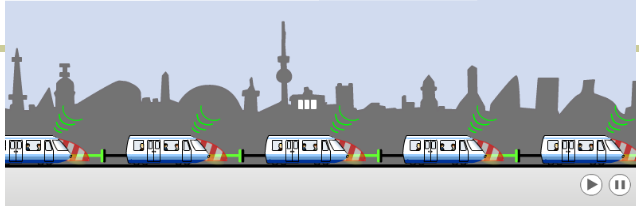
- Single HA ⟹ Compoesed HA



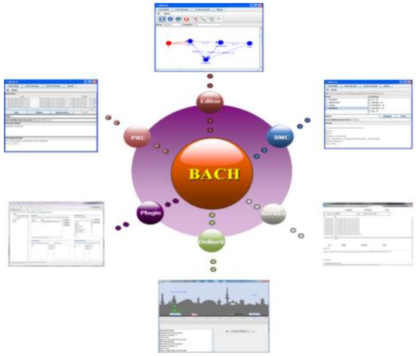
- Linear HA ⟹ Nonlinear HA



- Hybrid System ⟹ Cyber Physical



BACH



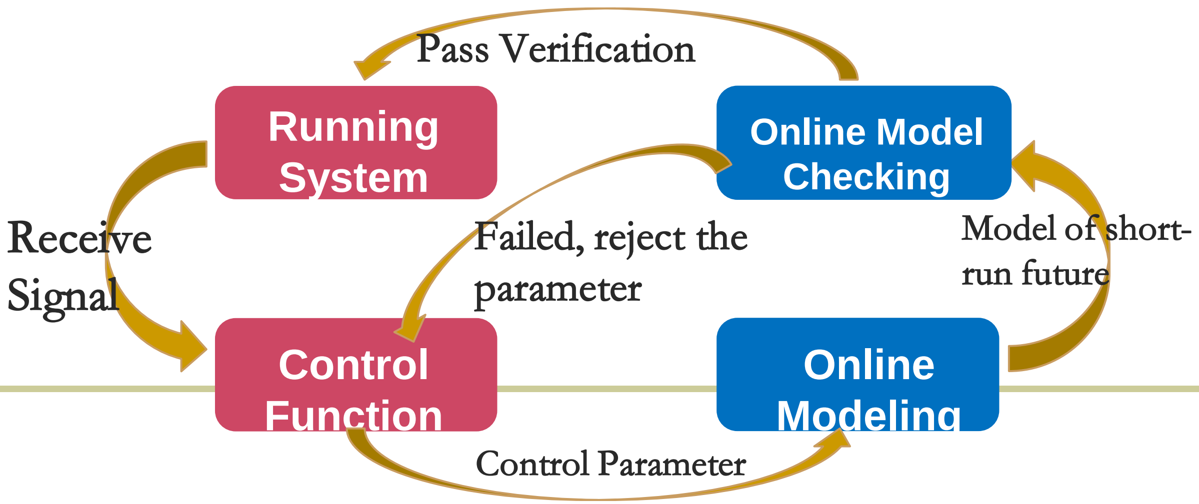
Current Achievement

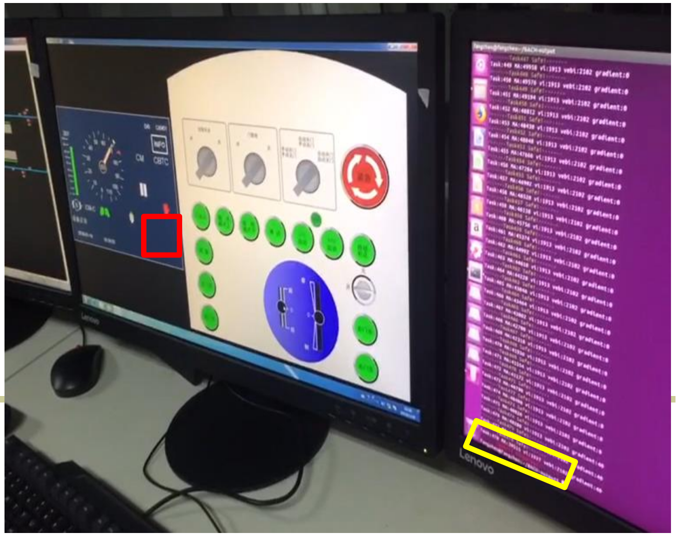
- Tool: BACH
  - Graphical Editor, Model Checker, Eclipse Plugin, Web Application… more than 8 components and 20 versions
  - More than 200 Globally Download, including researchers from UCB, CMU, UBC and engineers from industry.
  - BMC Area Chair of ARCH Competition 2017， 2018

- Publications
  - Around 40 papers: IEEE TC， IEEE TPDS、ACM TCPS、FMSD、 STTT、 RTSS、CAV、FMCAD、DSN、ICCPS、DATE、VMCAI、FORTE and so on
  - 11 Software Copyrights, 8 Chinese Patents

Selected Application: CPS

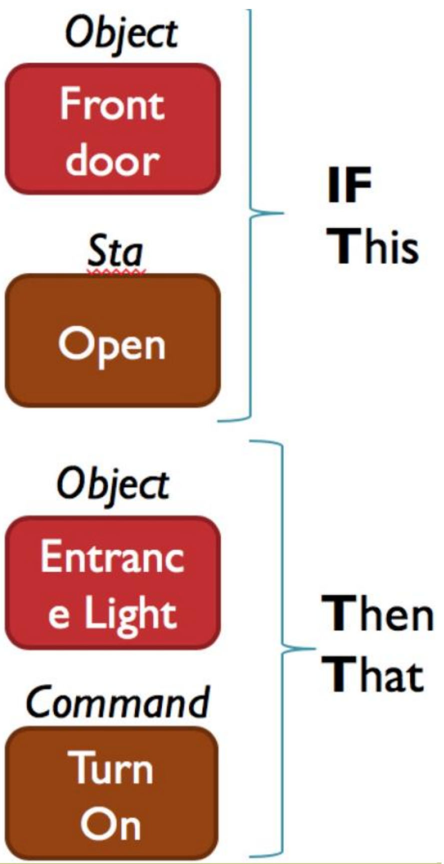
- Real-life CPS show high nondeterministic behavior
  ➔ classical offline model checking does not work
- Our solution:
  - Parametric hybrid system modeling, Online Concretization
  - Online periodical real-time hybrid systems model checking of time-bounded future!
- Implemented a special version BACH$_{OL}$ for CPS online verification
- Deployed on National Engineering Research Center of Rail Transportation Operation and Control System

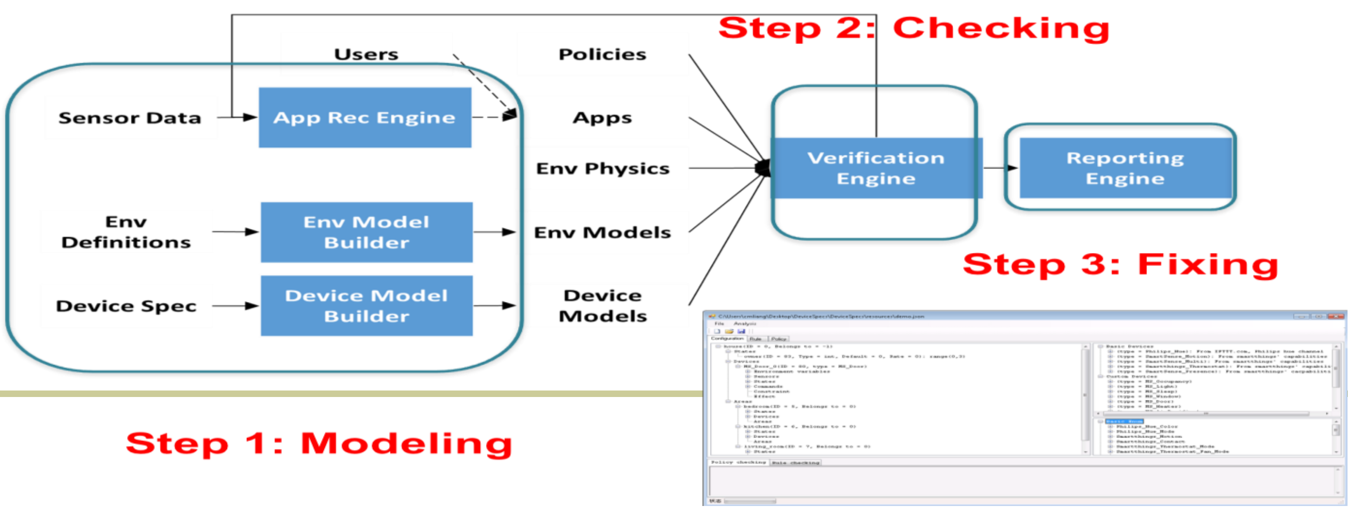




Selected Application: IoT

- IFTTT-style event triggering IoT system is widely believe to be an important enabling building block of IoT
- Will an IoT app meet an user's expectations? Will there be any unsafe consequences?
- We propose a framework of Modeling, Verification and Fixing of Smart Home System as Real time hybrid system automatically
- BACH is the underlying checker
- Selected into Microsoft TechFest'15 for technology transfer





Step 1: Modeling

Conclusion

- By isolating the discrete path and related continuous behavior into different layers, the **complexity** of our approach is well-controlled
- By integrating SAT, LP and IIS, the **performance** of our tool outperforms the state-of-the-art SMT solvers significantly
- Use the byproduct of BMC, IIS, to derive an unbounded result (***Extra Benefit!***)
- On going work: **Code Verification**
  - Software code shares similar feature with hybrid system
    - Transition system with constraints, infinite state space…
- **Public available** from http://seg.nju.edu.cn/BACH/

Thanks

Questions?