The ProB Validation Tool

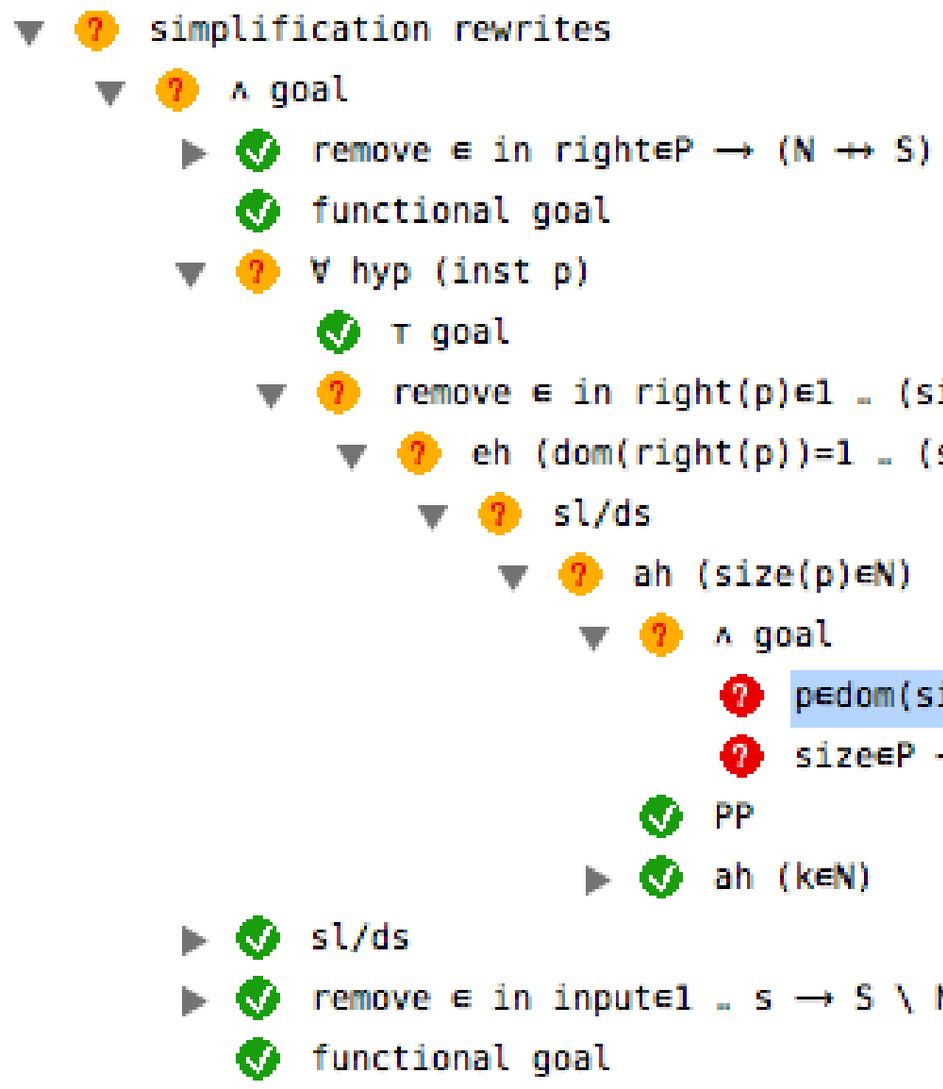
Michael Leuschel

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

ProB in 1 slide

- **Flexible** & Extensible **Validation** Tool for **High-level** Specification Formalisms
  - Multiple Languages:
    - B, Z, CSP, Event-B, Promela, dSL, ...
  - Multiple Validation Technologies derived from operational semantics:
    - Animation, Model Checking, Refinement Checking, ...

But we have proofs !?



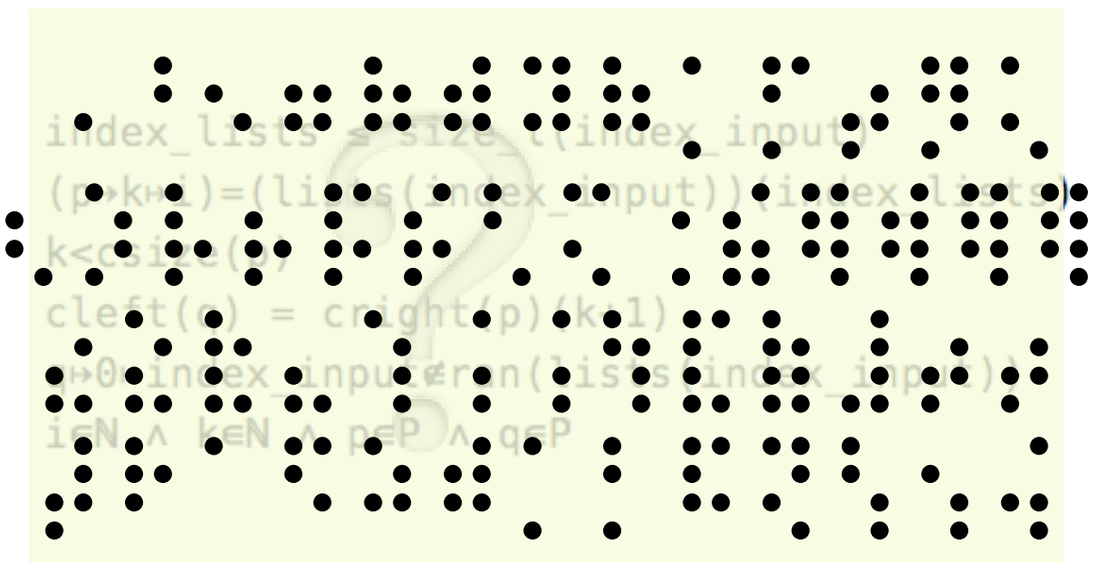
Axioms may be inconsistent

Deadlock

Domain Experts and Managers need to understand the models

Formal Models are hard to understand

What others see:



Summary: Why additional validation

- It is easy to overlook missing/wrong functionality
- Even proven models contain mistakes
- Even FM experts can make mistakes
  - Specifications get more and more complex
- Only Domain Experts can spot certain errors

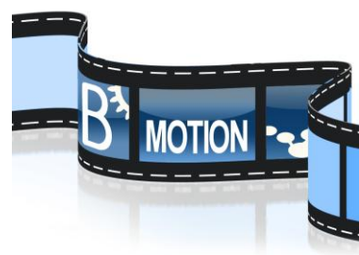
Additional Validation offered by ProB

- Animation: show behaviour of model in clear terms
- Graphical Domain Specific Visualization
- Visualization of State Space
- Model Checking
- Refinement Checking

Some Distinguishing Aspects of ProB

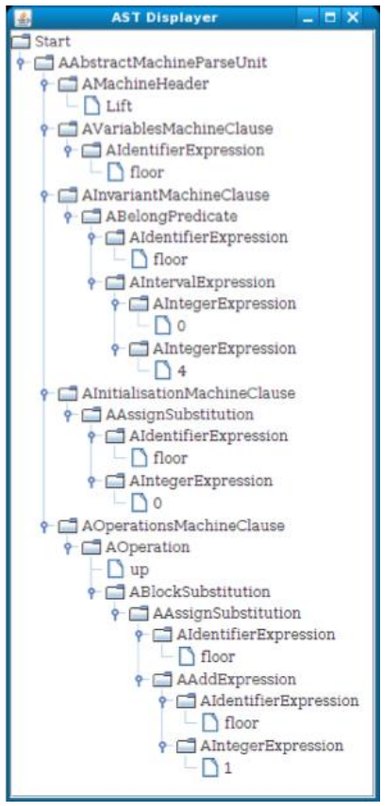
- Symmetry Reduction
- Constraint solving

- Dealing with large Data
- Supports directly high-level formalism
  - Formalisms can be combined (CSP||B)
- Graphical Animation with BMotionStudio



How ?

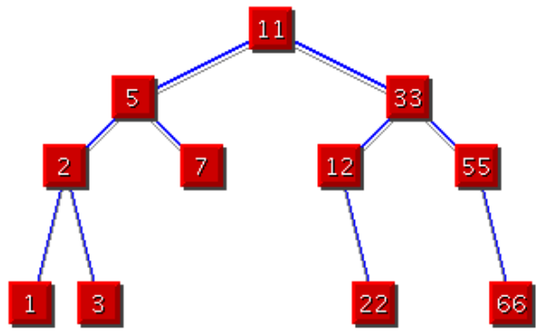
Scaling up ProB



- Uses Constraint Logic Programming
  - Many optimisations
- Handle industrial specifications in ProB:
  - New Parser, Atelier-B compliant
  - New Typechecker (unification-based)
  - Extended Interpreter: almost 100 % support for B

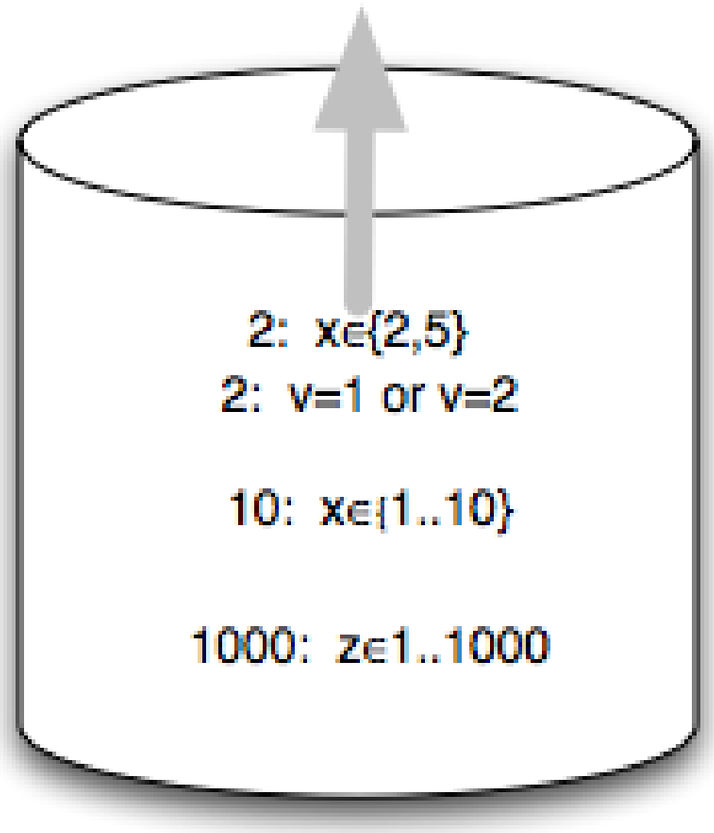
2. New Datastructure for Large Sets & Relations

- Before:
  - Sets represented as Prolog lists [int(1),int(2)] for {1,2}
- Now:
  - self-balancing AVL-Trees used by ProB kernel if possible



3. Improved Constraint Solving

- Heap of Choice Points for enumeration:
  - priority: estimated # of solutions
  - priority 0: gives (1) AVL tree



Priority Queue of Enumerations/Choice Points

dr = ran(ri) & ri =r~ & 3:dom(ri) &
r = **{(1|->1),(6|->2),(7|->3),(8|->4),(9|->5)}**

Could we have used other technologies ?

Proof

SAT

SMT

Sieve Experiment

```
numbers := numbers - ran(%n.(n:cur..limit/cur|cur*n))
```

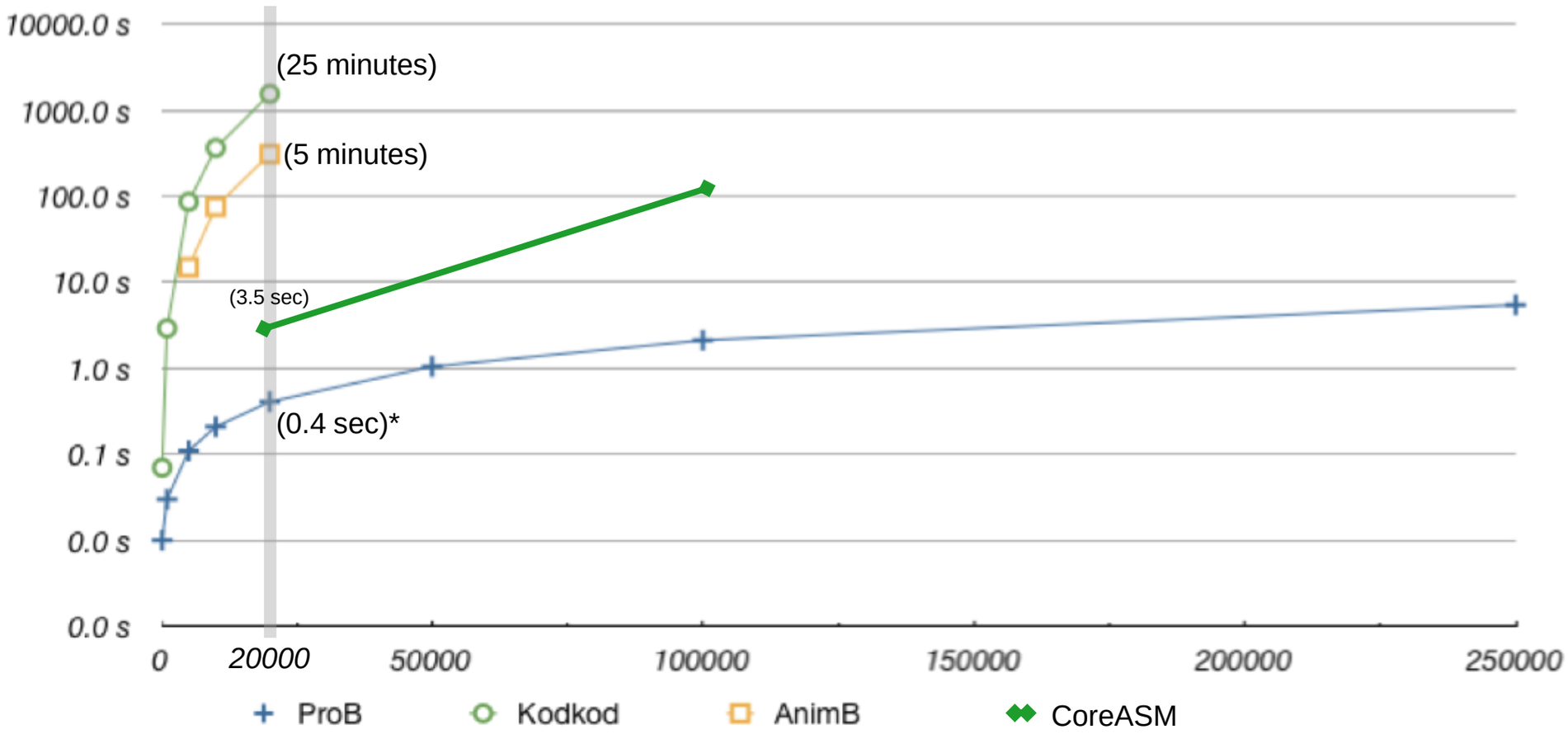
- For cur=2 & limit =
  - 10,000: 0.2 secs

older version:
out of memory after 2 minutes

  - 100,000: 2.1 secs
  - 1,000,000: 21.9 secs
- Could be further optimised

Sieve Experiment

First Step of Sieve Prime Number Algorithm; using a set representation



(cf Alloy)

(similar for Brama)

(thanks to Roozbeh Farahbod)

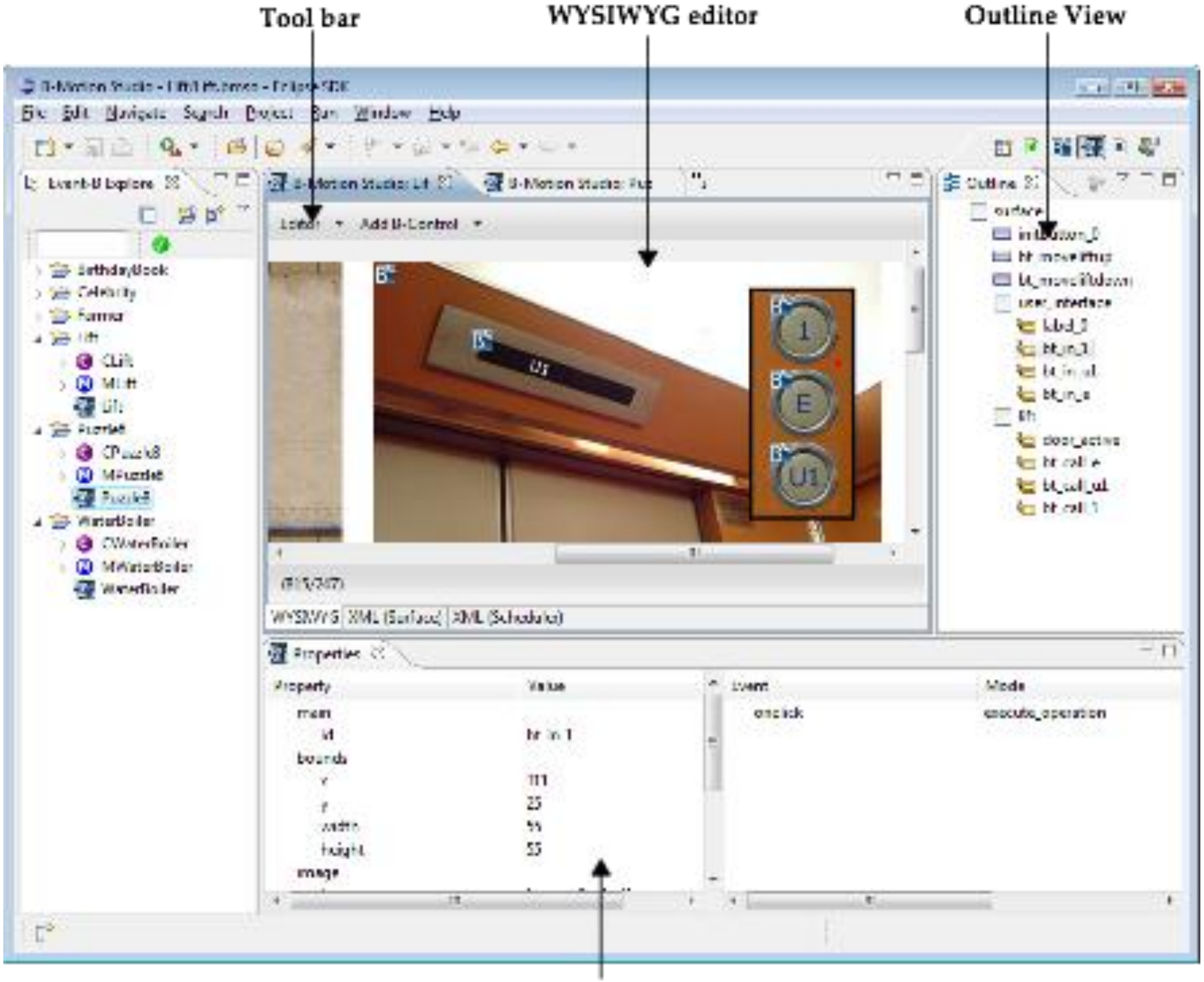
*(0.2 sec with ProB 1.3.2-beta5)

Disclaimer:
Alloy/Kodkod can be much better than ProB at other tasks !

BMotionStudio

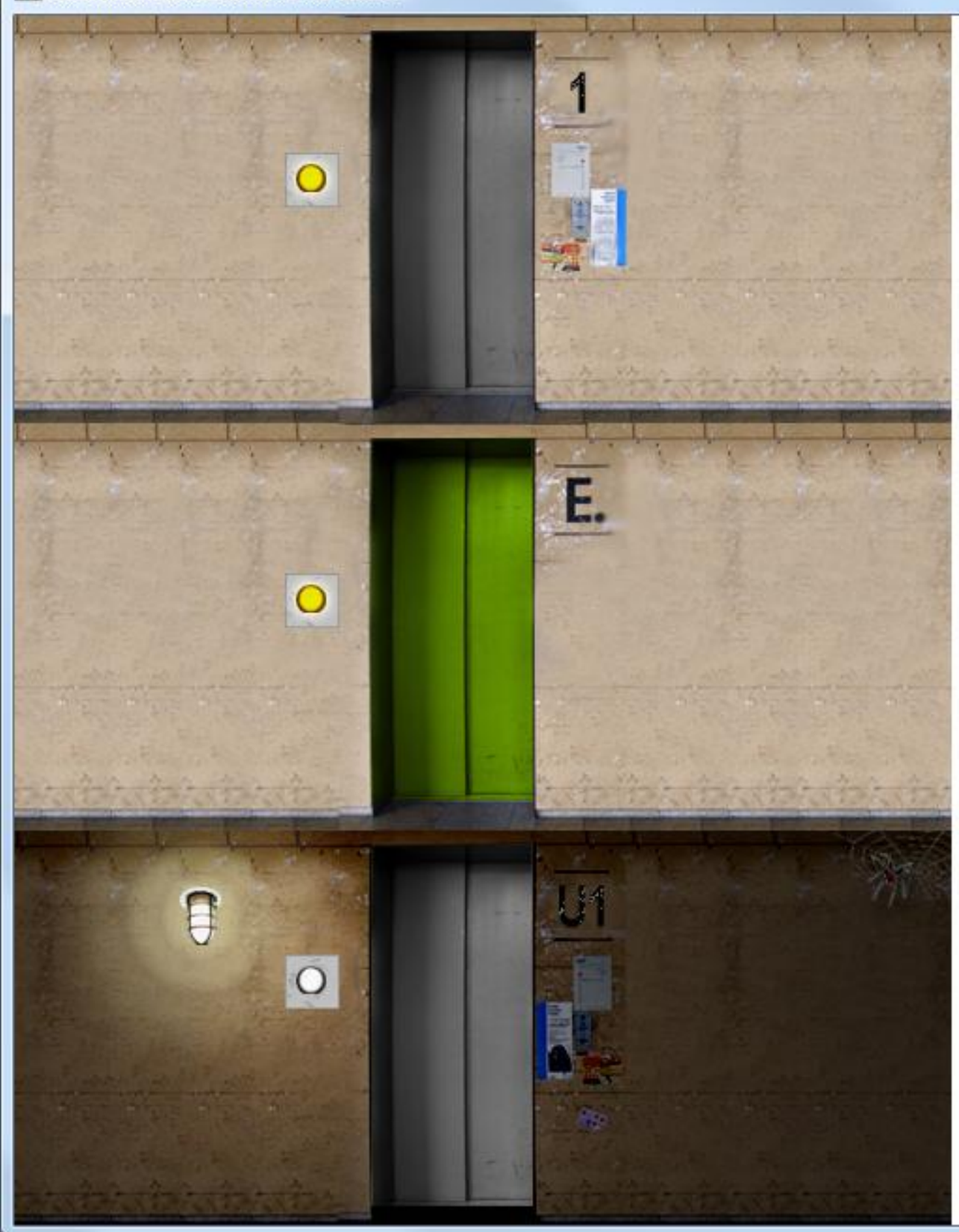
Problem-specific animation

BMotionStudio



[Ladenberger et al., FMICS'09]

Animation for machine: MLift.bum



Move Lift UP

Move Lift DOWN

Initialize machine

Thanks !

- Jens Bendisposto
- Carl Friedrich Bolz
- Nadine Elbeshausen
- Fabian Fritz
- Marc Fontaine
- Michael Jastram
- Li Luo
- Daniel Plagge
- Mireille Samia
- Corinna Spermann
- Dennis Winter

- Michael Butler
- Thierry Massart
- Edd Turner

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF