Universität Bremen

Automated Analysis of Robustness for Circuits

Görschwin Fey

University of Bremen, Germany
fey@informatik.uni-bremen.de

The City

- An old trading town (~1200 years old)
- Population 550,000

The University

- Founded in 1971
- Approx. 20000 students
- Computer Science
  - 20 Professors
  - ~1500 students

Universität Bremen

Automated Analysis of Robustness for Circuits

Görschwin Fey

University of Bremen, Germany
fey@informatik.uni-bremen.de

Outline

- Motivation
- What is Robustness?
- The Model and its Justification
- Experimental Results
- What about the Environment?
- Summary
- Future Work

Soft Errors Increase

Wilhelm Busch: Max & Moritz

- Aging/Induced
- Permanent/Temporary

Solution: Fault Tolerance

- Add fault tolerance
  - Multiple design levels
  - Algorithm vs. structure

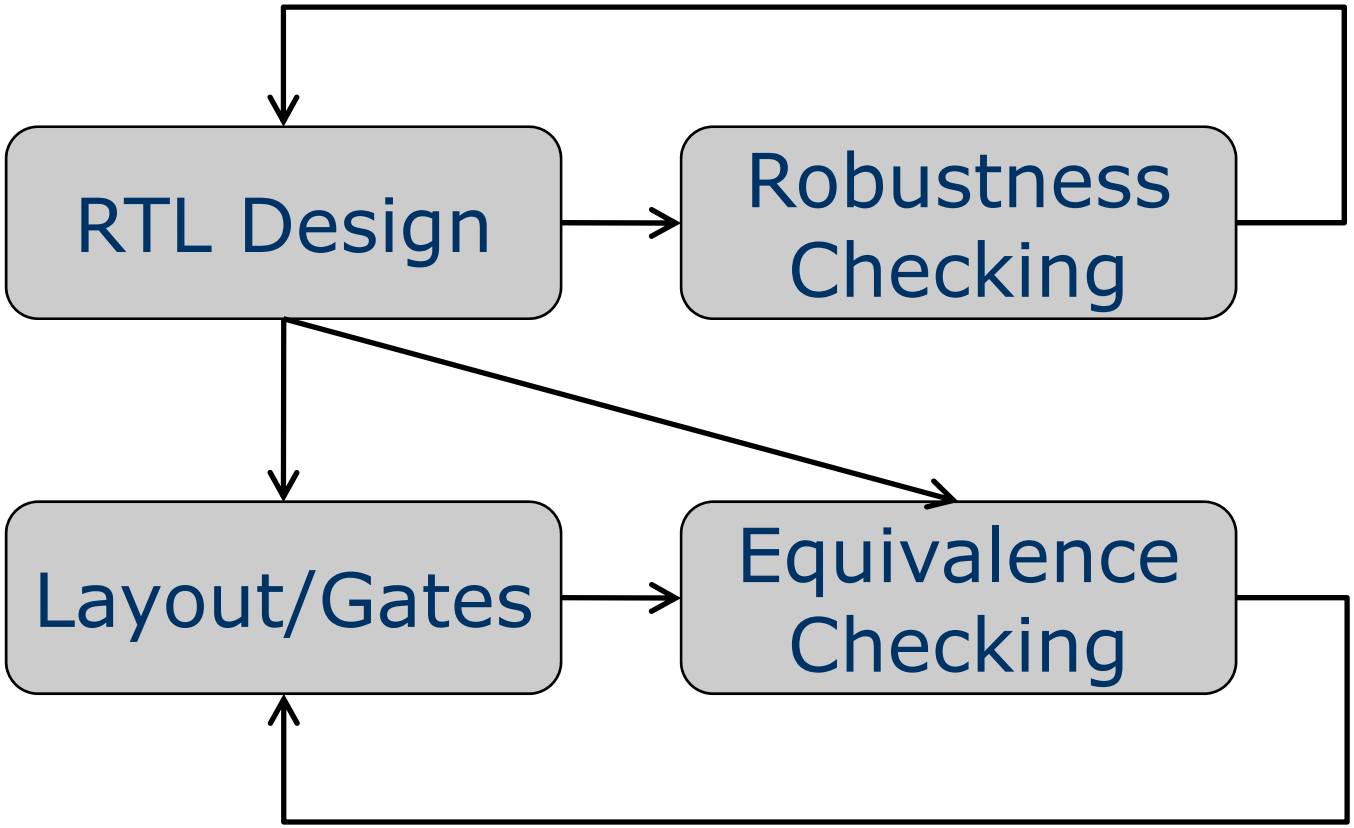
By Drift Words, flickr.com

Solution: Fault Tolerance

- Add fault tolerance
  - Multiple design levels
  - Algorithm vs. structure
- Error prone
  - Design faults

Implementation of fault tolerance requires verification

Proposed Flow



What is Robustness?

- Here:
  - Robustness ≠ Reliability
  - Robustness =
    Correct behavior under fault assumptions
- How to check and prove?

Assessing Robustness

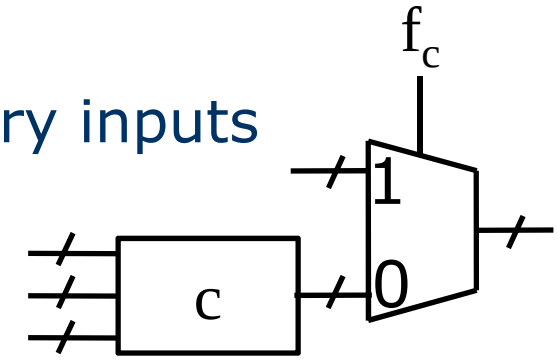
- Analysis using design mutations
- Simulation
- Emulation [CMR+2002,PCZ+2008]

} incomplete

- Formal analysis
  - Classification of states [Lev2005]
  - Based on properties [KPJ+2006, SLM2007]
  - BDD-based [BCT2007, HPB2007]
  - BMC-like model [FD2008]
  - No bounds; reachability issues

Establishing a Formal Model

- In practice
  - Circuits contain fault detection logic
  - Only one transient fault within certain number of cycles
- Therefore use
  - Model of Bounded Model Checking (BMC)
  - Non-deterministic fault model
    - Replace component by primary inputs
    - Applies to transient faults
    - Also catches other faults



Classification of Components

- **Unrobust** (*S*): influence on output behavior
- **Unclassified** (*U*): no influence on output behavior, but inconsistent state observable
- **Robust** (*T*): otherwise

- Invariant: $\{S, U, T\}$ is partitioning of C

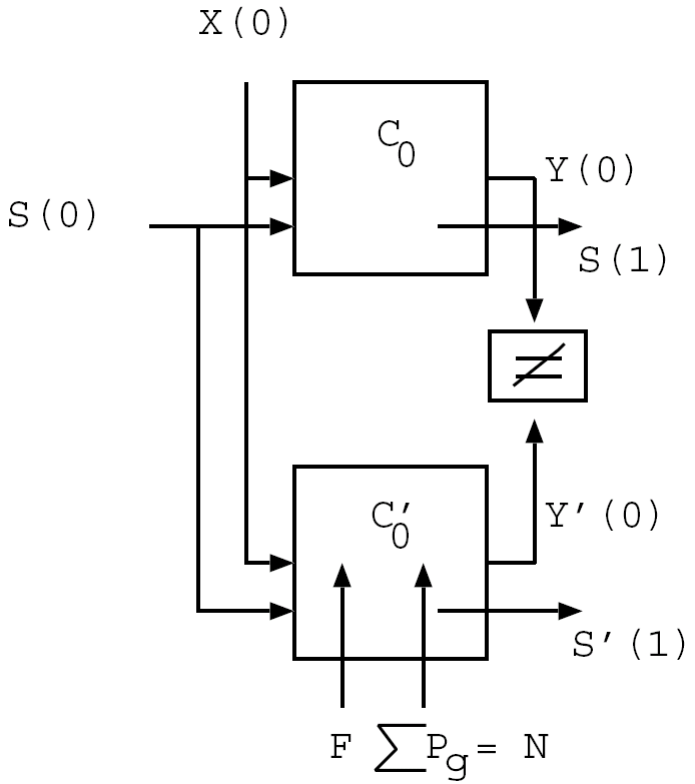
Bounds on Robustness

- **Unrobust** (*S*): influence on output behavior
- **Unclassified** (*U*): no influence on output behavior, but inconsistent state observable
- **Robust** (*T*): otherwise

- Invariant: $\{S, U, T\}$ is partitioning of C
- Lower bound: $R_{lb} = \frac{|T|}{|C|} = 1 - \frac{|S \cup U|}{|C|}$
- Upper bound: $R_{ub} = \frac{|T \cup U|}{|C|} = 1 - \frac{|S|}{|C|}$

Model (1)

Unrobust:



Influence on output behavior

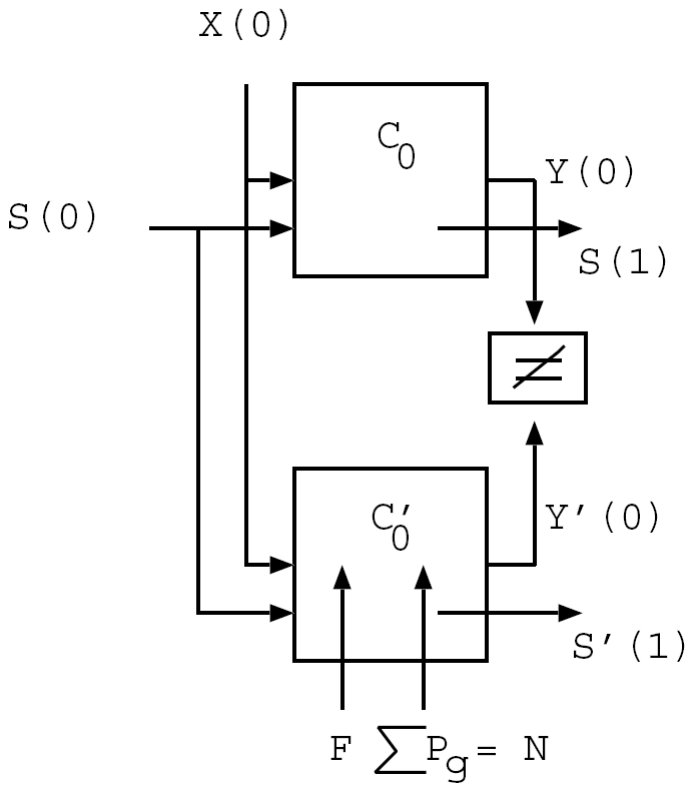
Addon:

- Fault detection logic
- Flags faulty behavior

→ Robust component

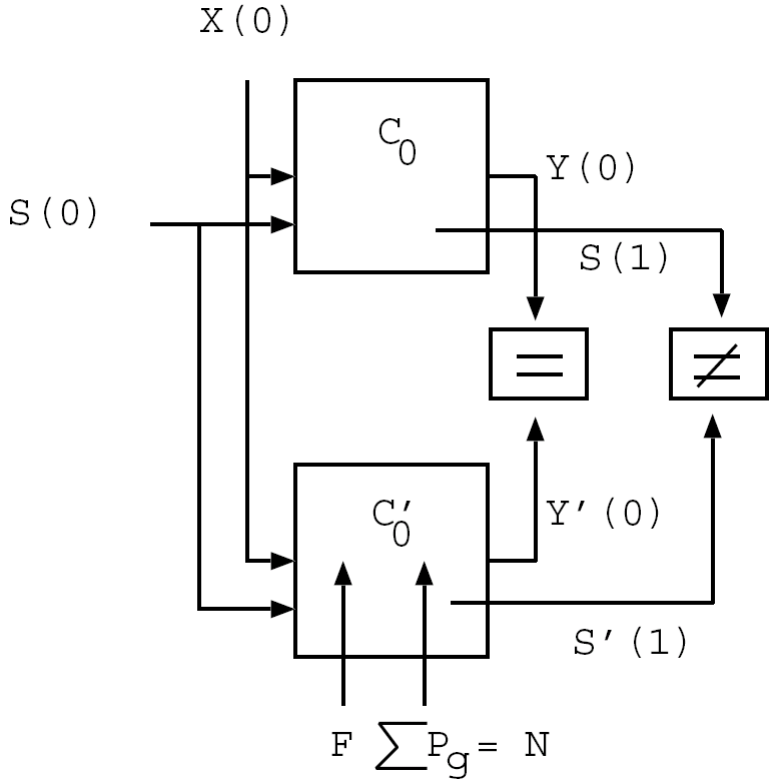
Model (2)

Unrobust:



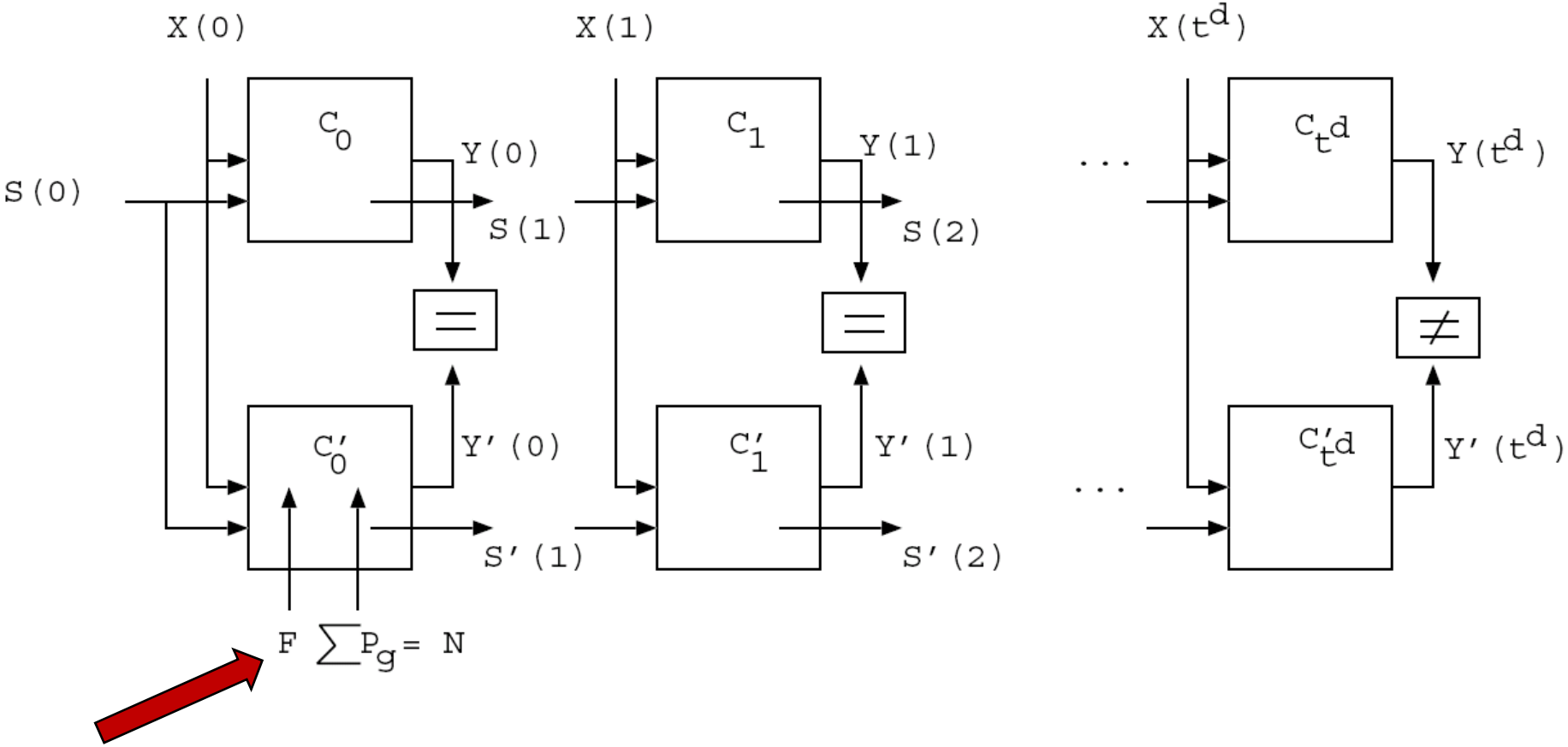
Influence on output behavior

Unclassified/Silent Data Corruption:



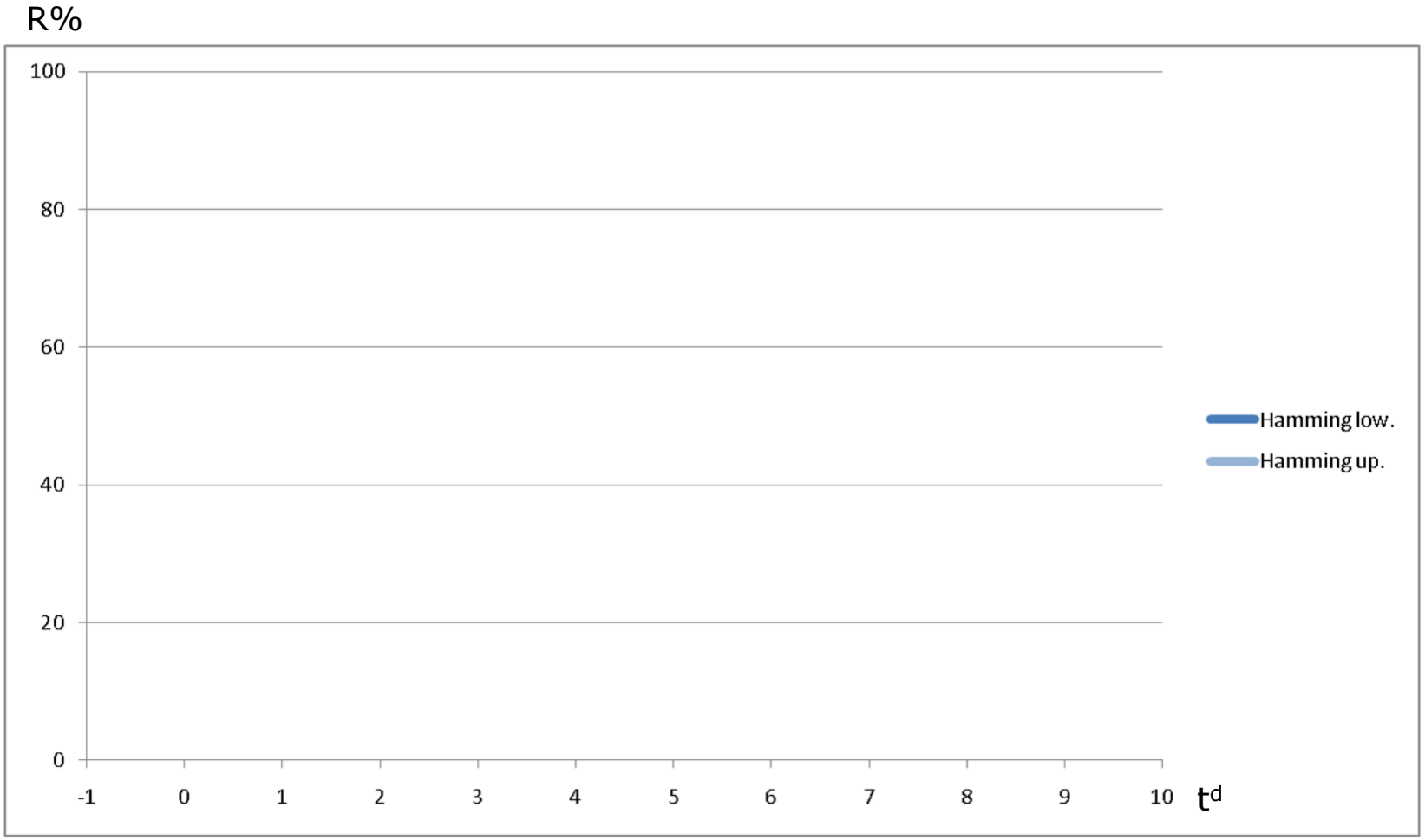
No influence on output behavior, but inconsistent state observable

Model (3)

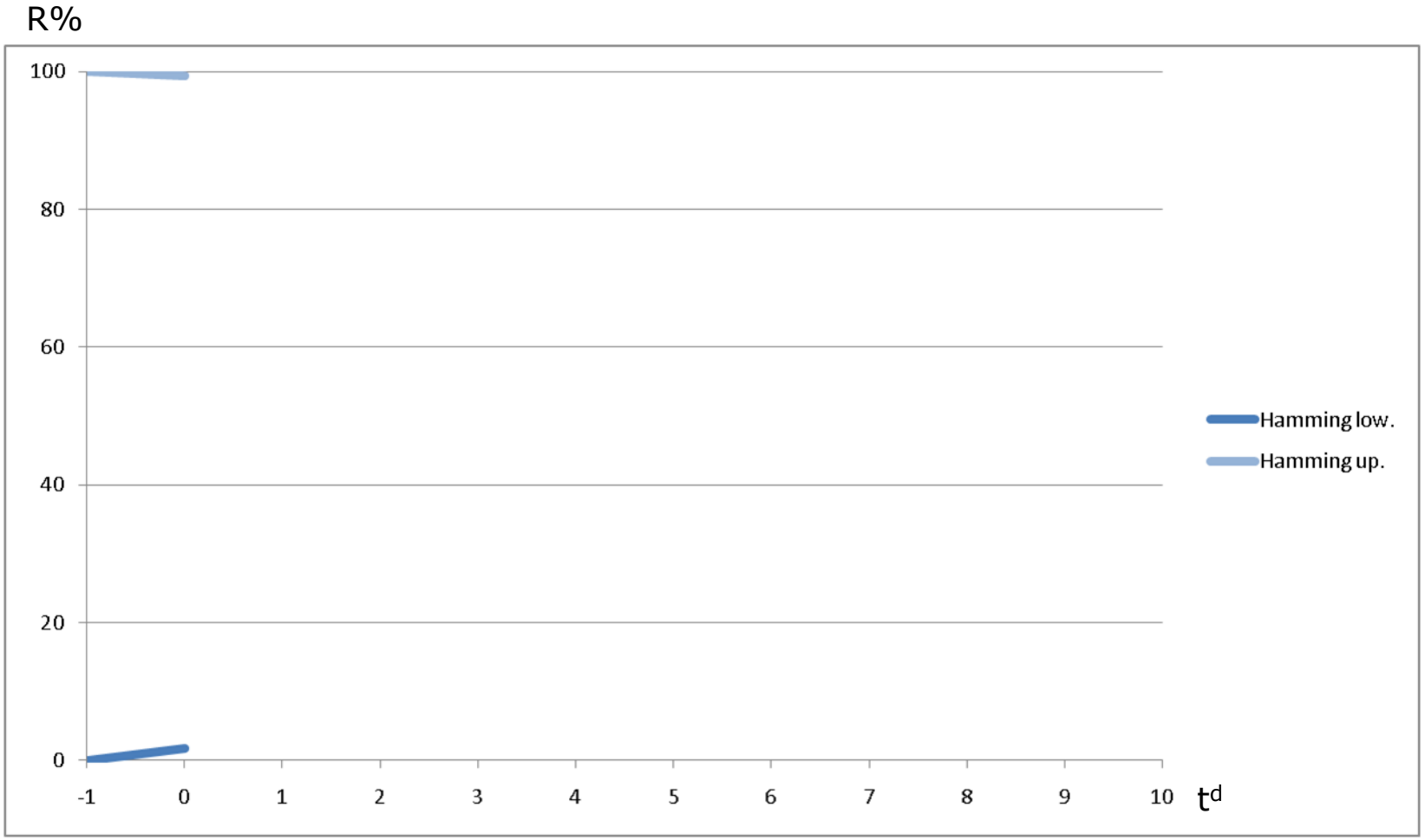


Single faults, i.e. in first time step only

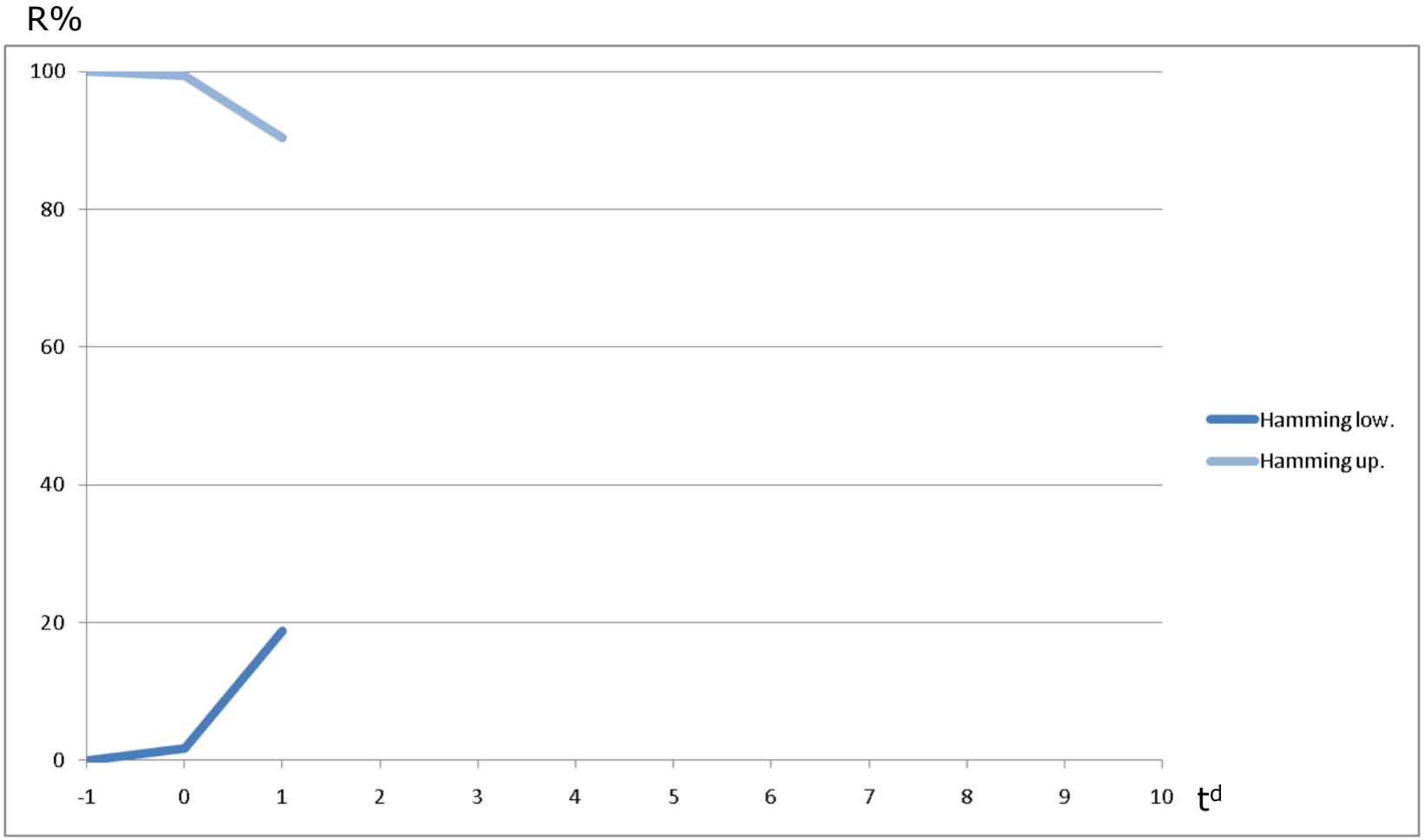
Bounds in Practice



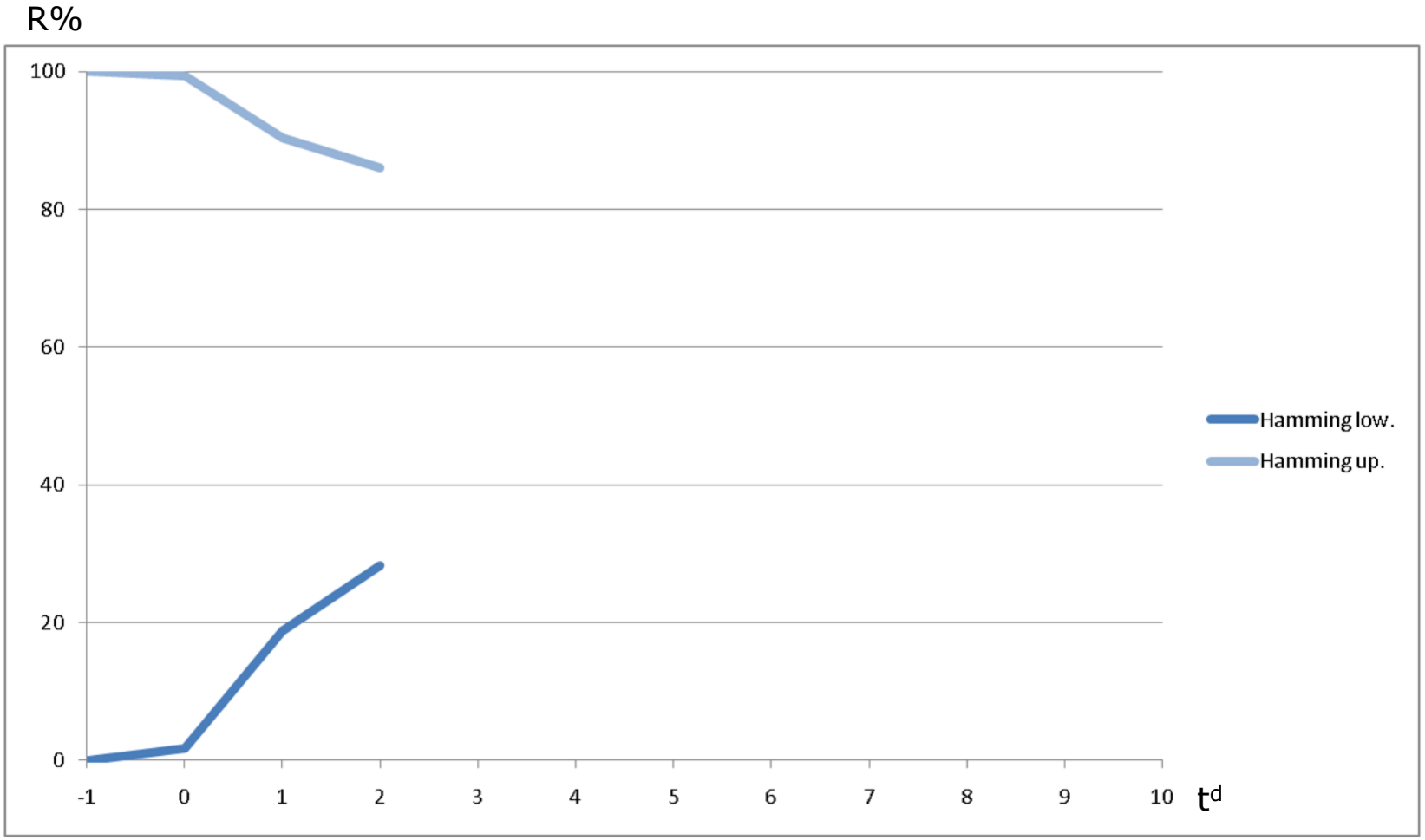
Bounds in Practice



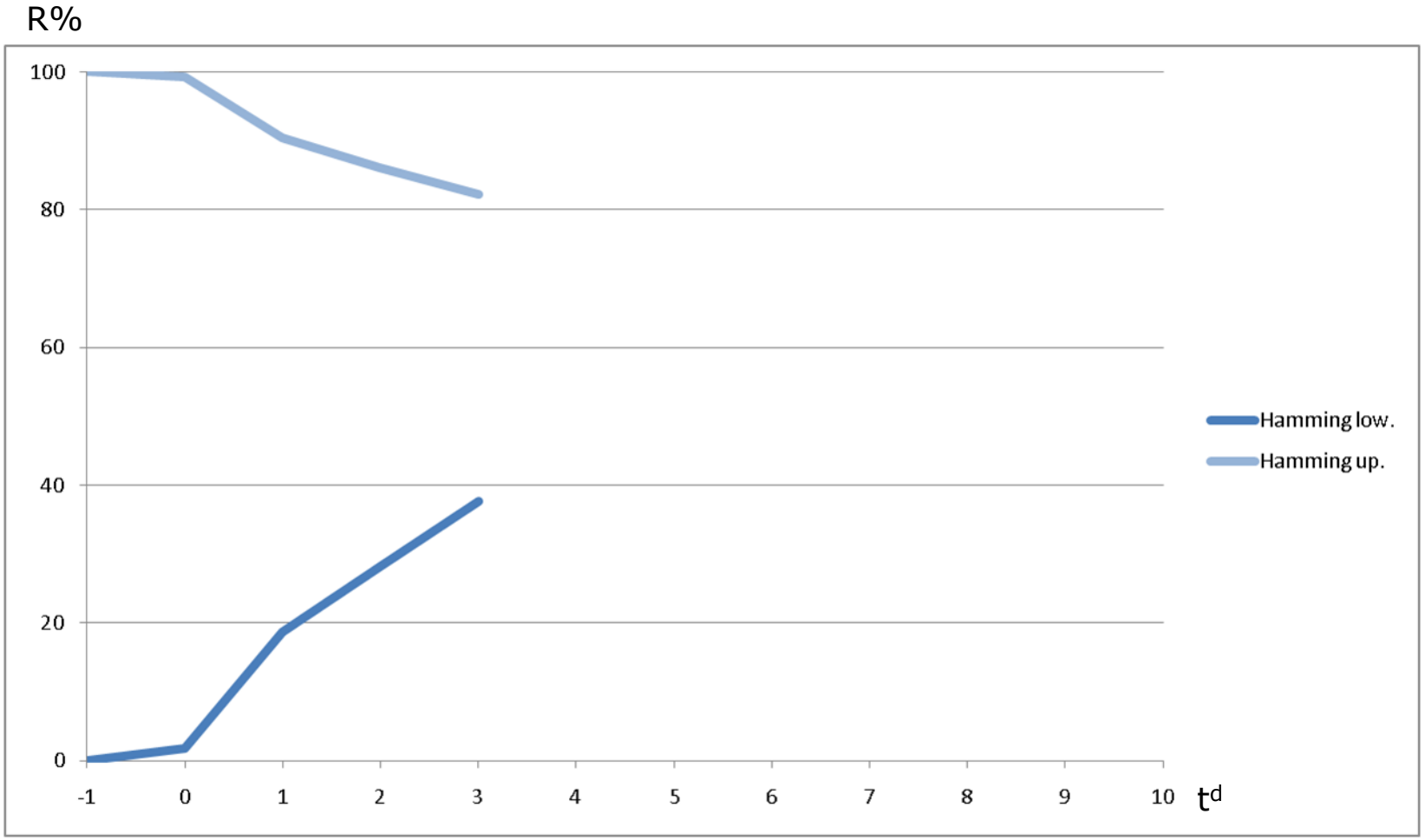
Bounds in Practice



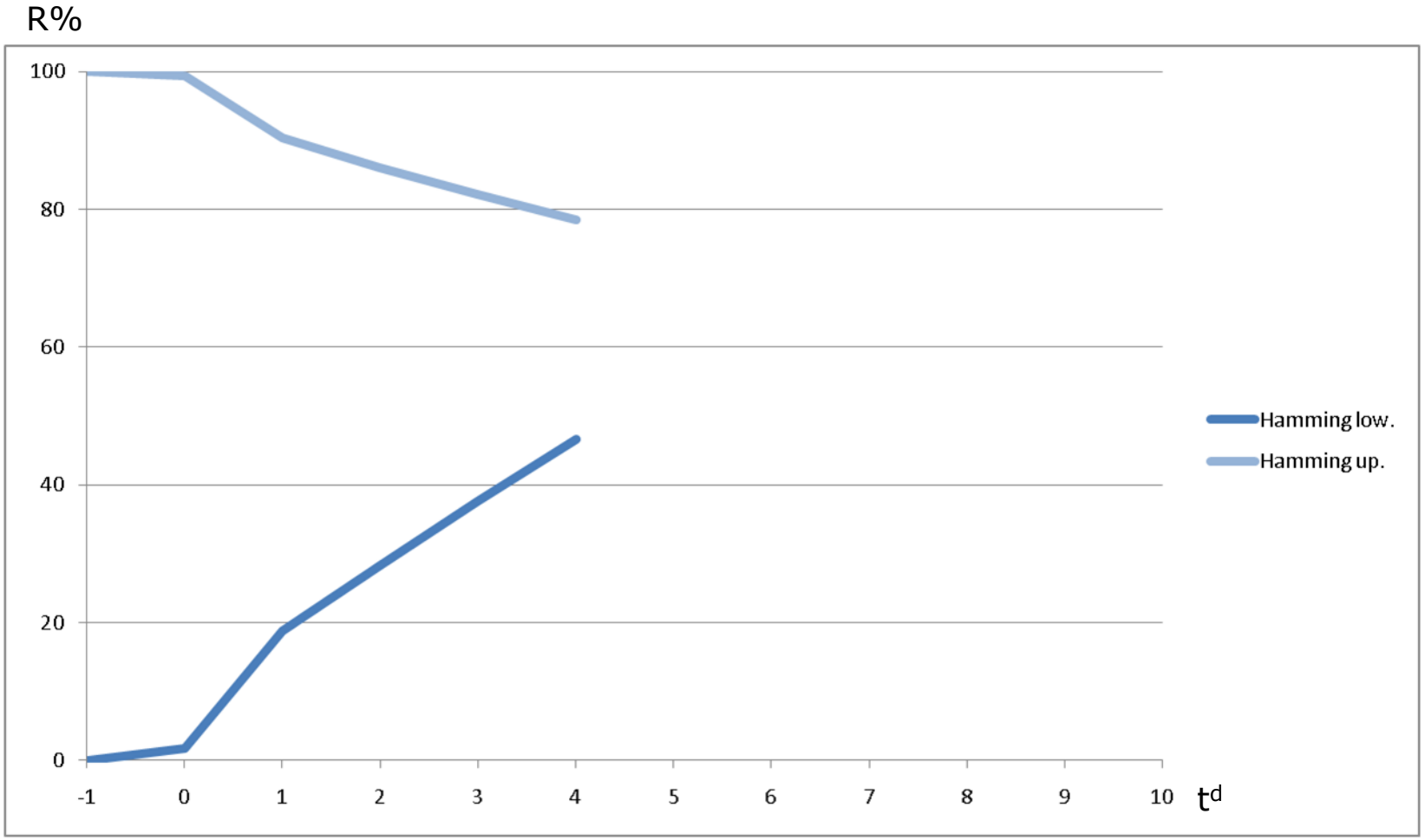
Bounds in Practice



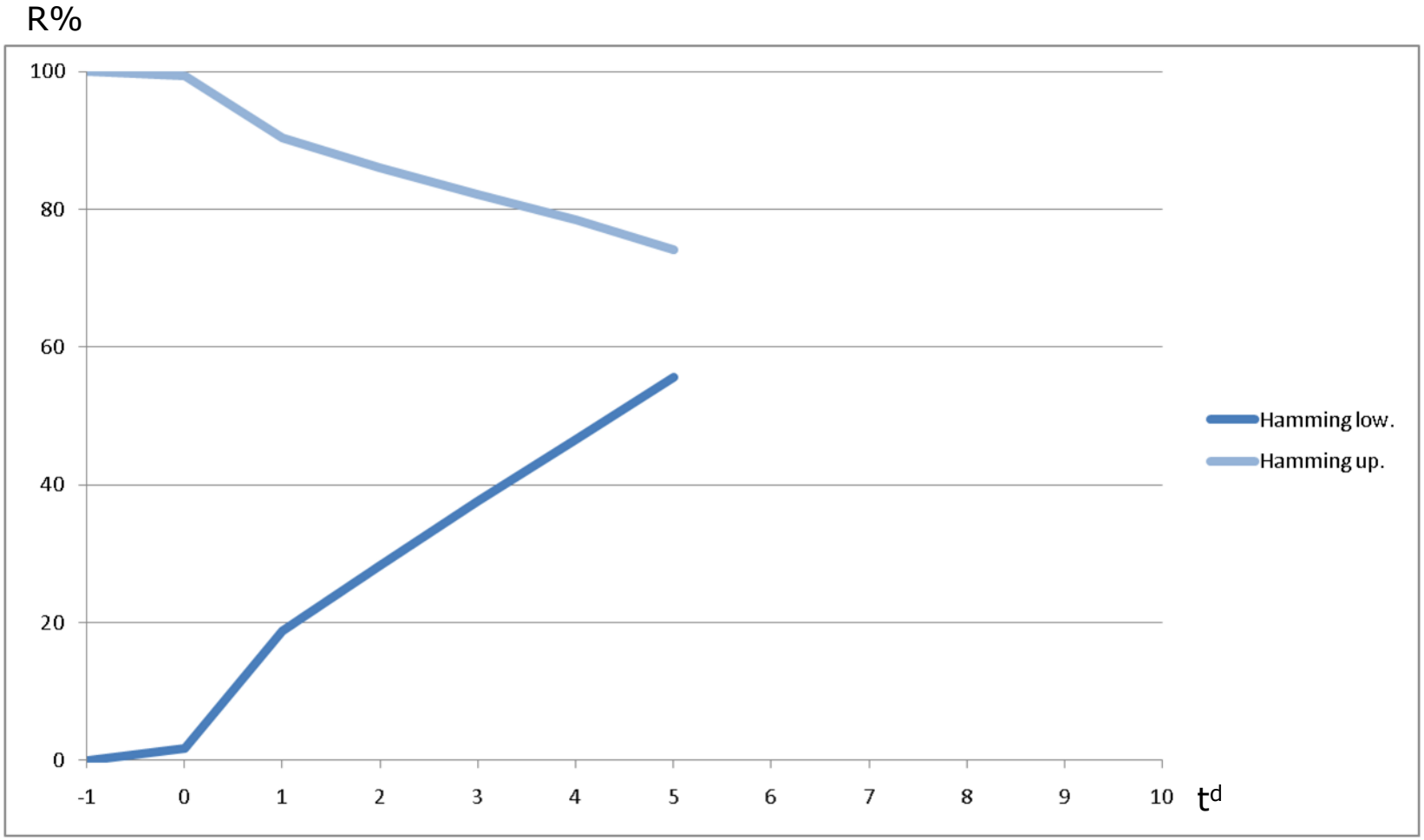
Bounds in Practice



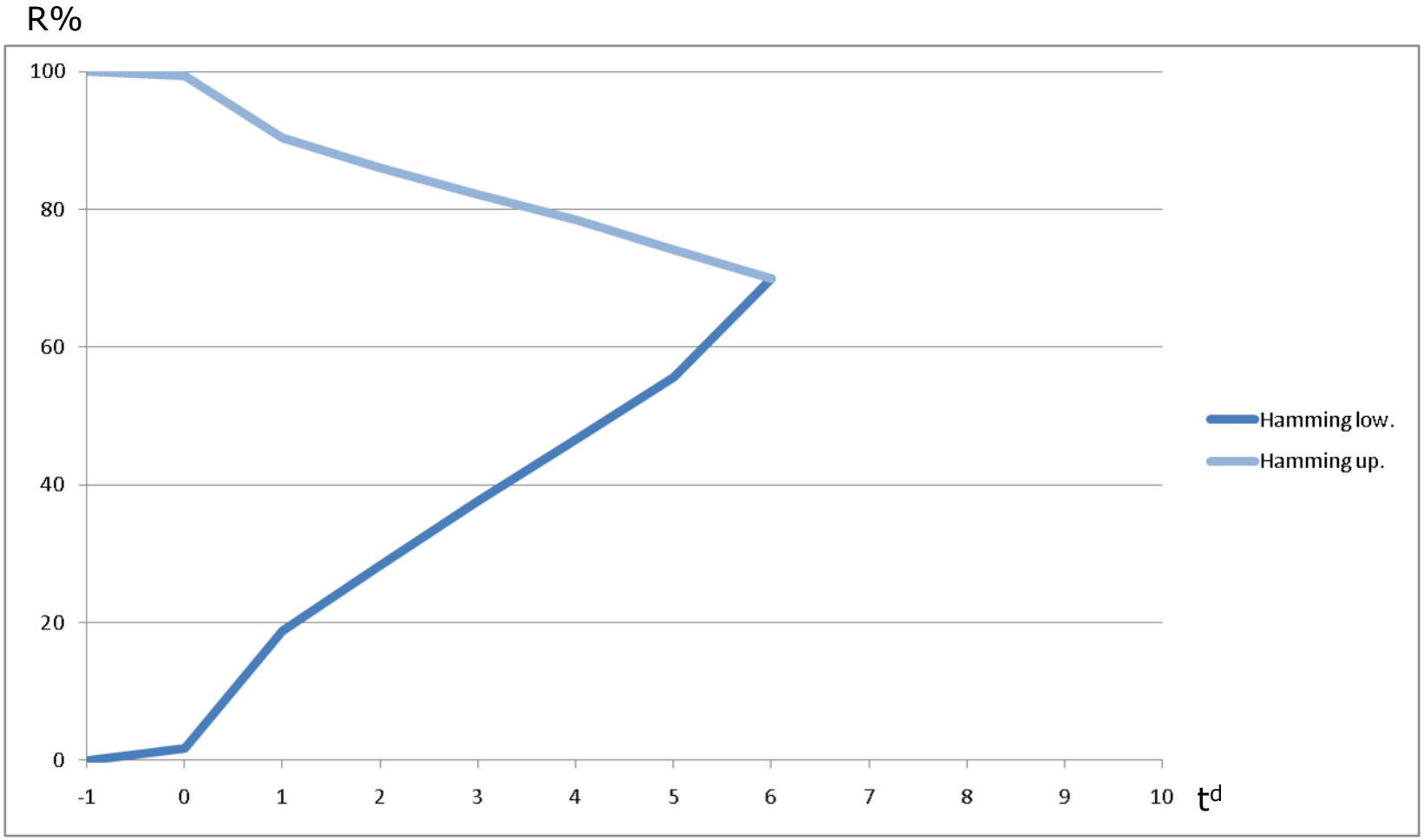
Bounds in Practice



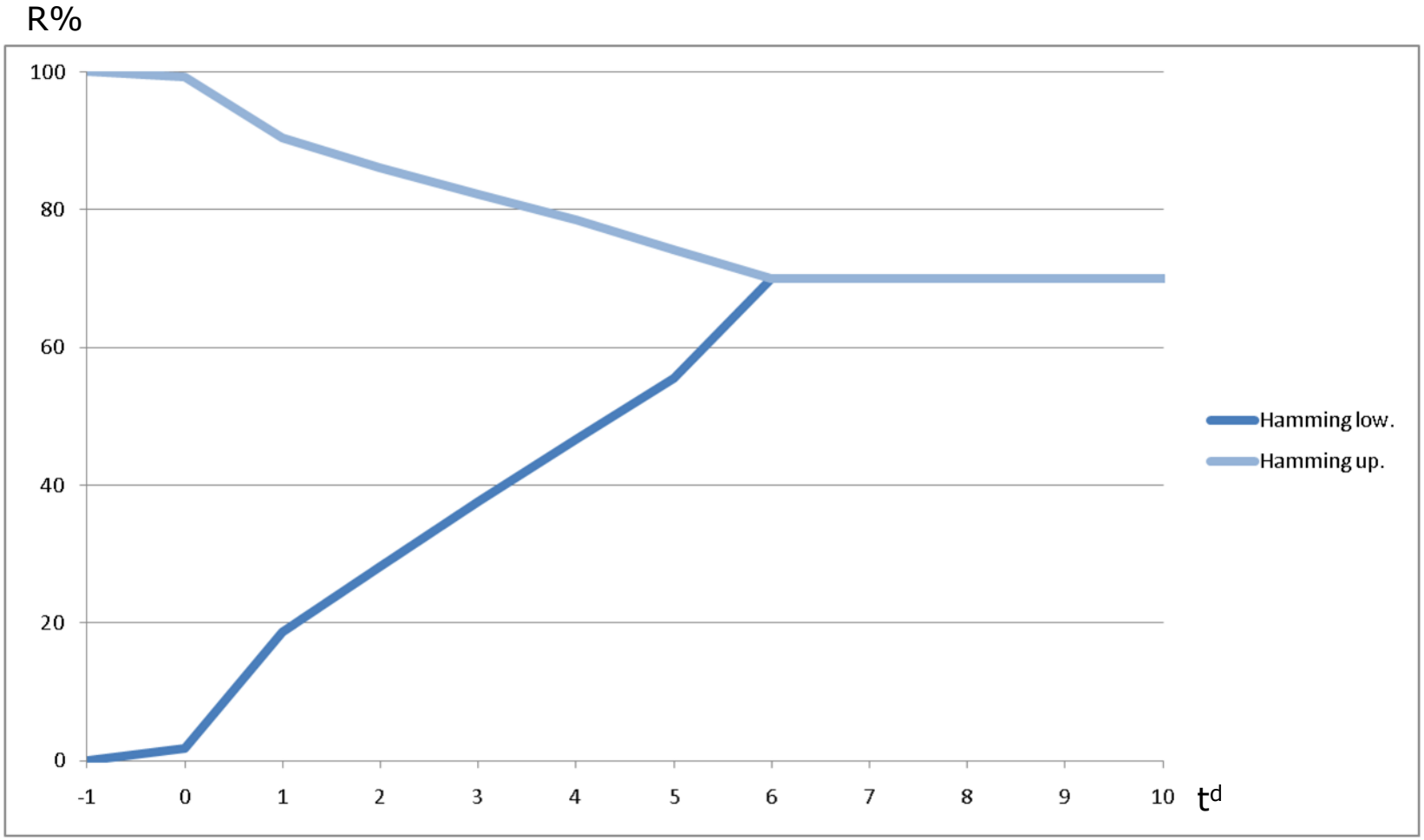
Bounds in Practice



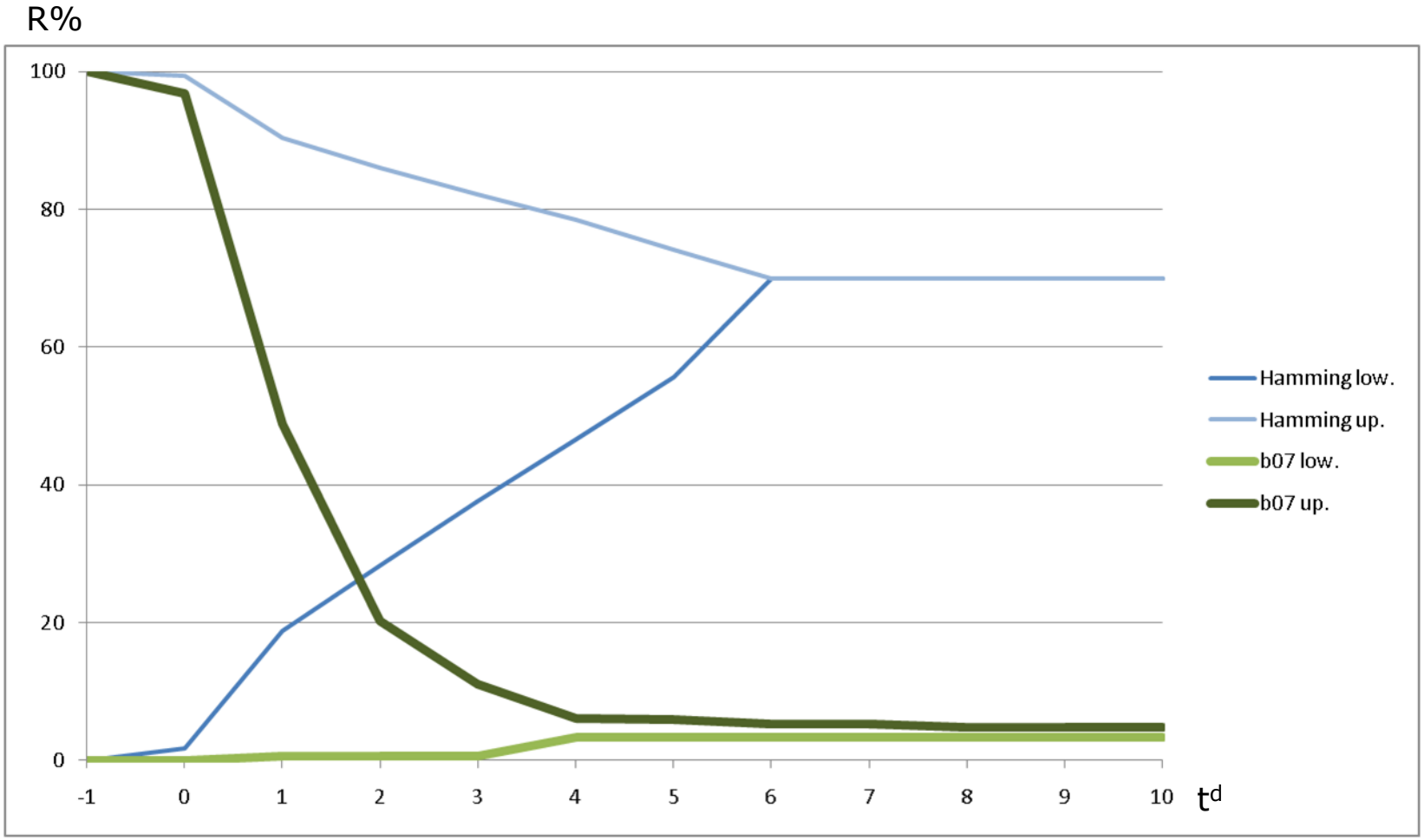
Bounds in Practice



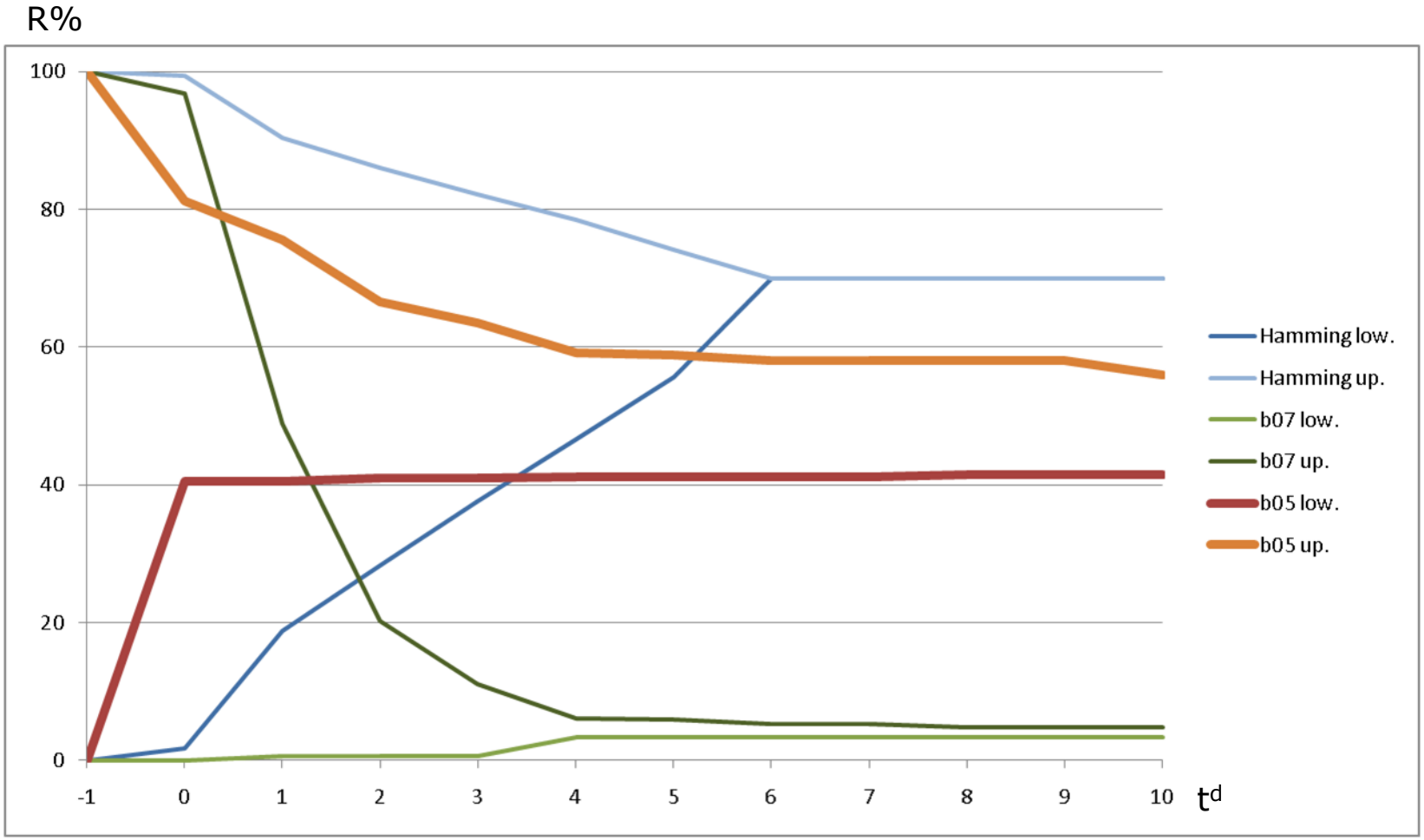
Bounds in Practice



Bounds in Practice

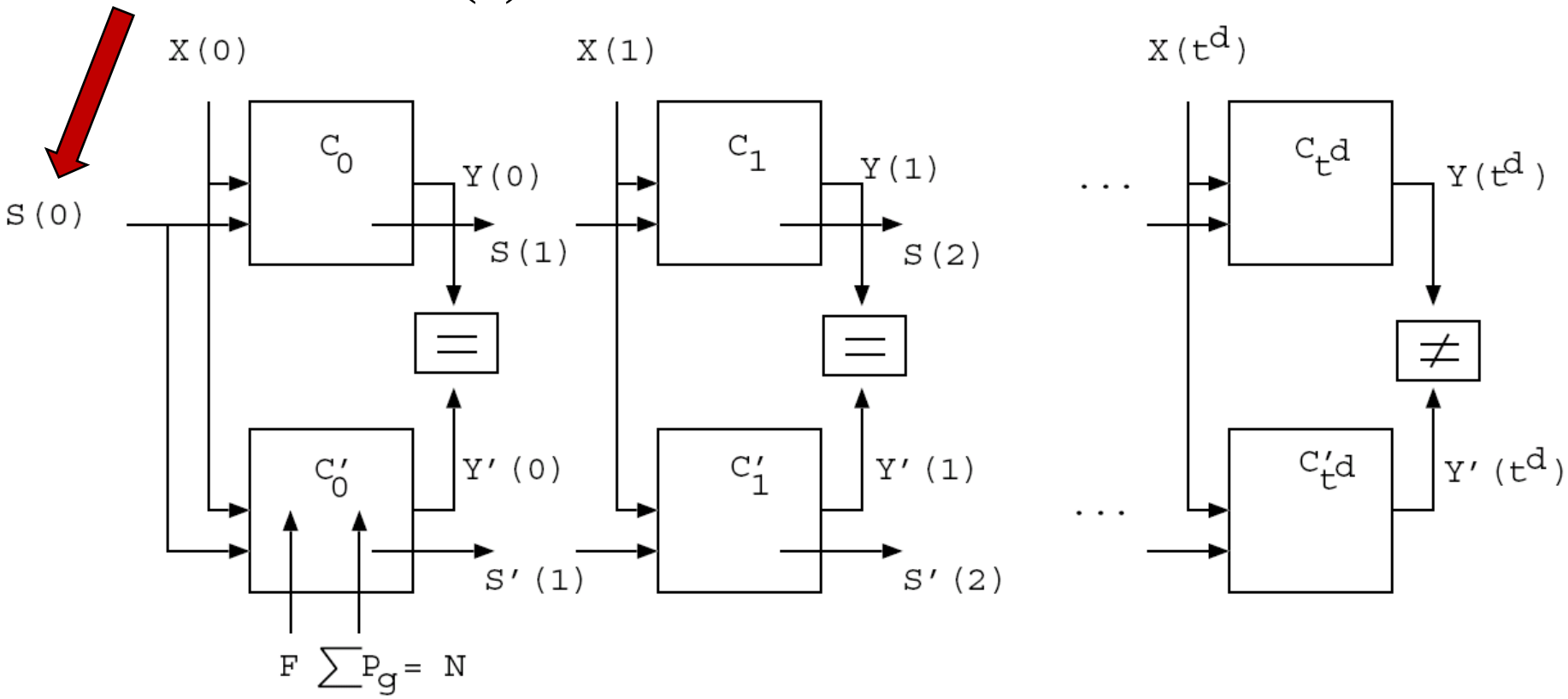


Bounds in Practice



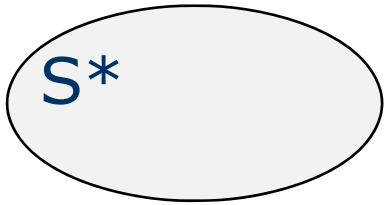
Initial State

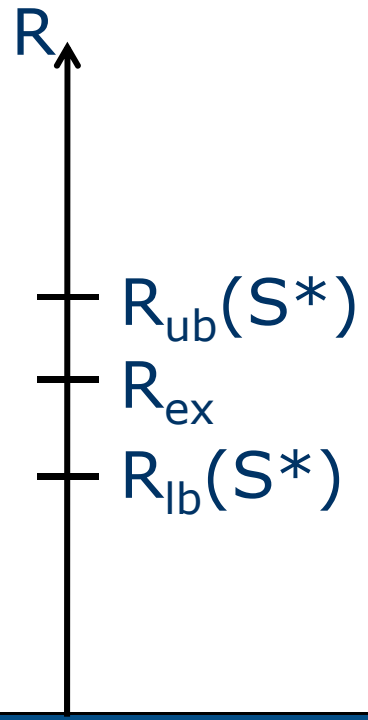
Restrictions on S(0) influence result



Approximate Reachability (1)

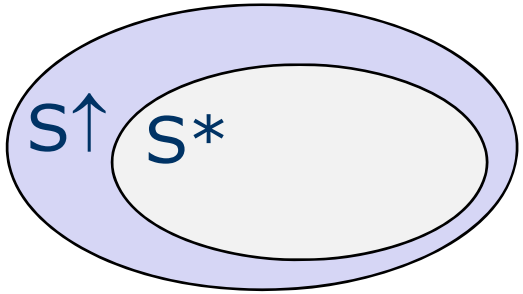
- S(0) = S* = Set of reachable states
  - Exact results for robustness
  - Reachability is expensive



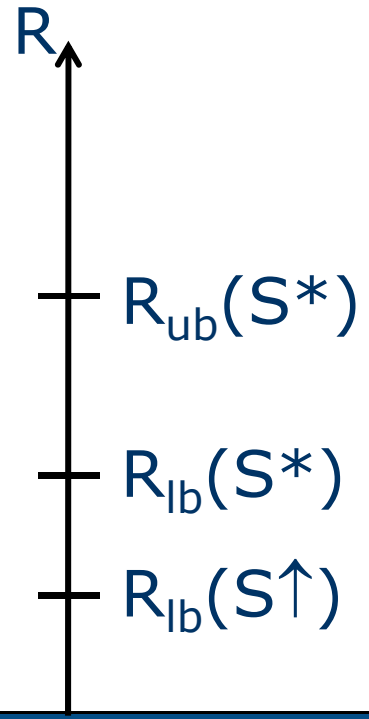


Approximate Reachability (2)

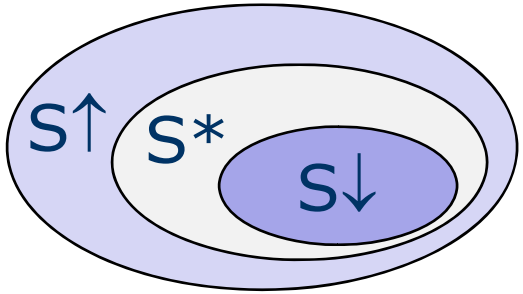
- S(0) = S* = Set of reachable states



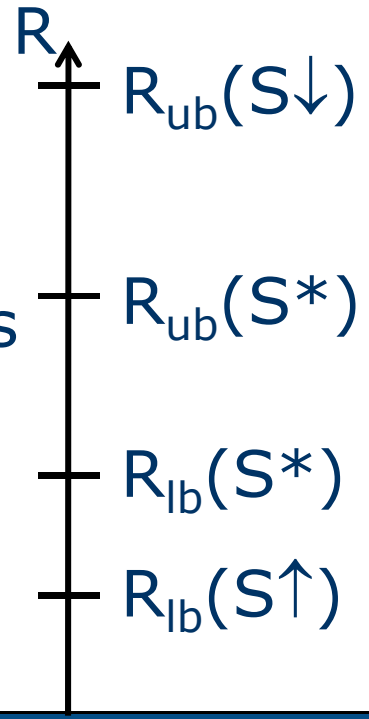
- $S\uparrow \supseteq S^*$ yields $R_{lb}(S\uparrow) \leq R_{lb}(S^*)$
  - Overapproximation – too many states, even faulty states
  - Classifies robust components as non-robust
  - Example: TMR-submodules in different states



Approximate Reachability (3)



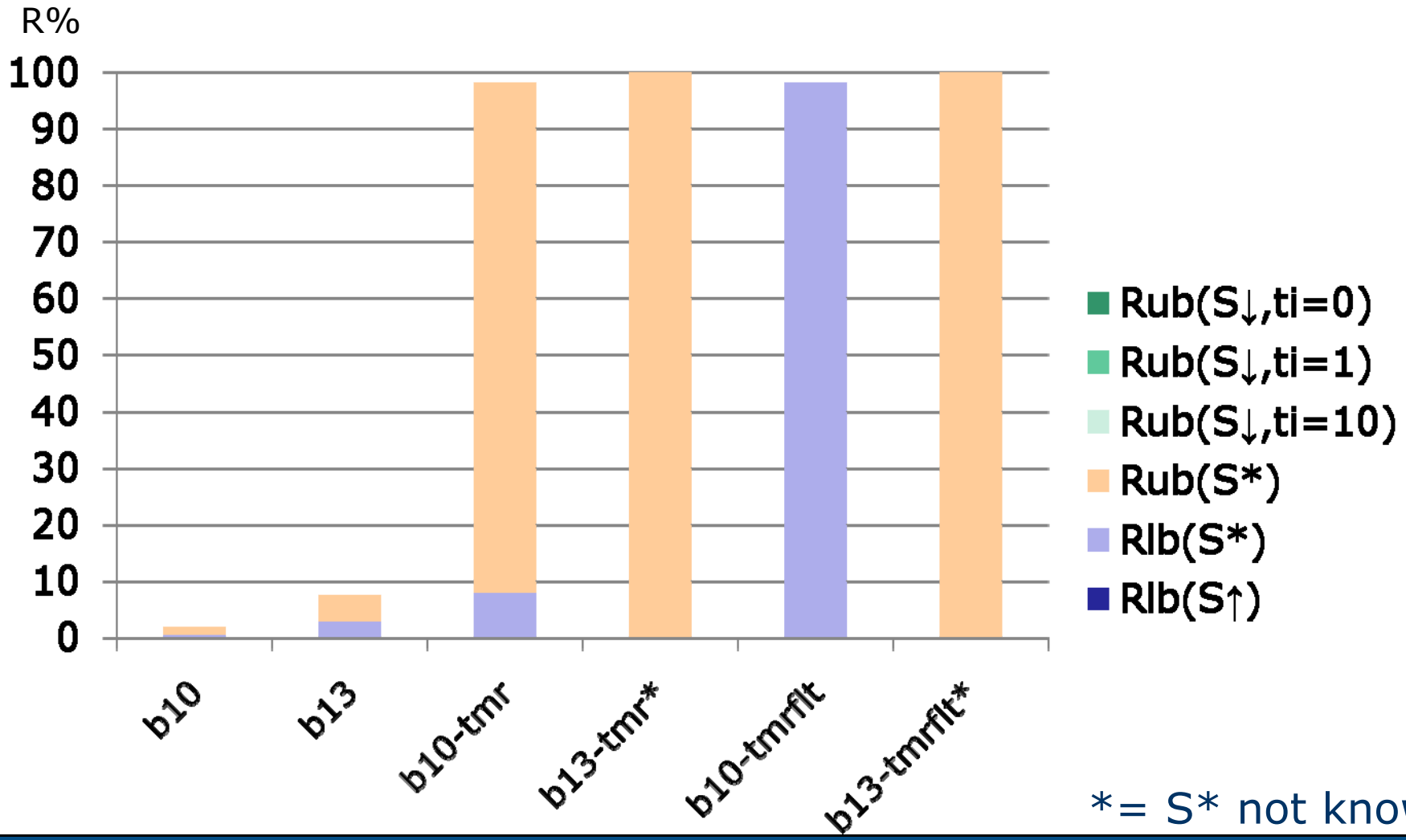
- S(0) = S* = Set of reachable states
- $S\uparrow \supseteq S^*$ yields $R_{lb}(S\uparrow) \leq R_{lb}(S^*)$
- $S\downarrow \subseteq S^*$ yields $R_{ub}(S\downarrow) \geq R_{ub}(S^*)$
  - Underapproximation – excludes states
  - Classifies non-robust components as robust
  - Example: ALU where ADD-instruction is not considered



Experimental Results

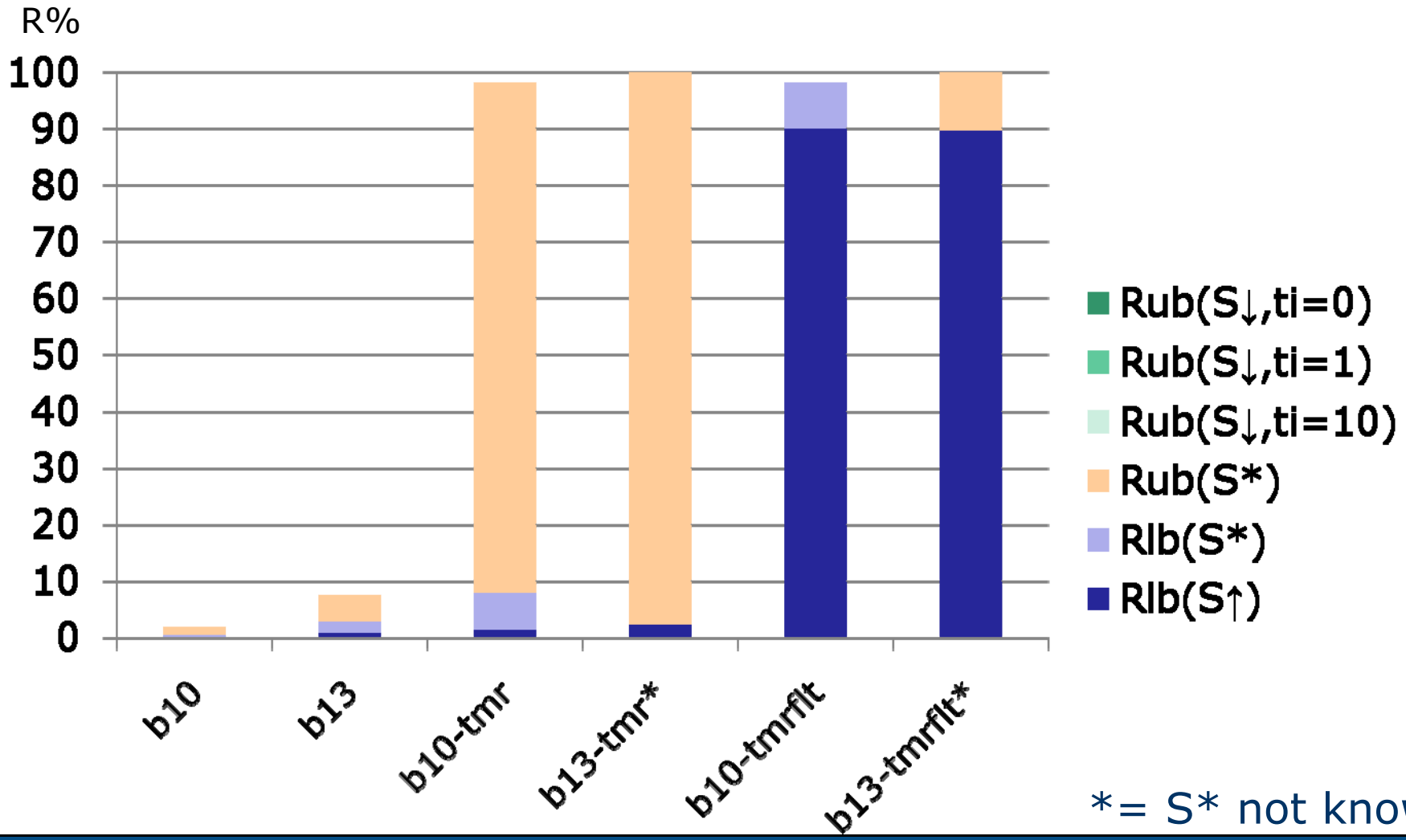
- ITC'99 benchmark circuits
  - Plain, TMR, TMR with fault detection
  - BDD-based reachability or SAT-based approximation
- Run time
  - Quite high (similar to seq. EC)
  - Up to 11 times improvement by low level techniques (dominators, incremental SAT)
  - ATPG approach does not yield benefit (on the benchmarks)

Experimental Results



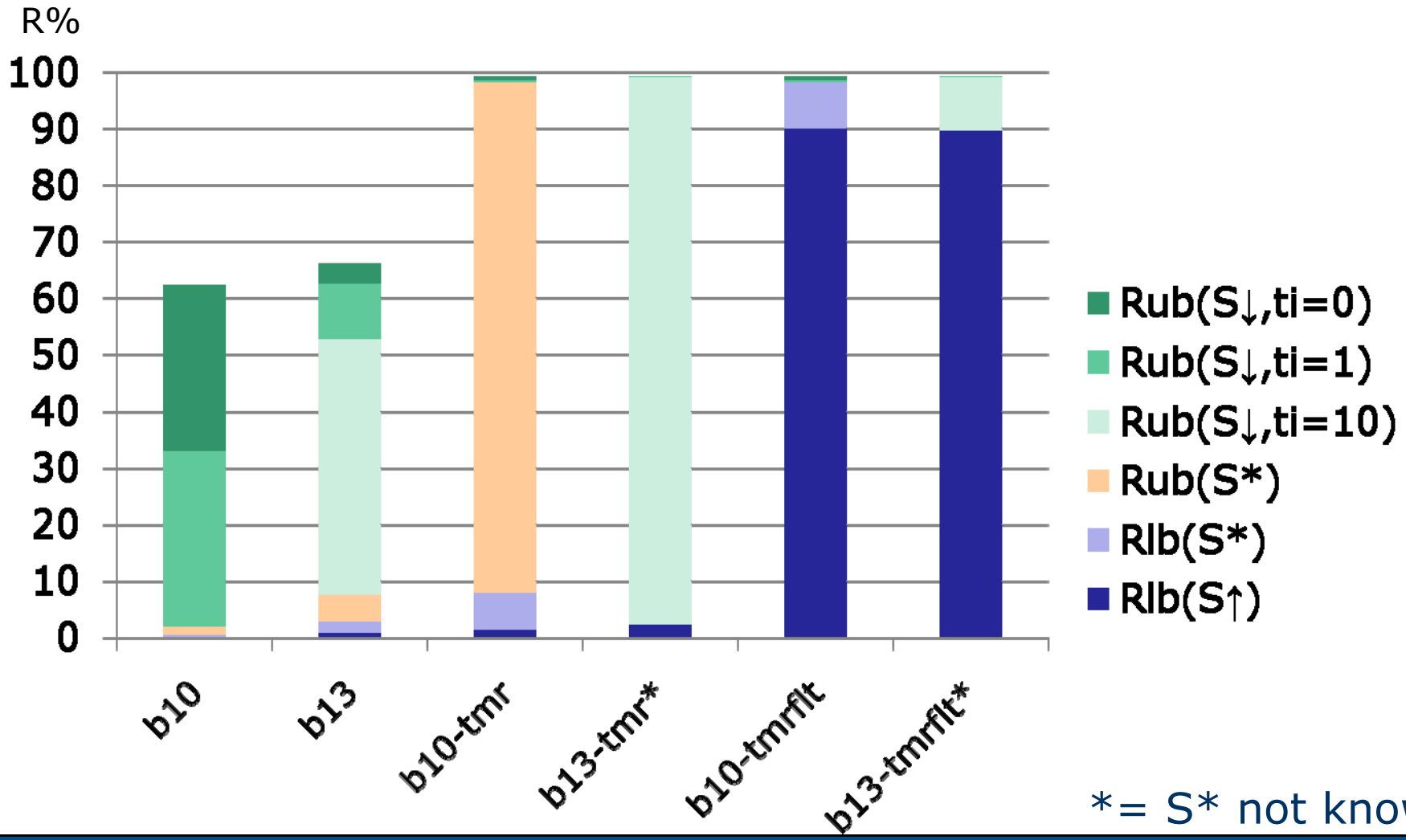
= S not known

Experimental Results



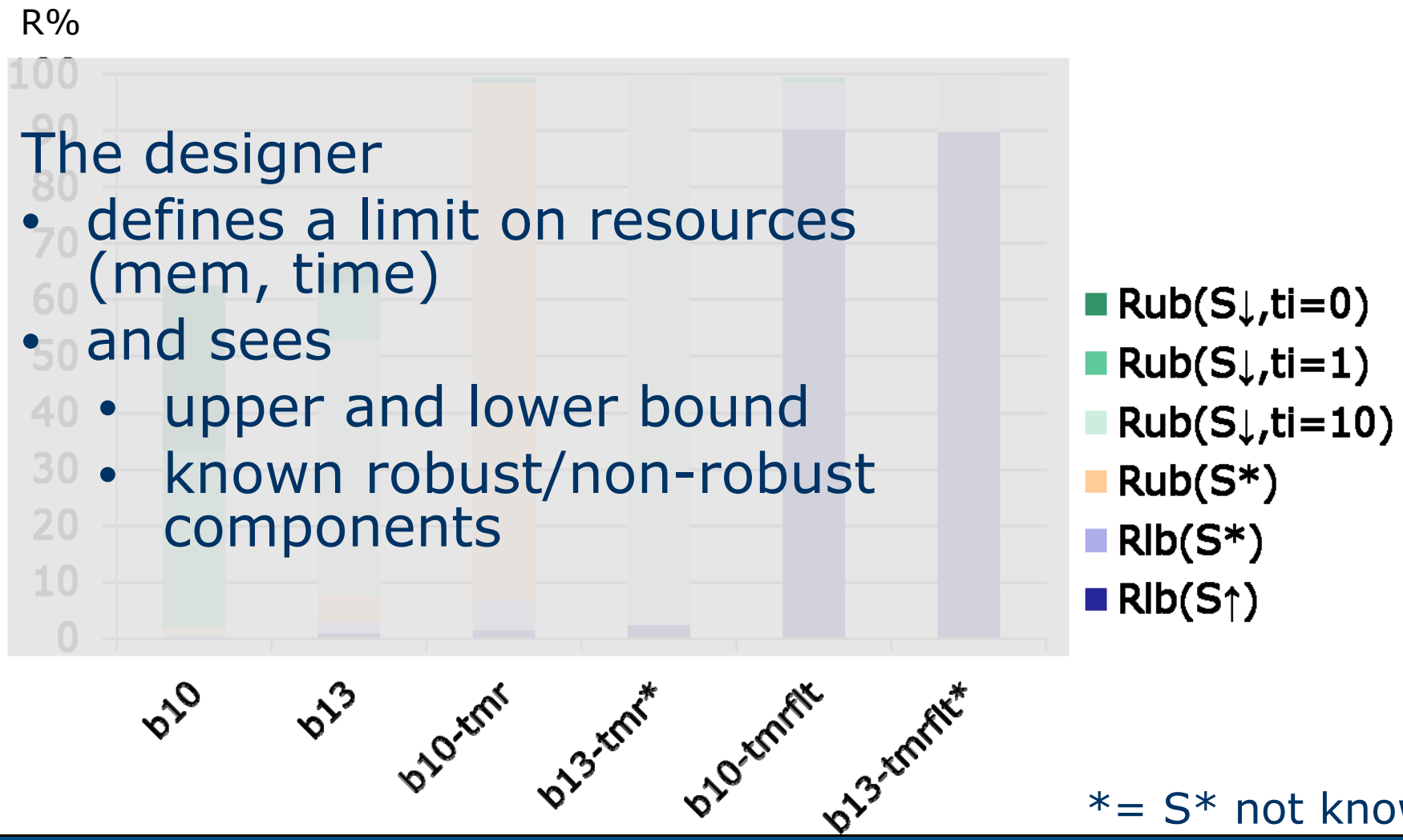
= S not known

Experimental Results



= S not known

Experimental Results



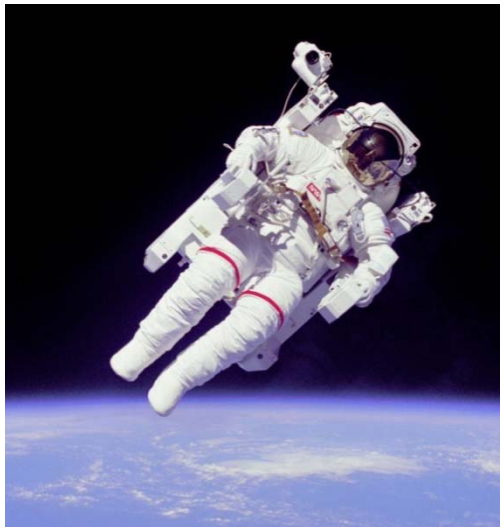
The designer

- defines a limit on resources (mem, time)
- and sees
  - upper and lower bound
  - known robust/non-robust components

= S not known

What about the Environment?

- Safe!
- Expensive?



NASA/courtesy of nasaimages.org

By Drift Words, flickr.com

- Unsafe?
- Cheap!

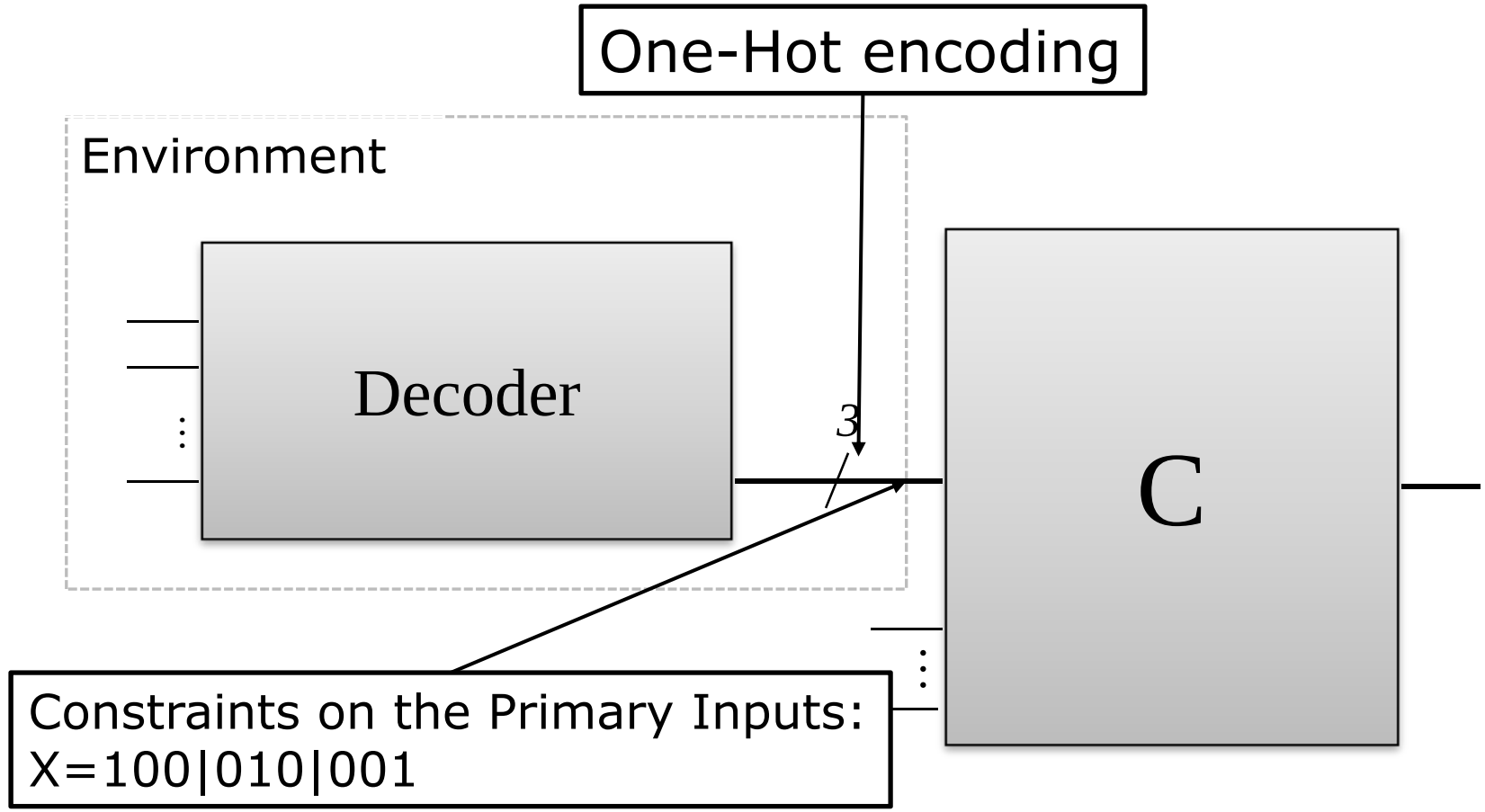
What about the Environment?

- Circuits are integrated in an environment
- Environment significantly influences required level of robustness
- Constraints for primary inputs and states

Questions:

- How to obtain?
- How to consider?
- How large is the influence?

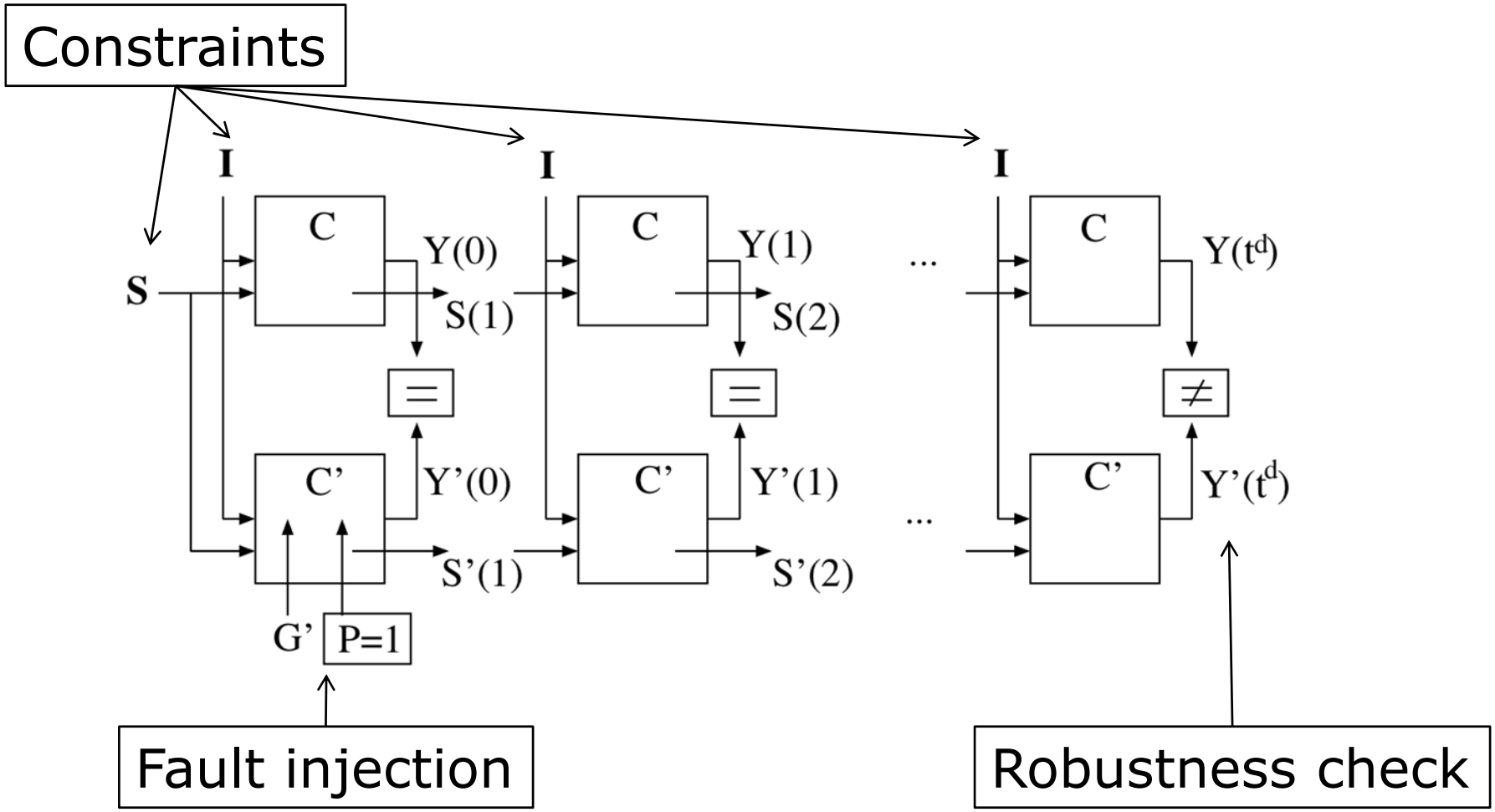
Environment Example



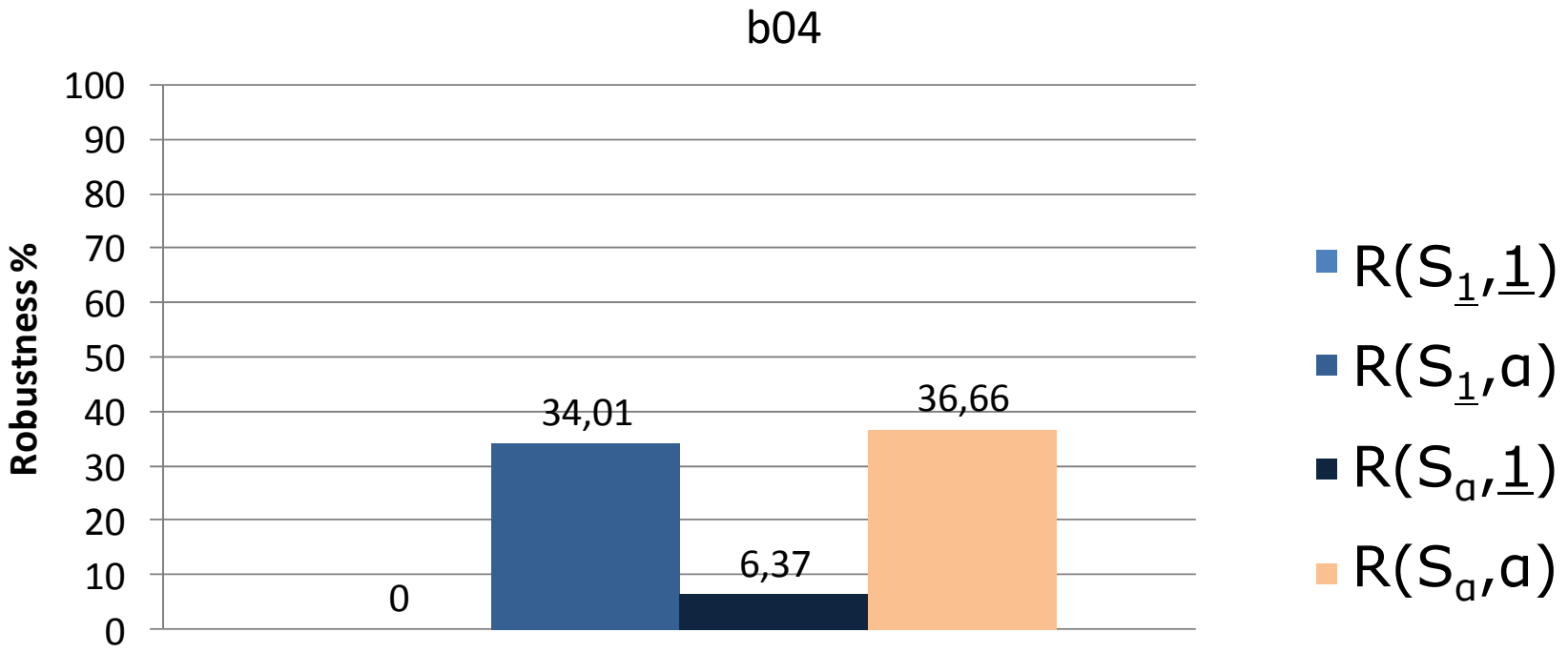
Obtaining Constraints

- Exact data are hard to obtain
- Over-approximation is required to avoid pruning
- But, in practice under-approximation:
  - Manual invariants
  - Testbench
  - In-field data

Experimental Setup - Model



Robustness



→ Robustness significantly influenced by constraints

Summary

- Automated analysis based on BMC
  - Fault model
  - Reachability
- Add-ons
  - Environment
  - Multiple faults
  - Towards probabilistic analysis

Future Work

- Efficiency
  - Hierarchical analysis
  - Abstraction
- Applicability
  - Compositional analysis
  - Real world examples

Acknowledgement

- Partially funded by DFG

- Together with
  - Rolf Drechsler
  - Stefan Frehse
  - André Sülflow