Universität Rostock Traditio et Innovatio

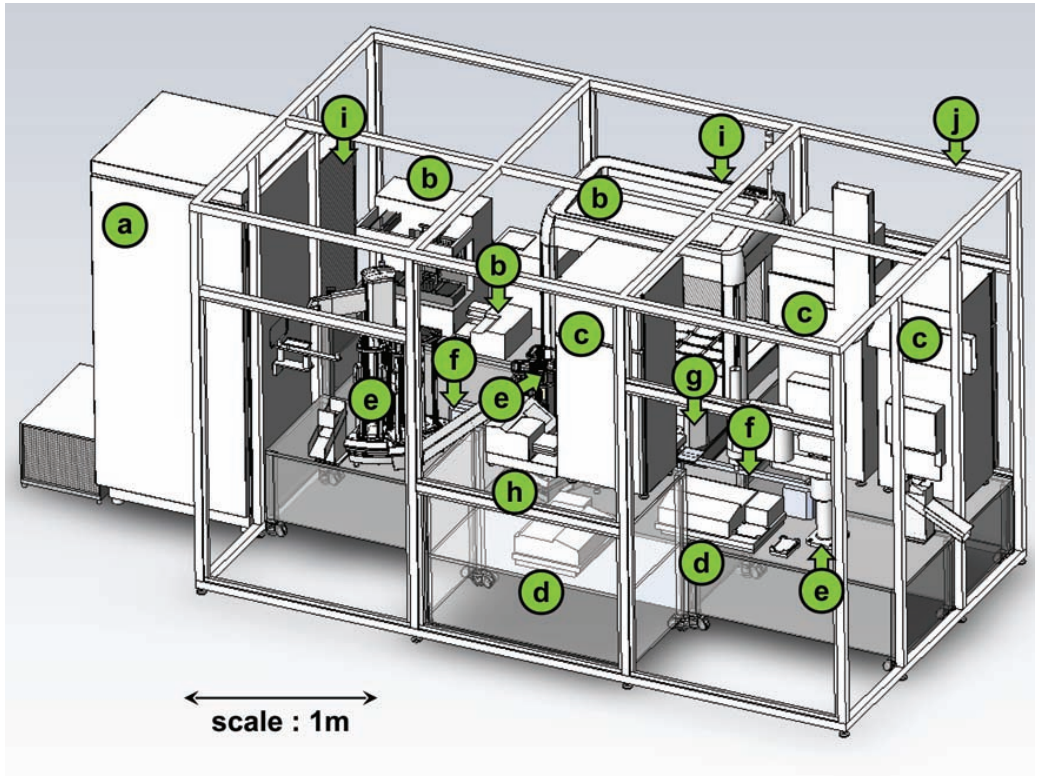
Using AI Planning to Automate the Performance Analysis of Simulators

19. 3. 2014, SIMUTools 2014 — Lisbon, Portugal

Roland Ewald

Sponsored by:

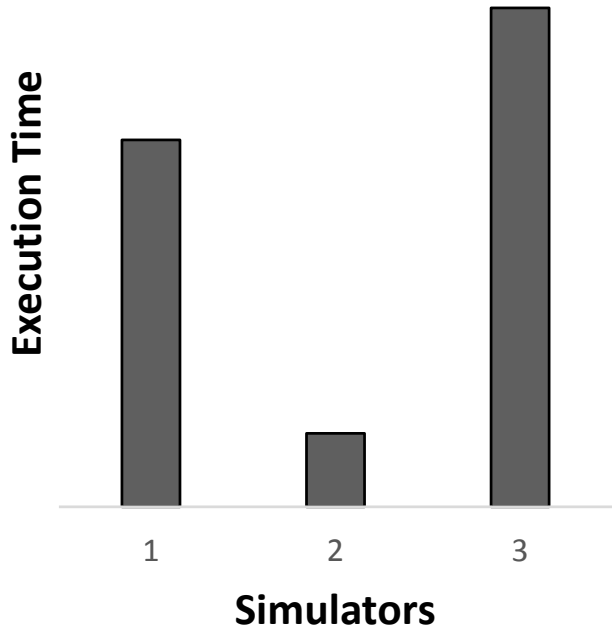
DFG Deutsche Forschungsgemeinschaft



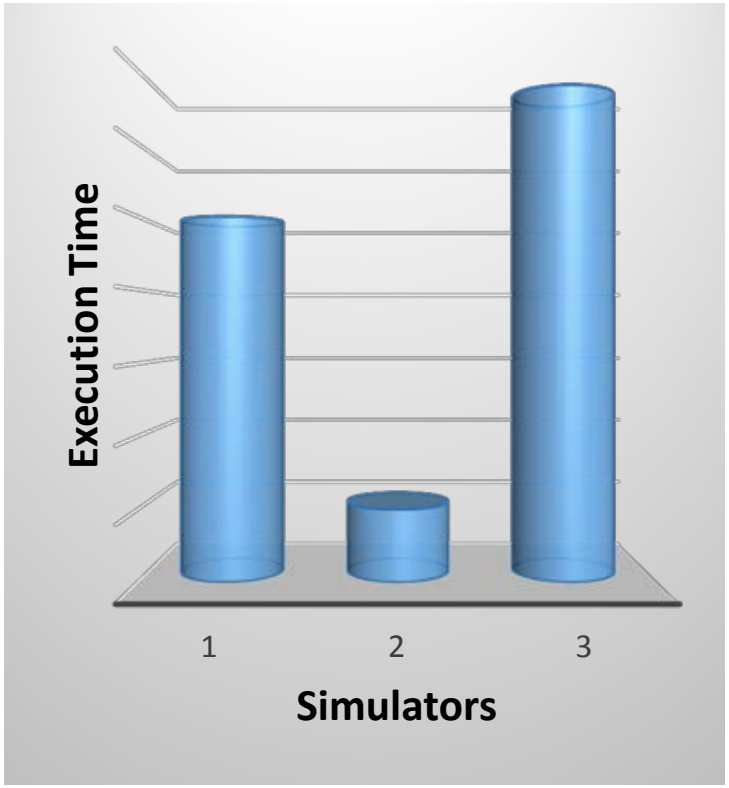
Left: Victorian era student microscope, by Tom Blackwell (flickr.com, CC-BY-NC)
*Right: King et al., **The automation of science**, Science 324 (5923), pp. 85-89, **Apr. 2009**.*

What about Simulator Performance Analysis?

1989



2014



The Problem

- Manual analysis is cumbersome (less exploration, more errors)
- Constant reinvention of the wheel (scripts, statistics, design)
- Implicit, undetected bias (little reproducibility, credibility, efficiency)

The Problem

- Manual analysis is cumbersome (less exploration, more errors)
- Constant reinvention of the wheel (scripts, statistics, design)
- Implicit, undetected bias (little reproducibility, credibility, efficiency)

A performance study has context & is done for a reason.
Let the user express this, automate everything else.

Scenario: Comparing two Simulators

Hypothesis:

$$\forall m \in \mathcal{M} : moreEntities(m, x_{hyp}) \Rightarrow faster(B, A, m)$$

- A, B: simulators
- $\mathcal{M}$: models (problem space)
- $moreEntities$, $faster$: predicates
- x_{hyp}: hypothetical threshold

Non-deterministic Outcomes

$$\forall m \in \mathcal{M} : moreEntities(m, x_{hyp}) \Rightarrow faster(B, A, m)$$

1. B outperforms A: $faster(B, A, m) \wedge \neg faster(A, B, m)$
2. A outperforms B: $faster(A, B, m) \wedge \neg faster(B, A, m)$
3. A *and* B perform similarly: $\neg faster(A, B, m) \wedge \neg faster(B, A, m)$
4. A, B, or both crashed.
5. Another error occurred.

Falsification Approaches

$$\forall m \in \mathcal{M} : moreEntities(m, x_{hyp}) \Rightarrow faster(B, A, m)$$

1. Randomly sample $m \in \mathcal{M}$ (computing budget allocation?)
2. Statistical tests (runtime distribution?)
3. Train performance predictors, try only promising $m \in \mathcal{M}$
4. Analyze sensitivity of A and B performance, sample accordingly

Falsification Approaches

$$\forall m \in \mathcal{M} : moreEntities(m, x_{hyp}) \Rightarrow faster(B, A, m)$$

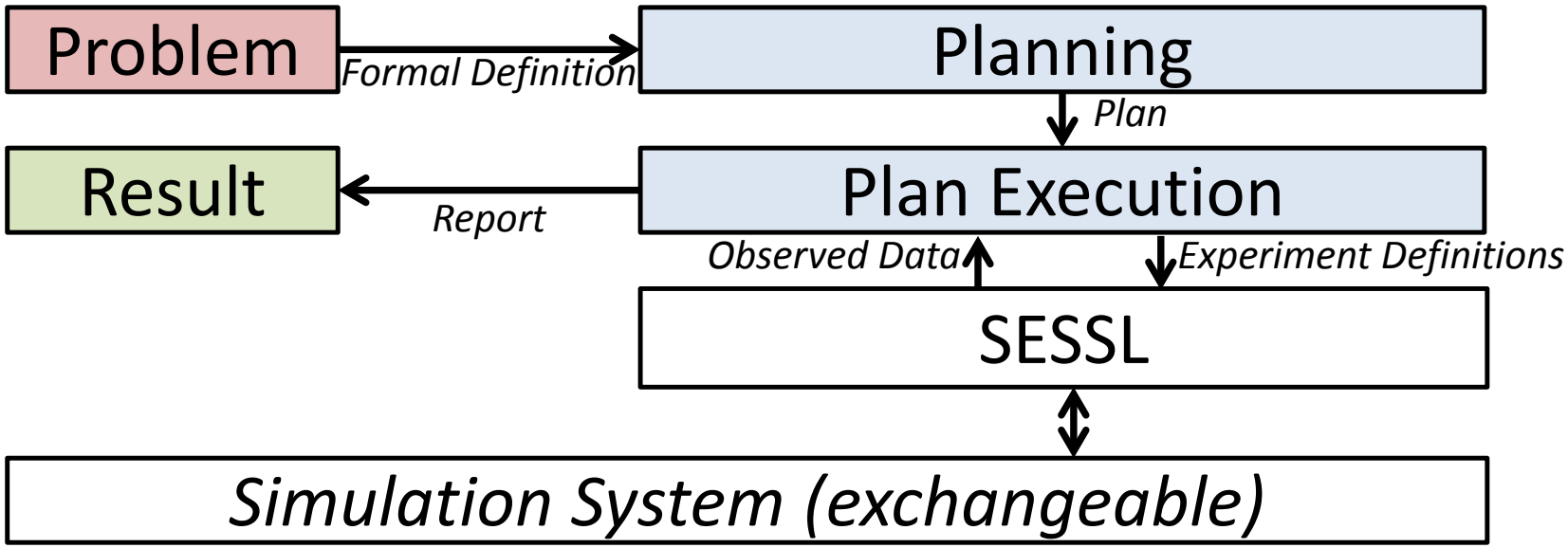
1. Randomly sample $m \in \mathcal{M}$ (computing budget allocation?)
2. Statistical tests (runtime distribution?)
3. Train performance predictors, try only promising $m \in \mathcal{M}$
4. Analyze sensitivity of A and B performance, sample accordingly
5. ... or other **experiment sequences**.

ALESIA

A system for automatic simulator performance analysis.

- Hypothesis-driven experimentation
- Support for performance predictors (knowledge representation)
- Automatic result analysis $\Rightarrow$ feedback loop
- Integration of methods from statistics, OR, AI etc.

ALESIA: Overall Structure



Sample Input

1. User domain (context)
2. Preferences
3. Goal (¬hypothesis)

```
val result = submit {
  SingleModel("java://examples.sr.LinearChainSystem")
} {
  TerminateWhen(WallClockTimeMaximum(seconds = 30))
} {
  exists >> model | hasProperty("qss")
}
```

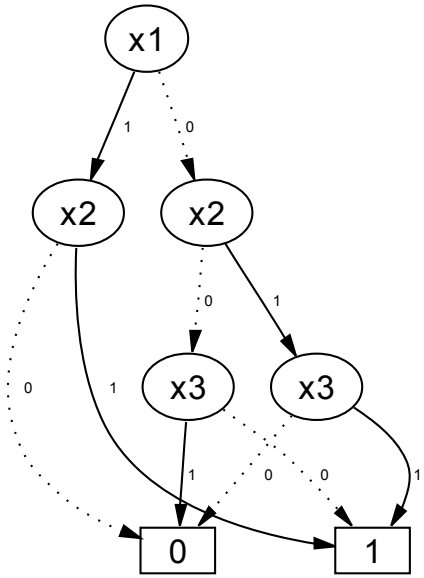
Sample Actions

- Generated by `ActionSpecification` instances
- Depend on user domain
- Non-deterministic

Action *loadSingleModel*:
precondition: $\neg depleted \wedge \neg loadedModel$
effect: $depleted \vee loadedModel$

Action *checkQSSProperty*:
precondition: $loadedModel$
effect: $hasProperty(qss) \vee (\neg hasProperty(qss) \wedge \neg loadedModel)$

Non-Deterministic AI Planning



(from Wikipedia, cc-by-sa)

- Non-determinism: Plan $\Rightarrow$ Policy
- Symbolic Model Checking
- Binary Decision Diagrams (BDDs)
- Algorithms by Cimatti et al., Rintanen

Planning algorithms:

Cimatti et al.: Weak, strong, and strong cyclic planning via symbolic model checking, Artificial Intelligence, 147(1-2), 2003
Rintanen: Complexity of conditional planning under partial observability and infinite executions, European Conference on AI, 2012
Ghallab, Nau, Traverso: Automated Planning: Theory & Practice, 2004

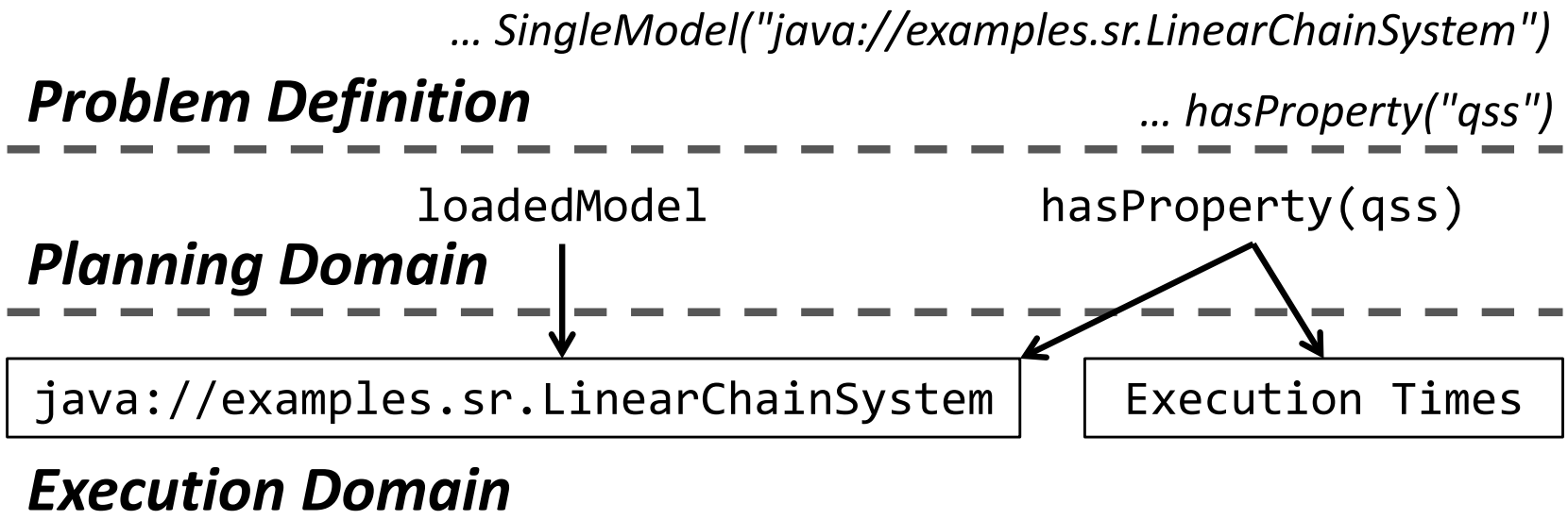
Sample Plan for Goal *hasProperty*(*qss*)

Plan types:

- Weak
- Strong-cyclic
- Strong

```
if(¬depleted ∧ ¬loadedModel)
    use loadSingleModel
else if(loadedModel)
    use checkQSSProperty
else error
```

Plan Execution



Simulation System Independence via SESSL

```
import sessl._
import sessl.james._ //ALeSiA is used with the JAMES II binding

execute{
  new Experiment {
    model="java://examples.sr.LinearChainSystem"
    replications=10
    stopCondition=AfterWallClockTime(seconds=3) or AfterSimSteps(10e06)
  }
}
```

Ewald, Uhrmacher: SESSL: A Domain-Specific Language for Simulation Experiments, ACM TOMACS 24(2), 2014

Example: Simulators for Chemical Reaction Networks

```
val result = submit {
    // Problem Domain (Context):
  ModelSet("java://examples.sr.LinearChainSystem",
   ModelParameter("numOfSpecies",1,10,1000),
   ModelParameter("numOfInitialParticles",10,100,10000)),
  SingleSimulator("nrm", NextReactionMethod()),
  SingleSimulator("dm", DirectMethod())
} { // Preferences:
 DesiredSingleExecutionWallClockTime(seconds = 1),
 TerminateWhen(MaxOverallNumberOfActions(100))
} { // Hypothesis:
exists>>model|(hasProperty("qss") and isFasterWCT("dm", "nrm", model))
}
```

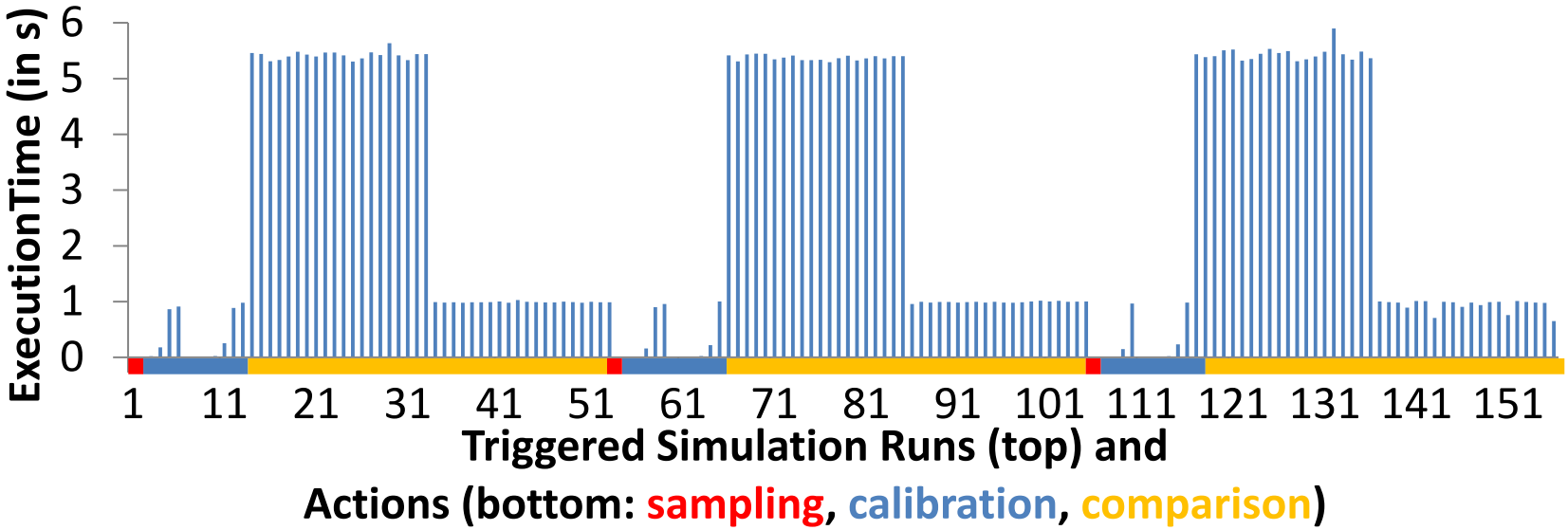
Stochastic Simulation Algorithms:

Gillespie: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, J Comp Phys 22, 1976
Gibson, Bruck: Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels, J Chem Phys 104, 2000

Benchmark Model:

Cao, Li, Petzold: Efficient formulation of the stochastic simulation algorithm for chemically reacting systems, The J Chem Phys 121(9), 2004

Sample Scenario: Execution Trace



Summary

- **Goal:** automate simulator performance analysis
- **Method:** AI planning and plan execution with feedback loop
- **Results:**
  - Works in principle, but action specification requires more meta-data
  - Ongoing: integration of sophisticated methods
  - Future: DSLs for context & hypotheses, distributed execution

Thank you. Questions?

ALESIA is open source (Apache 2.0 license).

`https://bitbucket.org/alesia`