Exploiting Hierarchy in the Abstraction-Based Verification of Statecharts Using SMT Solvers

Bence Czipó[1], Ákos Hajdu[1,2], Tamás Tóth[1], István Majzik[1]

[1]Department of Measurement and Information Systems,
Budapest University of Technology and Economics

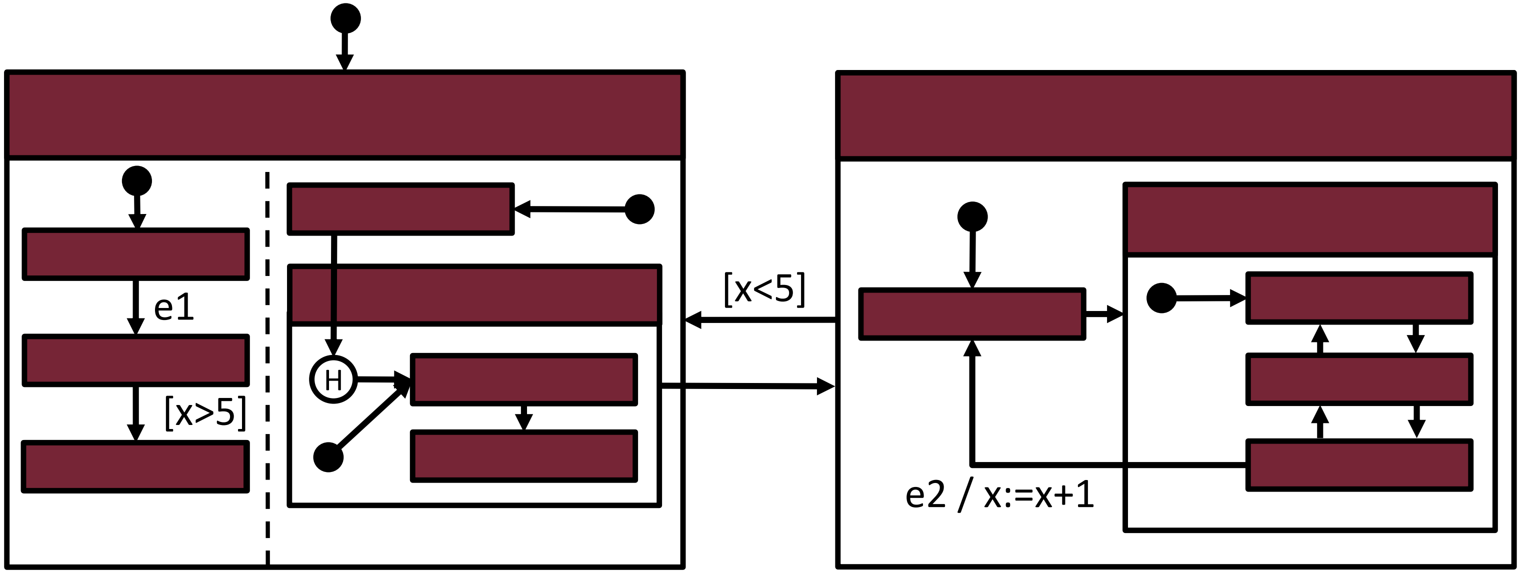
[2]MTA-BME Lendület Cyber-Physical Systems Research Group,
Budapest, Hungary

FESCA 2017, Uppsala, Sweden, 22.04.2017.

Introduction

Formal modeling

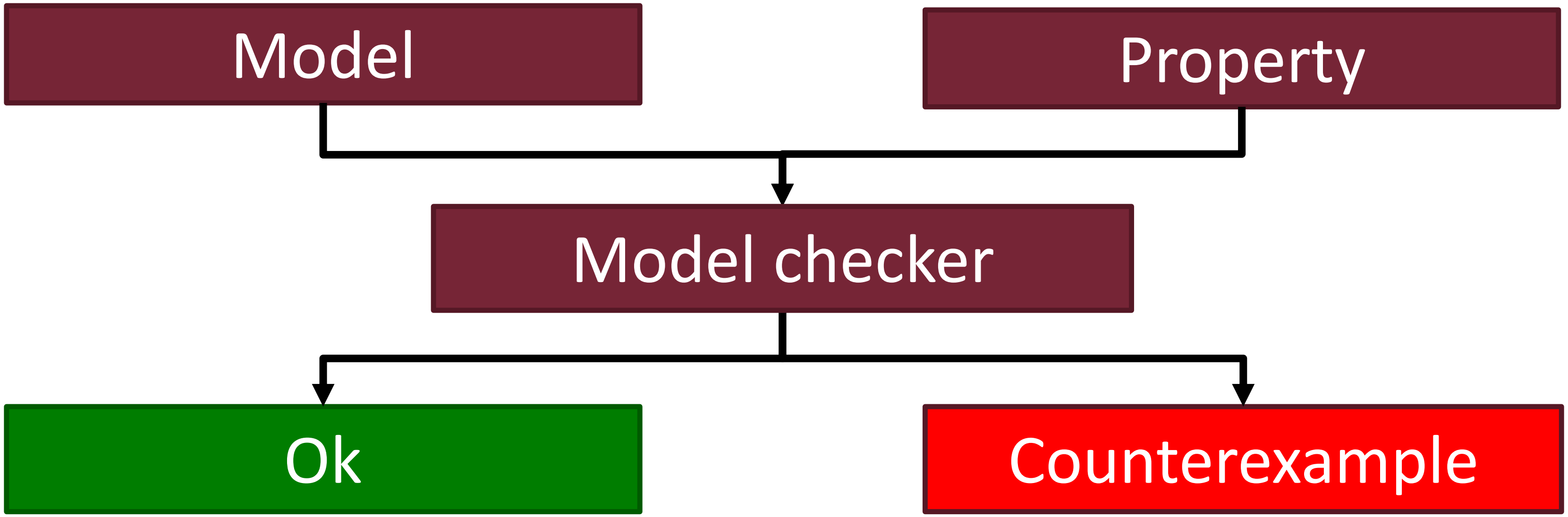
- Hierarchical statecharts
  - Modeling state-based systems
  - High level
  - Formal semantics



Formal verification

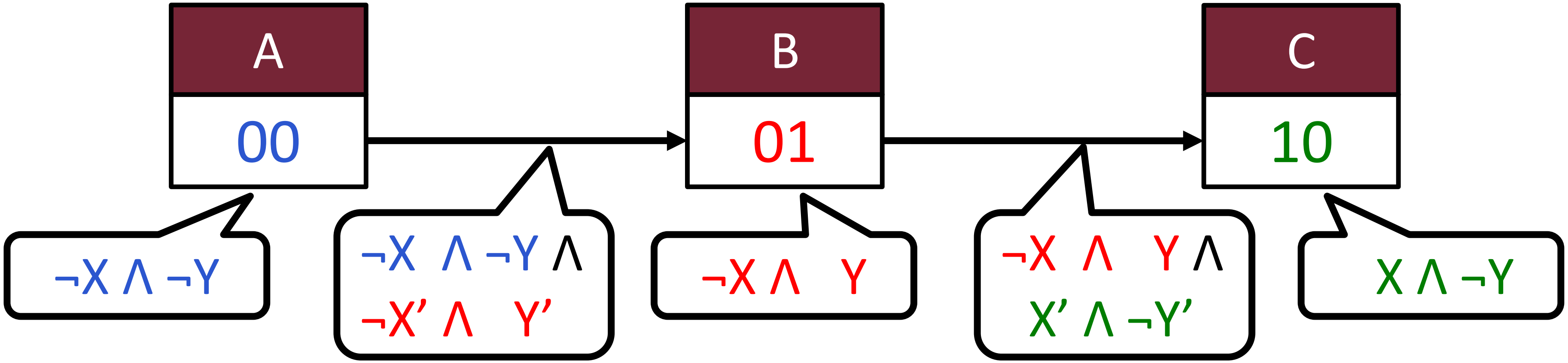
- Proving correctness
- Model checking
  - State space explosion

We focus on reachability



Efficient model checking

- Encoding states/transitions to logical formulas



A

00

$\neg X \land \neg Y$

$\neg X \land \neg Y \land$

$\neg X' \land Y'$

B

01

$\neg X \land Y$

$\neg X \land Y \land$

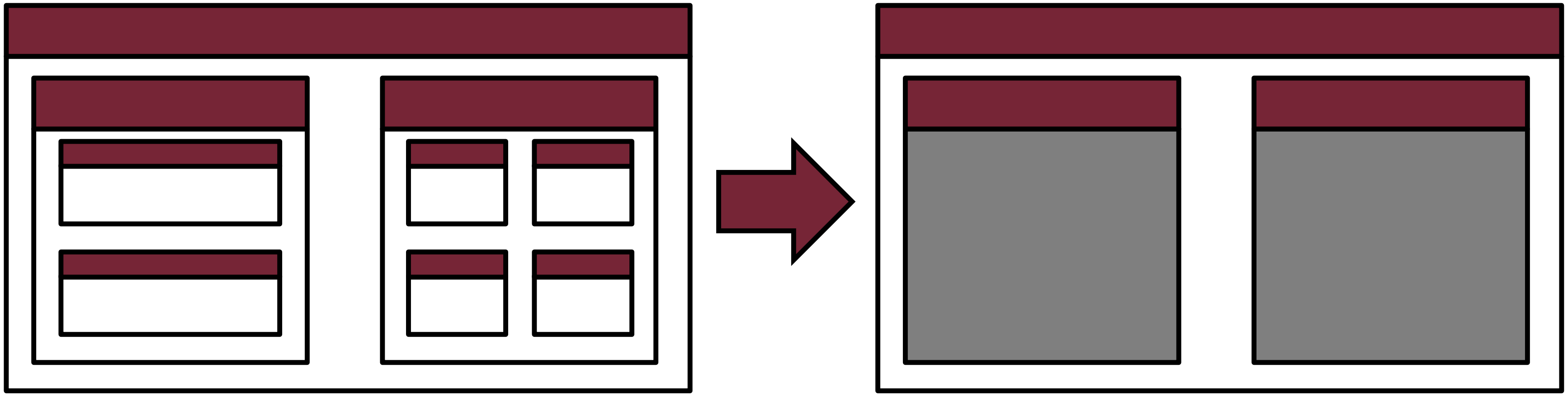
$X' \land \neg Y'$

C

10

$X \land \neg Y$

- Abstraction and refinement



Motivation

- Model checking of statechart models
  - Complex models, large number of state configurations

    → Abstraction, bounded model checking

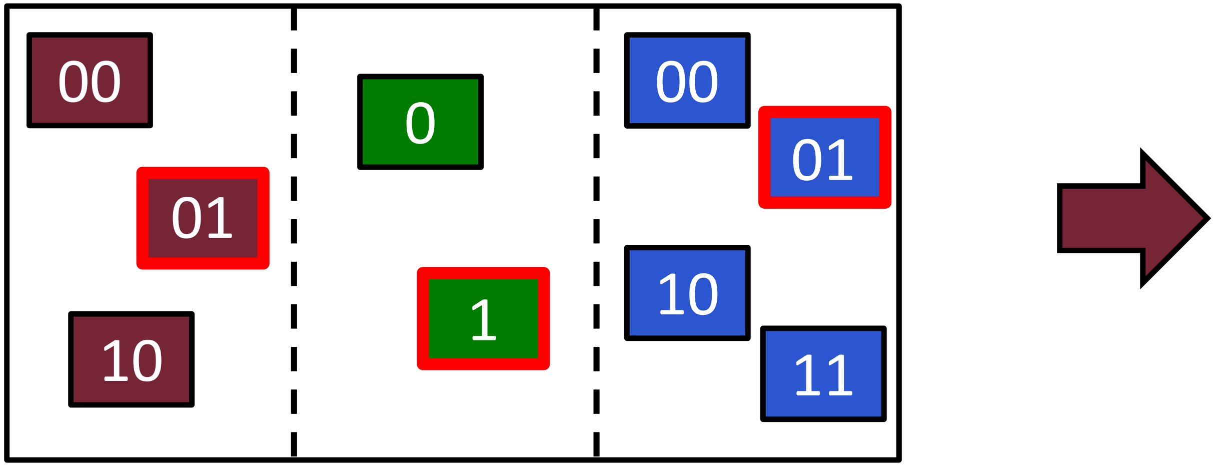
- Abstraction-based model checking
  - CEGAR: Counterexample-Guided Abstraction Refinement
  - Natural abstraction based on hierarchy and variables

- State space exploration and bounded model checking
  - Application of SAT/SMT solvers → encoding needed
  - Preserving hierarchy and parallelism for abstraction

Hierarchy preserving encoding

Encoding parallel regions

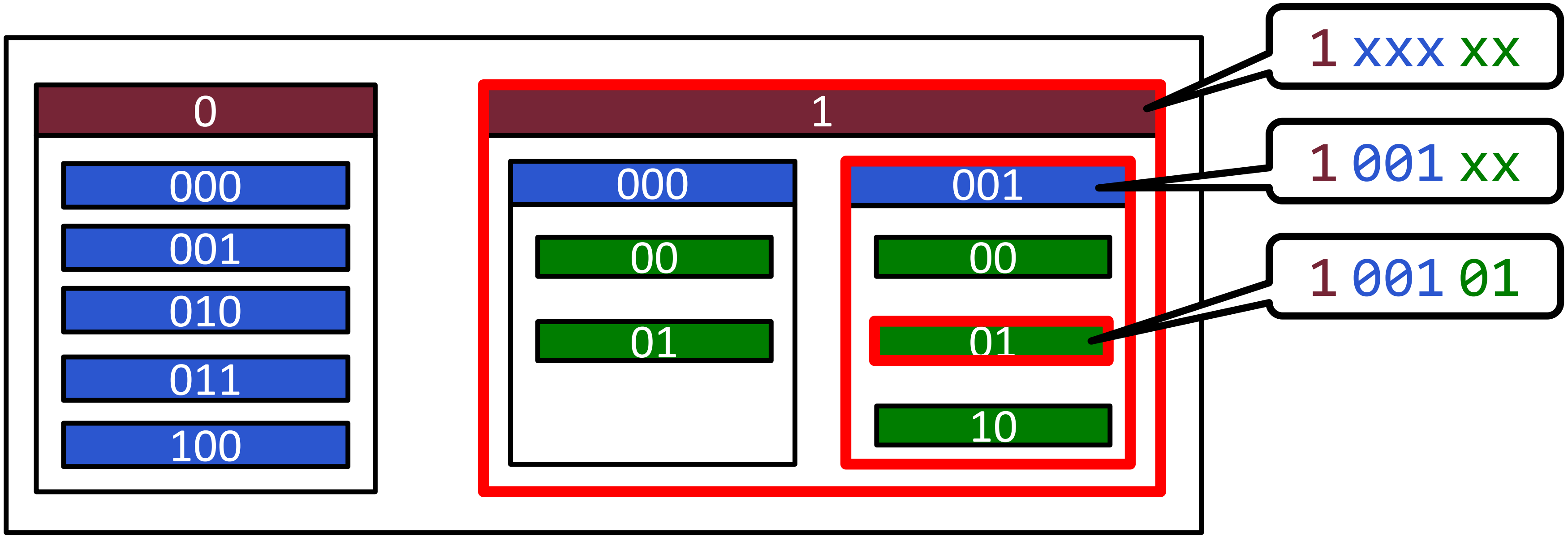
- Parallel regions
  - Each region gets its own segment
  - Can refer to individual states
    - Fill other segments with don’t care bits
  - Can refer to a whole configuration



```
01 x xx
xx 1 xx
xx x 01
-- - --
01 1 01
```

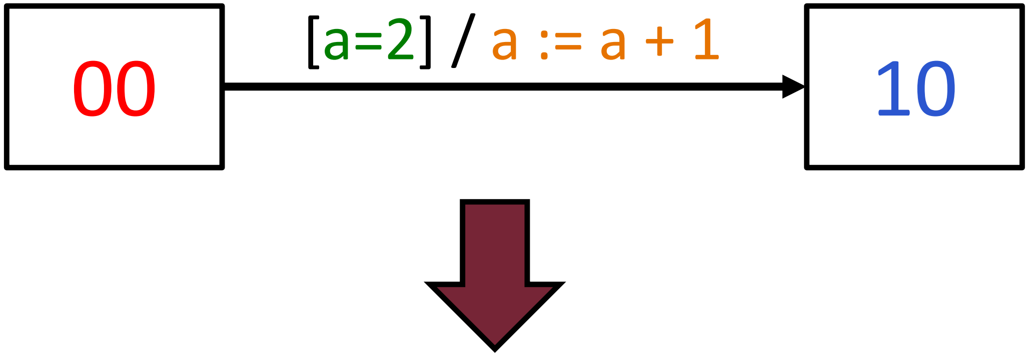
Encoding hierarchy

- State hierarchy
  - Each level gets its own segment
  - Can refer to composite states
    - Fill remaining bits with don’t care bits
  - Can refer to simple states
    - Using segments of parent states



Other supported elements

- Variables of the statechart
  - Extra variables besides the encoding
- Transition expressions: SMT formulas
  - Guards
  - Assignments

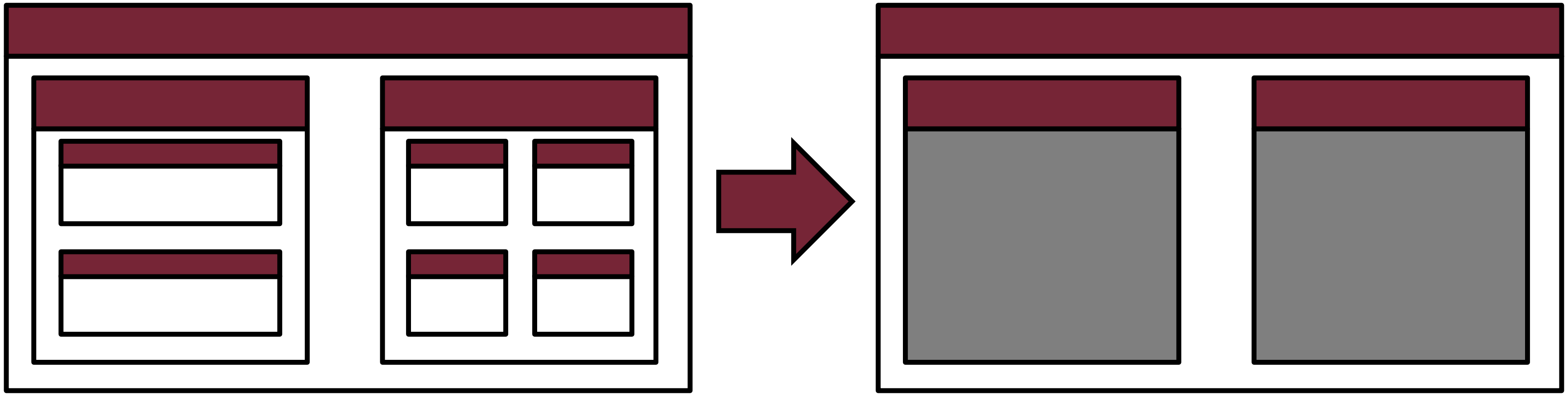


$$\neg X \land \neg Y \land X' \land \neg Y' \land a = 2 \land a' = a + 1$$

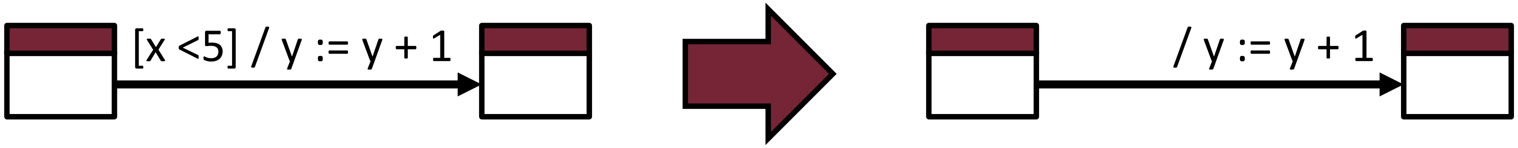
Applying CEGAR

Abstraction of statecharts

- Expand composite states up to a certain depth



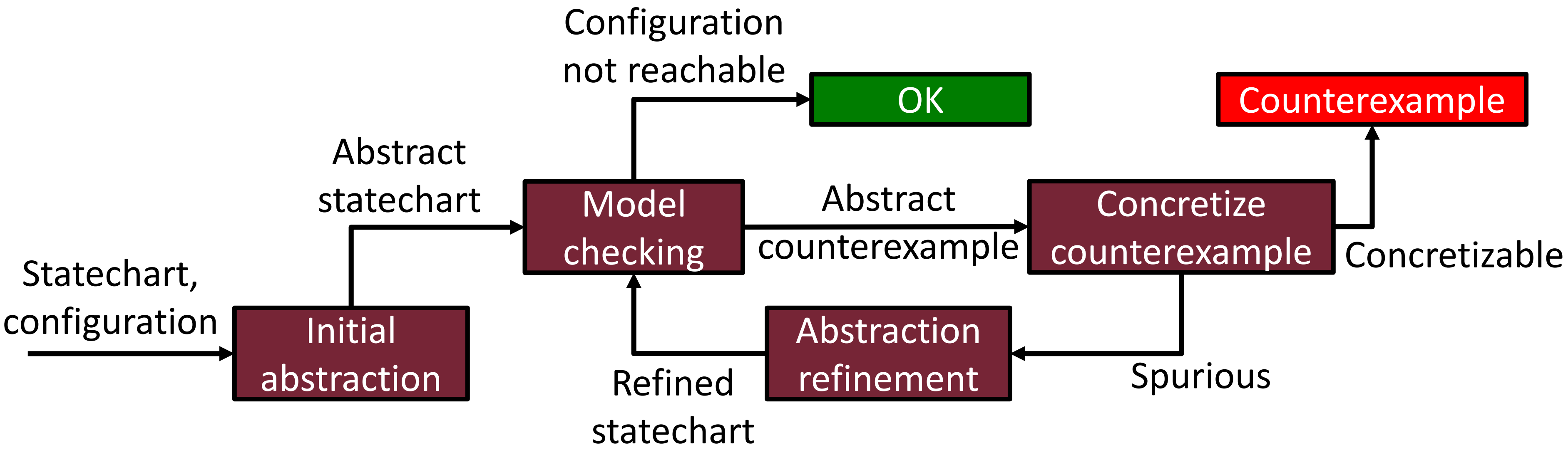
- Hide certain variables and expressions



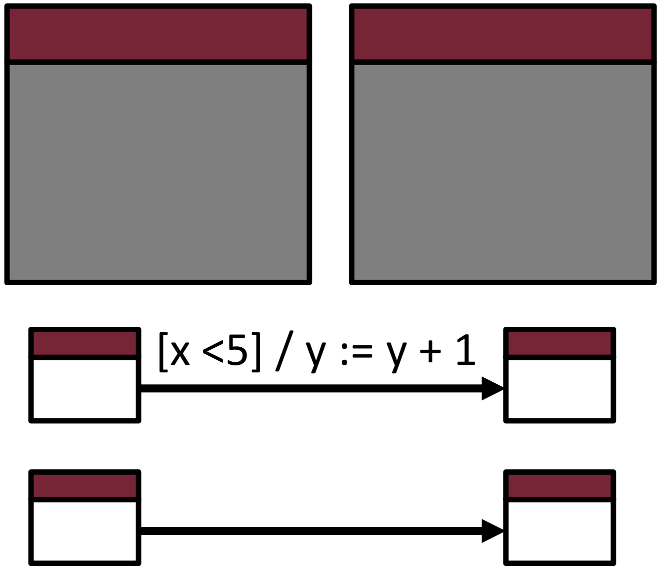
- Precision of abstraction? Fine ↔ coarse
  - Determine automatically: CEGAR

CEGAR

- Counterexample-Guided Abstraction Refinement
  - Start with a coarse abstraction
  - Refine until proper precision is reached
- CEGAR adapted to reachability in statecharts



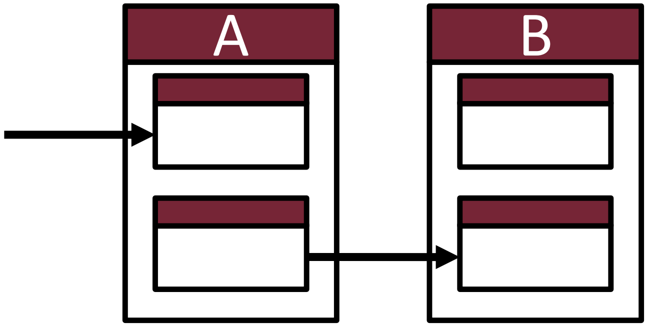
Initial abstraction, model checking

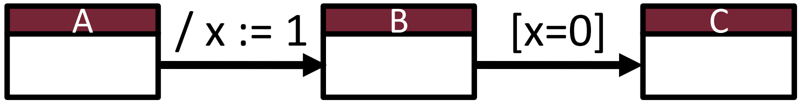


- Initial abstraction
  - Only the top level is expanded
  - Variables
    - All visible (states only abstraction)
    - All hidden (generic abstraction)
- Model checking
  - Using the encoding and an SMT solver
  - Bounded model checking (BMC)
    - Find counterexamples within a bound *k*
  - Systematic exploration
    - Explore abstract state space

Concretization, refinement

- Concretization
  - Abstract counterexample: sequence of abstract states
  - Find corresponding concrete sequence
  - Similar to bounded model checking
- Refinement (in case of spurious counterexample)
  - No concrete transitions
    - Expand hierarchy one level deeper
  - Transition not enabled
    - Due to hidden variables
    - Make variables visible

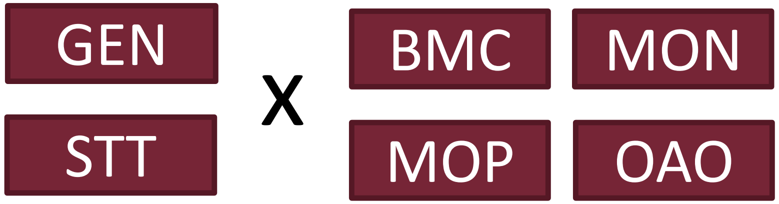




Evaluation

Implementation

- 2 abstractions
  - States-only (STT)
  - Generic (GEN)
- 4 model checkers
  - Bounded (BMC)
  - Systematic exploration
    - MON: basic implementation
    - MOP: uses push-pop functionality of solver
    - OAO: lazy exploration (one state at once)



GEN

STT

x

BMC

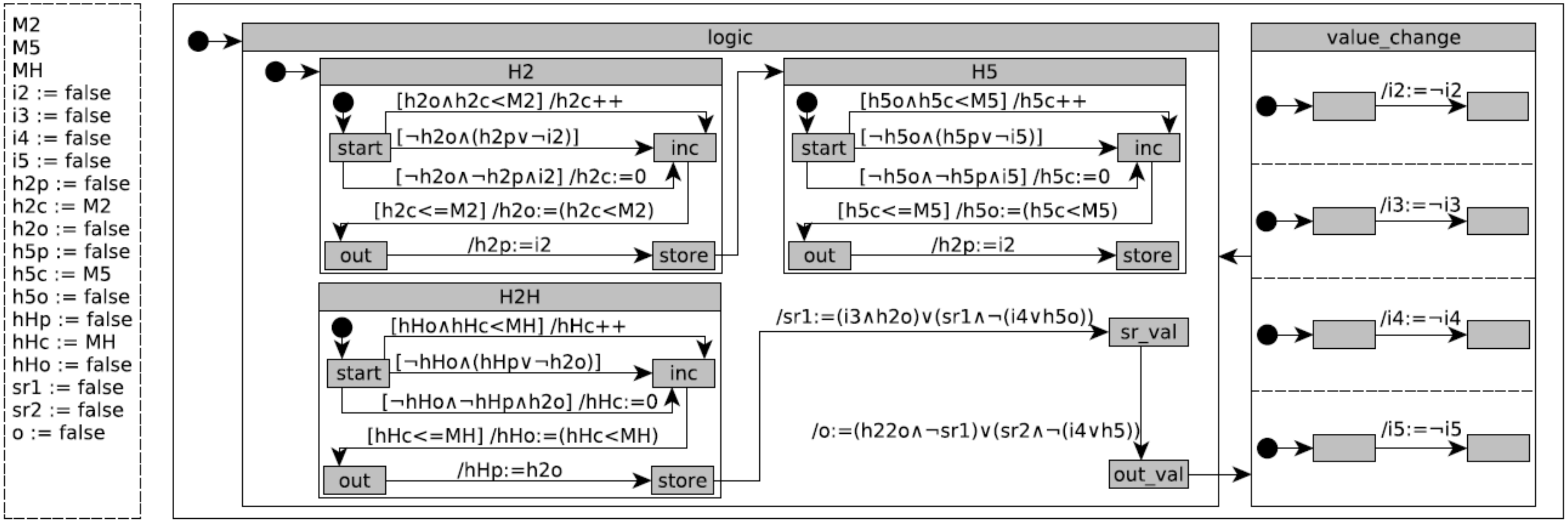
MON

MOP

OAO

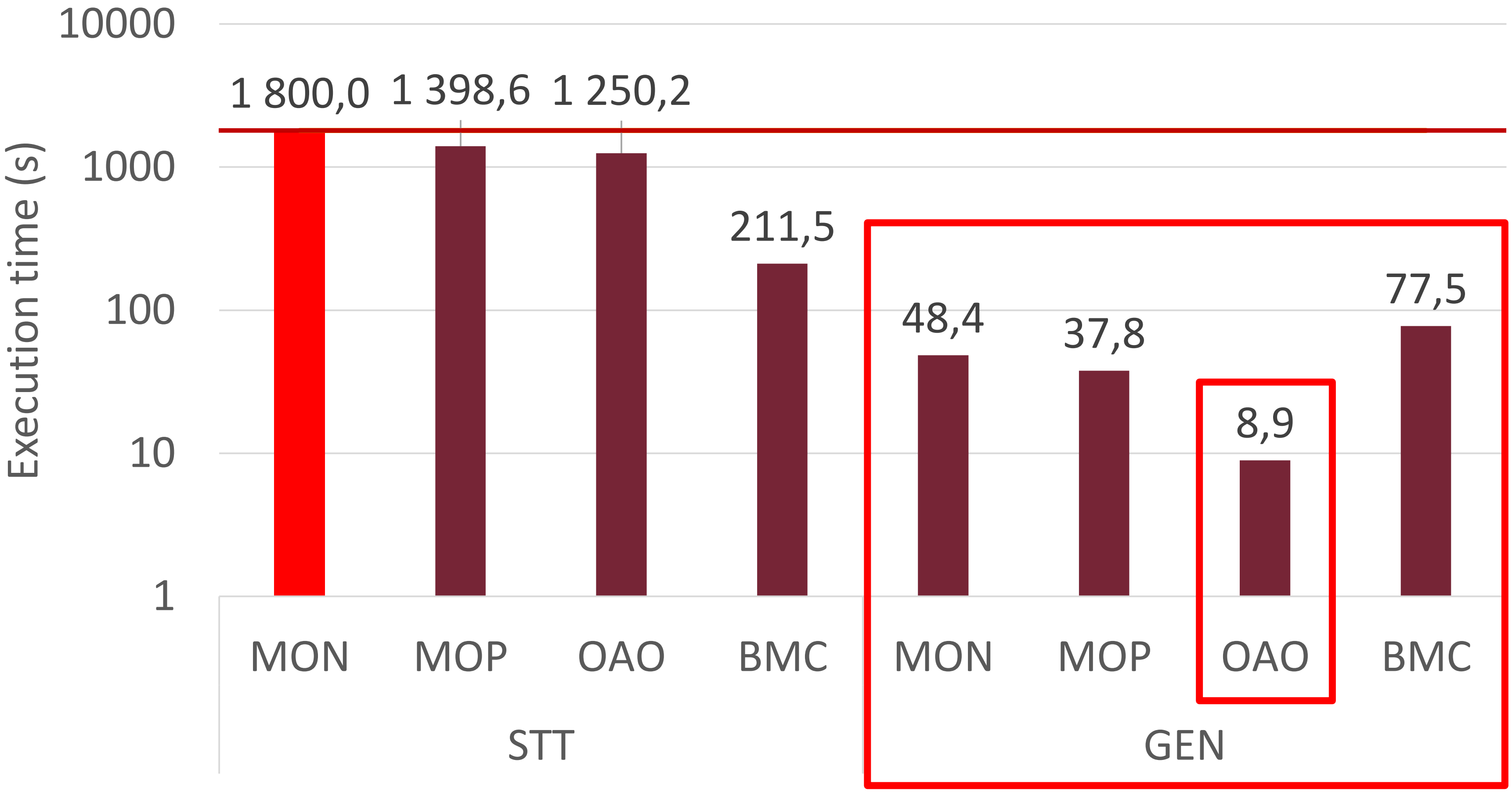
Case study

- Evaluation: industrial control system
  - (Part of) the safety logic of a power plant
  - Parameterizable (size of state space)
  - 27 states (5 composite, 22 simple) in 9 regions (4 parallel)
  - 16 variables (3 int, 13 bool)
  - 27 transitions



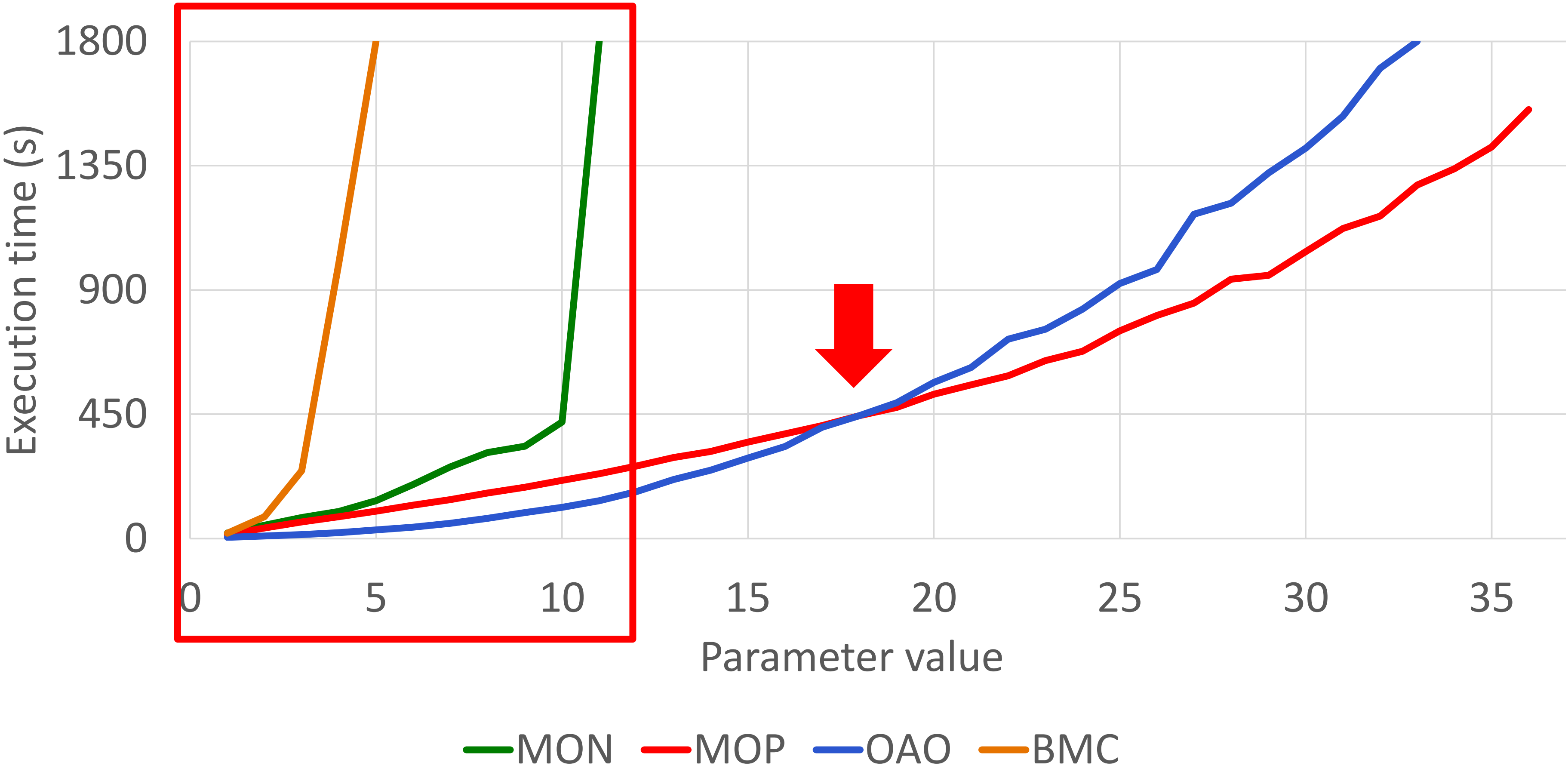
Evaluation

- Results for small parameter value



Evaluation

- Scalability with the increase of the parameter
  - Generic abstraction only

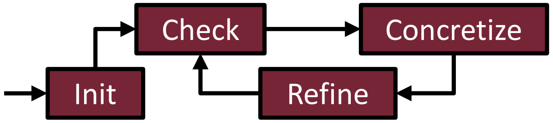


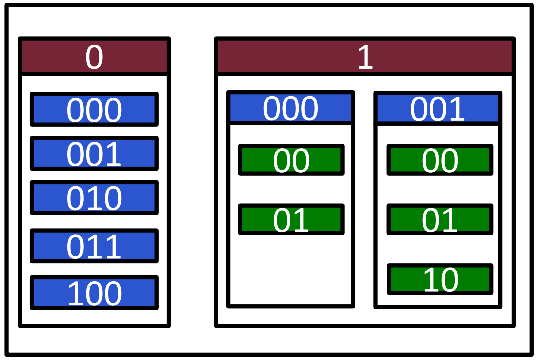
Conclusions

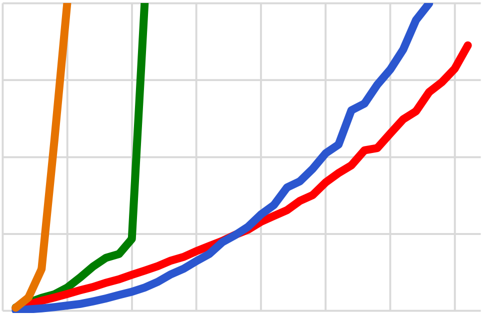
Conclusions

- Results
  - Adaptation of CEGAR to statecharts
    - Abstraction and refinement techniques
    - Exploiting hierarchy
  - Based on hierarchy preserving encoding
    - Utilizing SMT solvers
  - Evaluation
- Future work
  - Extending the supported elements
  - Further abstractions and refinements
  - Compare to other algorithms/tools

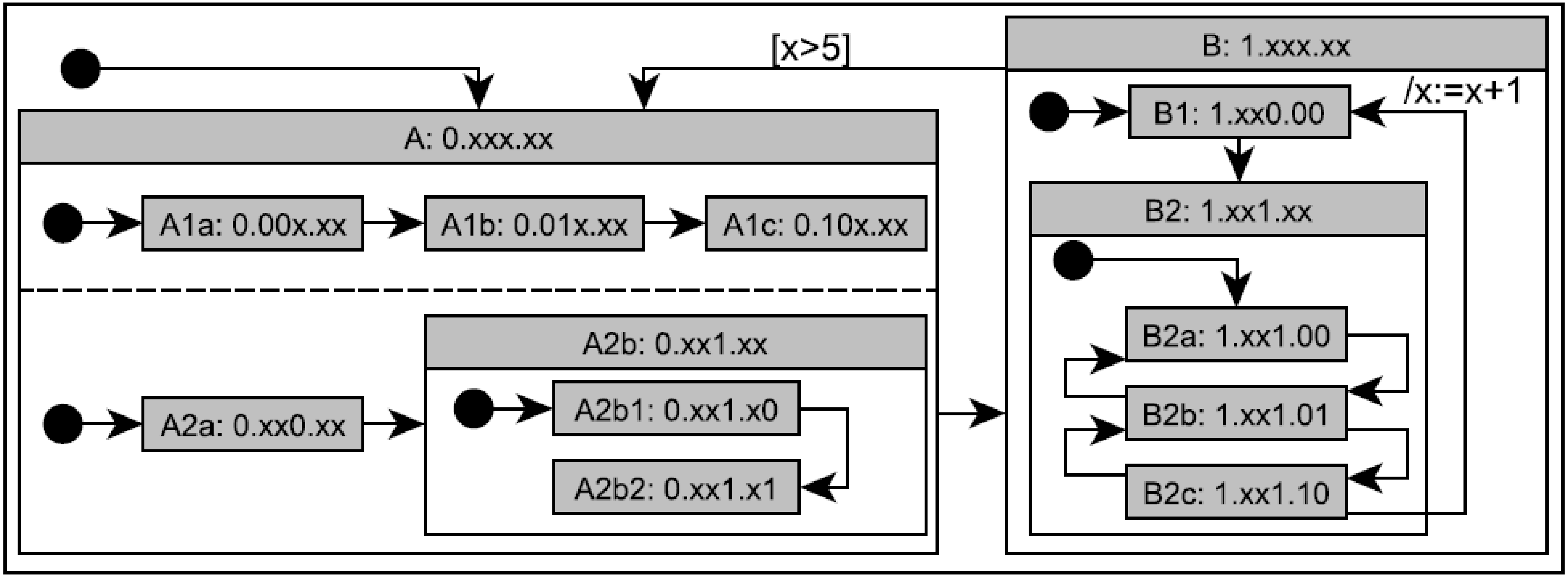
hajdua@mit.bme.hu
inf.mit.bme.hu/en/members/hajdua


Init
Check
Concretize
Refine

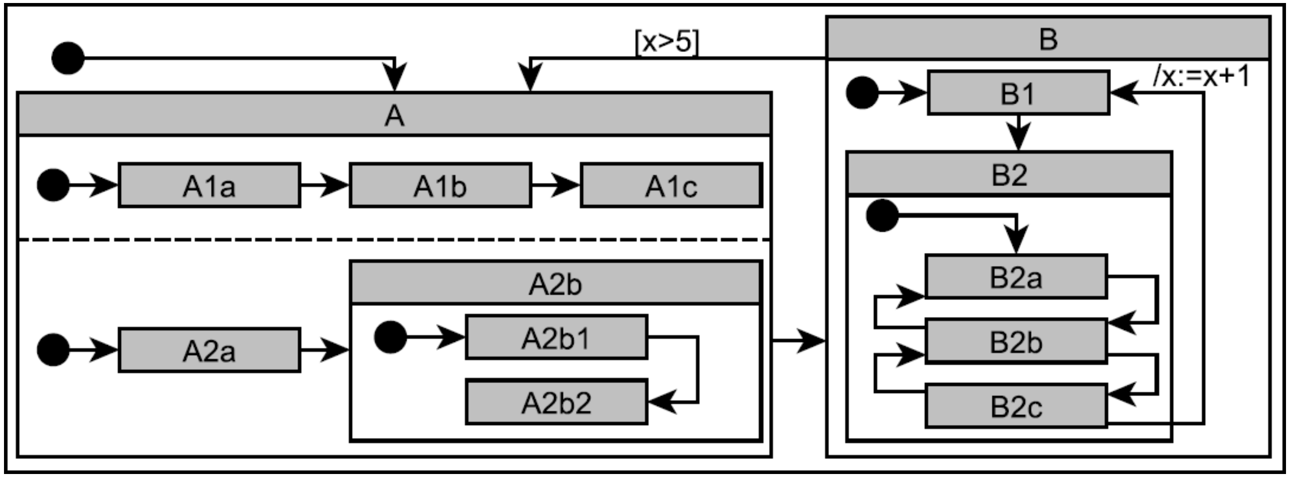




Encoding example

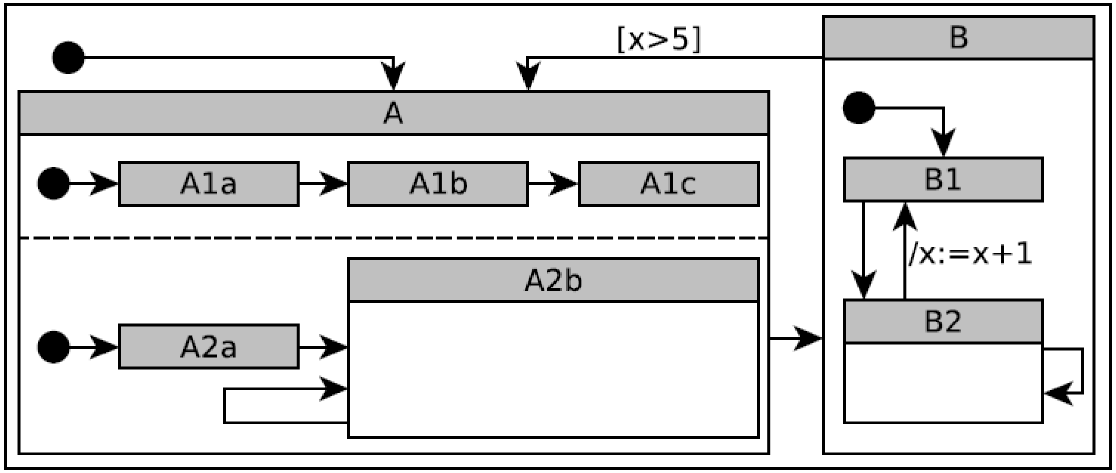


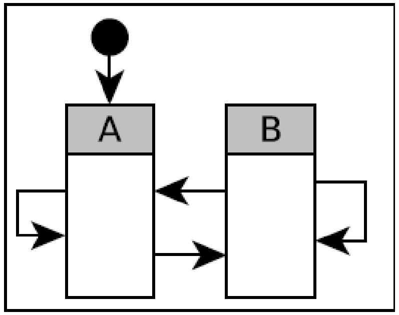
Abstraction example



Fine

Coarse





Evaluation details

- For small parameter value

Abstraction	Checker	Time (s)	Iterations	Max. configs.	Final configs.
STT	MON	Timeout	*(2)*	*(8610)*	*(8610)*
STT	MOP	1399	5	17036	2855
STT	OAO	1250	5	17036	2855
STT	BMC	211	5		
GEN	MON	48	12	1484	1484
GEN	MOP	38	12	1484	1484
GEN	OAO	9	12	1484	1484
GEN	BMC	77	12		