Università degli Studi di Trento

Simulating Midlet's Security Claims with Automata Modulo Theory

Fabio Massacci, Ida Siahan
University of Trento

ACM SIGPLAN Third Workshop on Programming Languages and Analysis for Security (PLAS 2008)
Tucson, Arizona, June 8, 2008

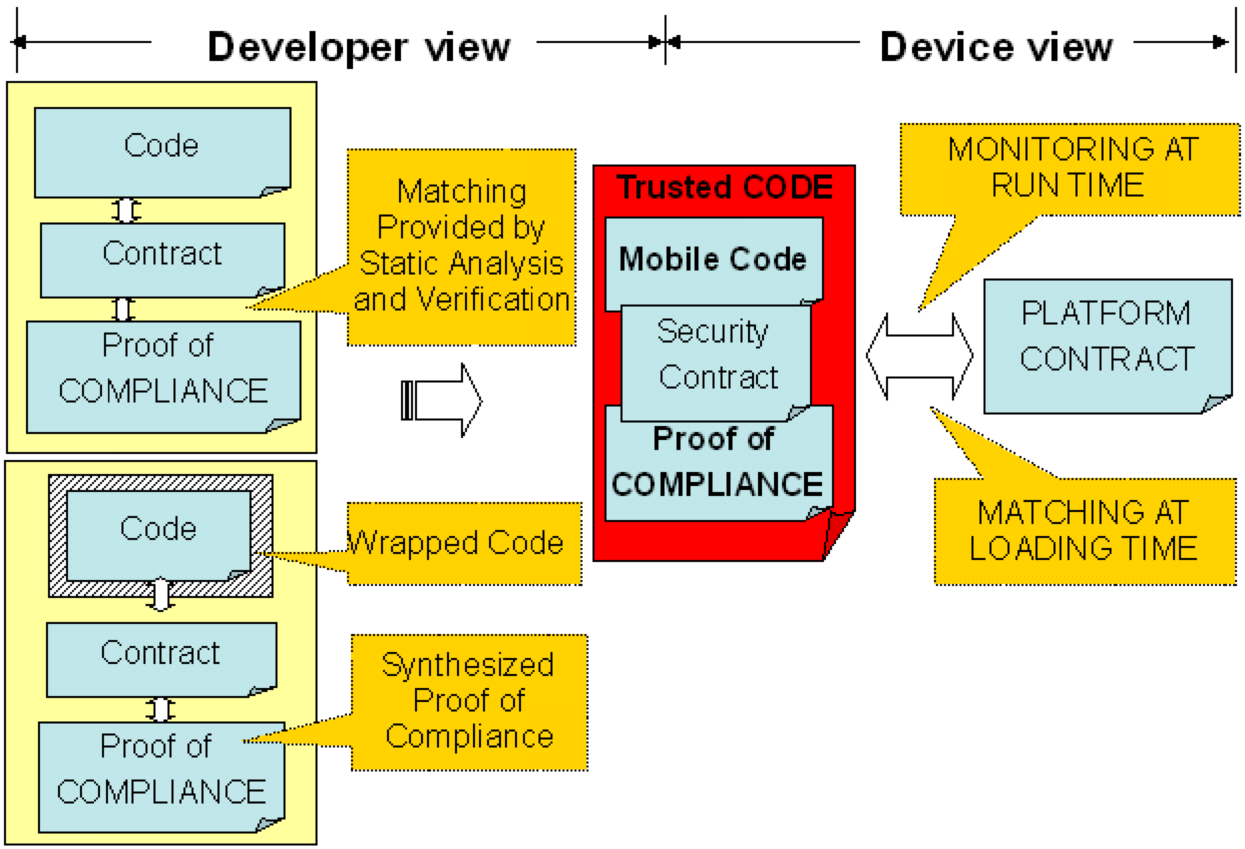
Outline

- Motivation
- Security x Contract
  - Concepts
  - Workflow
- Automata Modulo Theory (AMT)
  - AMT Theory
  - Contract/Policy Matching
- Conclusions
  - Issues yet to be addressed

Motivations

- A validation infrastructure exists
  - – A signature is checked on the device
  - – No semantics is attached to it
- Some technologies exist
  - – Static analysis to prove program properties
  (Leroy et al., Morriset et al., Fournet et al.)
  - – Monitor generation for complex properties
  (Havelund & Rosu, Erlingsson et al., Hamlen et al., Ligatti et al.)
- Security-by-Contract (SxC) puts them together
  - – Use contracts as semantics for the signatures
  - – Use static analysis and monitors as basis

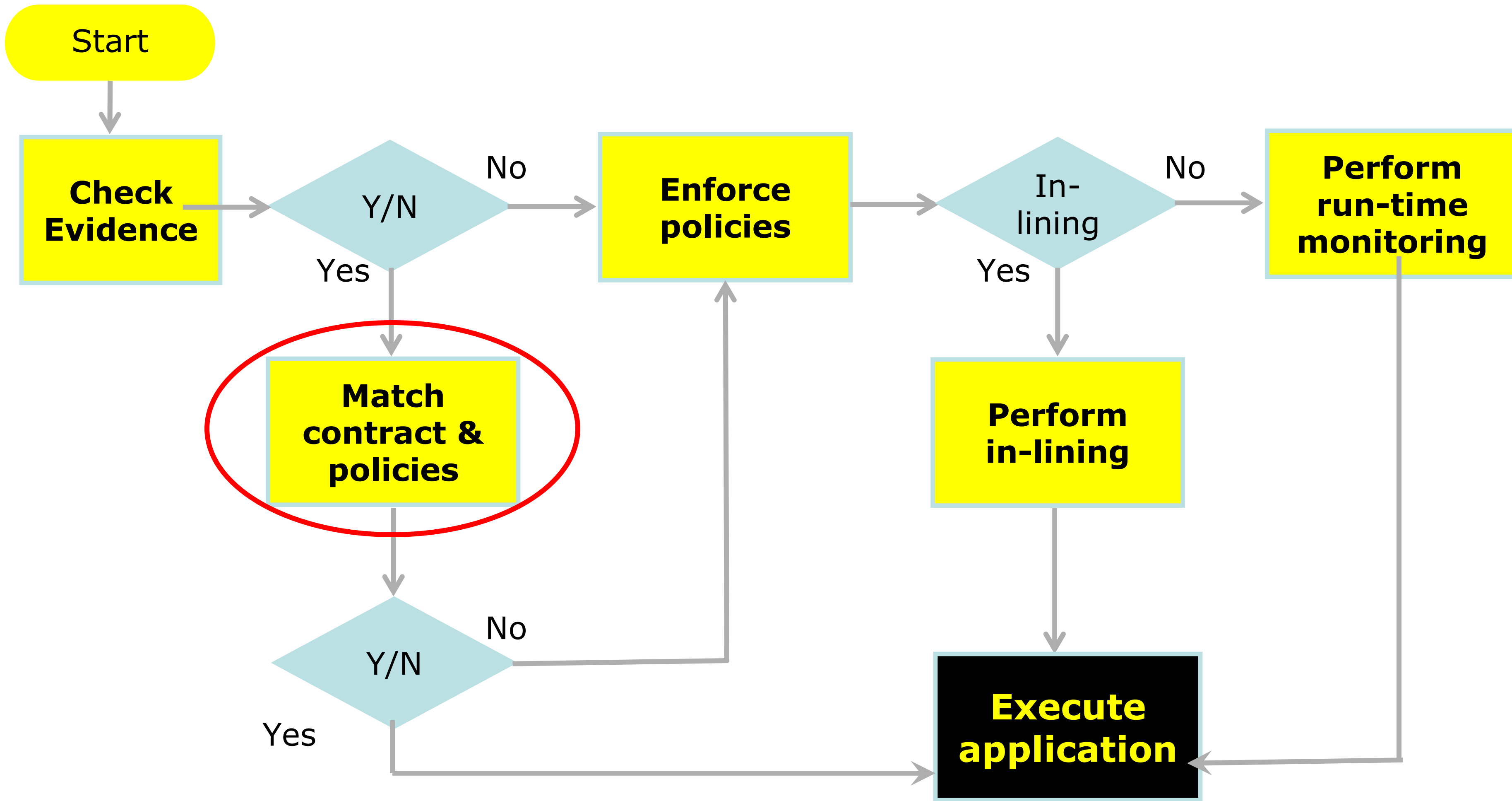
Mobile Code Components



Key Concepts

- Contract carried by application;
  - Claimed Security behavior of application
  - (Security) interactions with its host platform
  - Maybe with Proof that code satisfies contract
- Policy specified by a platform.
  - Desired Security behavior of application
  - Fine-grained resource control
- But I trust nobody, I just need policy monitor
  - Monitoring ONLY part of the story…

SxC Workflow – User's View



Contract vs Policy

Contract = What you Claim

Rules

- Used Methods
- Bounds on Methods Args
- Bounds on ret Values
- Allowed Sequences
- *Achievable Obligations*

Policy = What you should at most do

Rules

- Possible Methods
- Constraints on Methods Args
- Constraints on ret Values
- History-based access control
- *Desired Obligations*

Language Containment = Simulation of Finite Automata?

[NORDSEC'07] [PLAS'08]

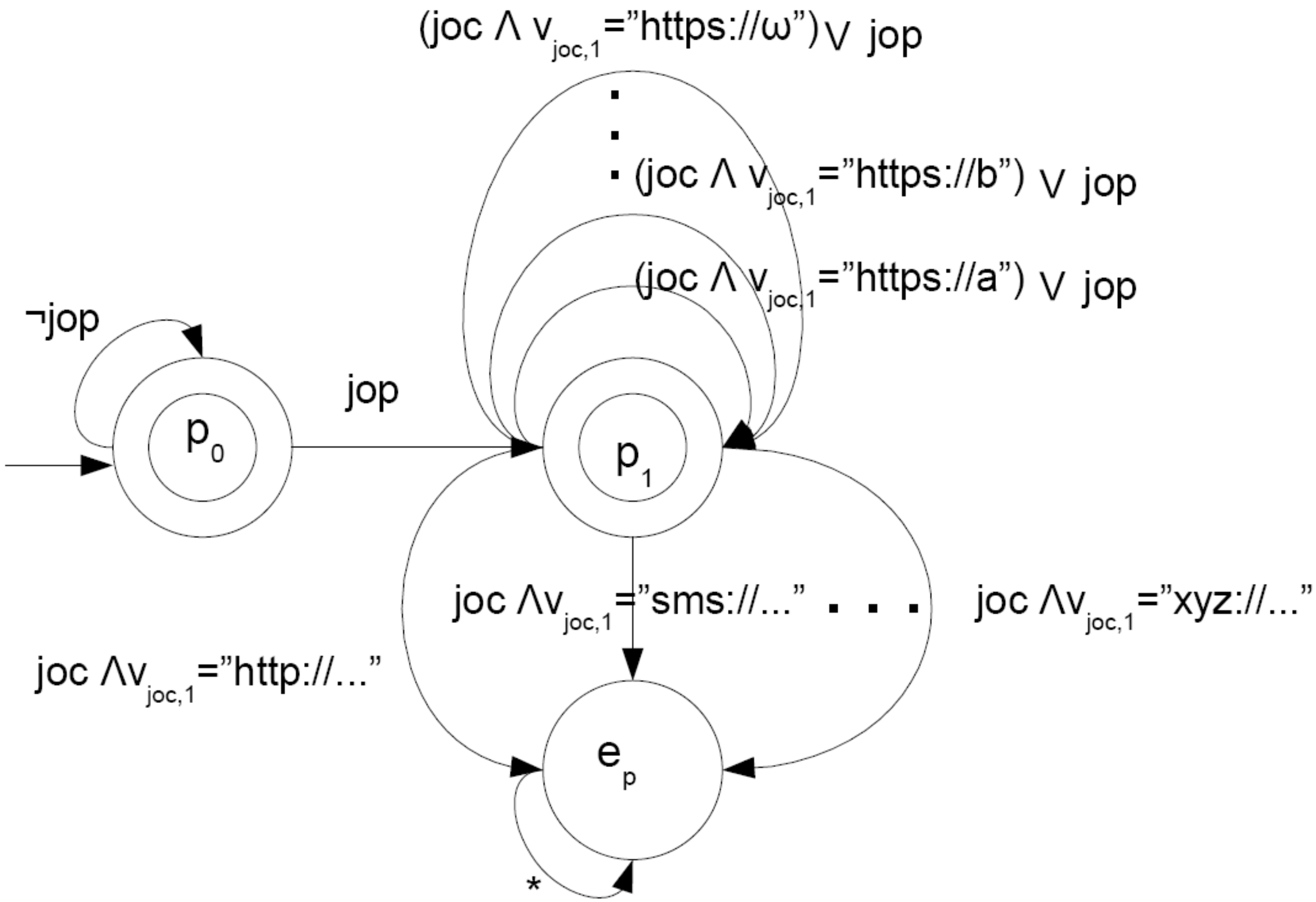
(Sekar et al. 2003)

Contract vs Policy: Practice

Contract = What you Claim	**Policy = What you should at most do**
Rules	**Rules**
• No connections After PIM was opened	• After PIM was accessed only secure connections can be opened
• Exec Connector.open() only if PIM.openPIMList() never called before.	• Exec Connector.open(url) only if url starts with “https://”

Language Simula Nondeterministic complementation????

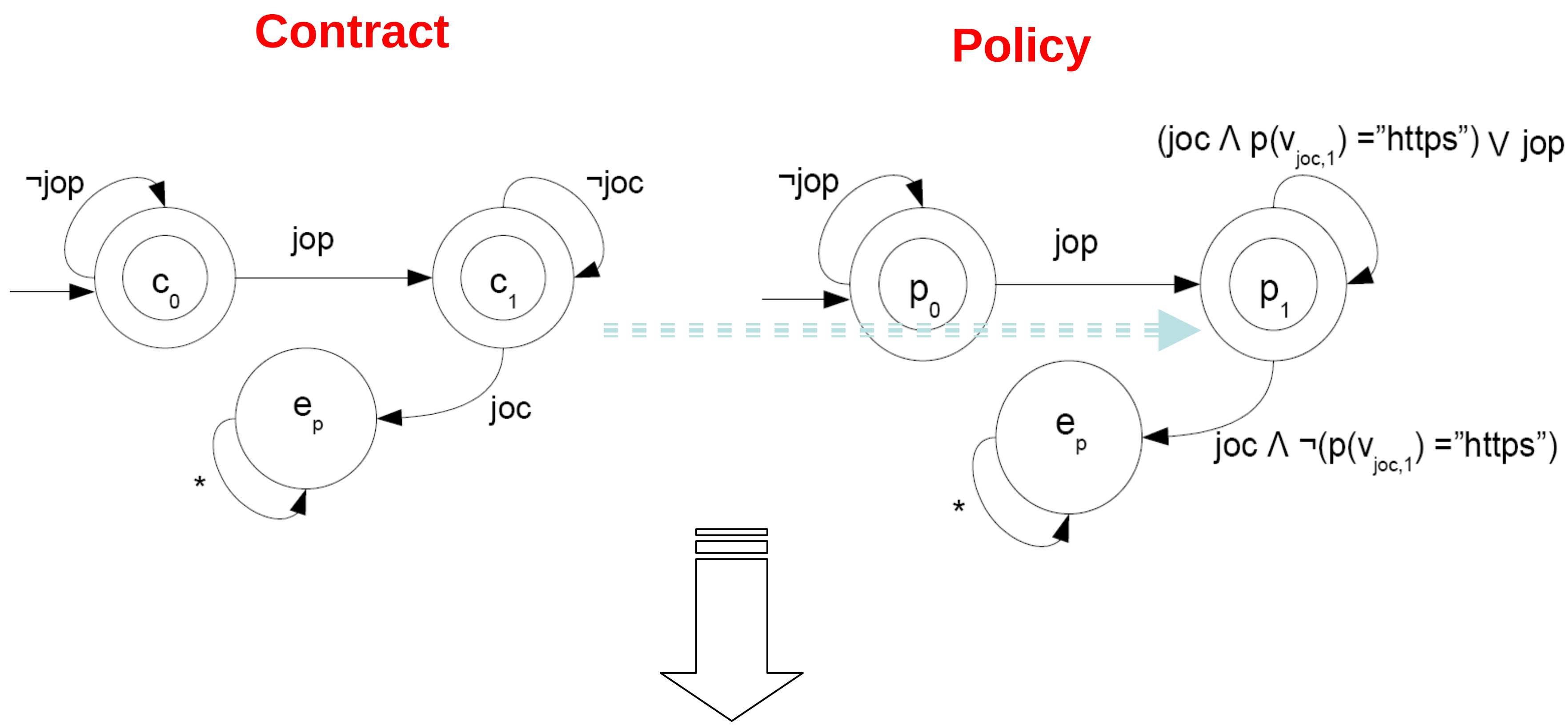
A Practical “Infinite” Policy



What's Automata Modulo Theory ($\mathcal{AMT}$)?

- *Finite state* Automata
  - They represent the security behavior (claimed or desired)
- But *Infinite edges*
  - Url starting with "https://" are not that few…
  - Battery Levels less than 30%
- Yet Finitely represented with Expressions
  - m=Java.IO.Connector &&
  - protocol(x)==https && protocol(x)==http
  - applicationType(x)!=jpg || appType(x)=appType(y)
- Needed Decidable theory for expressions

Contract vs Policy in $\mathcal{AMT}$



Simulation of Automata Modulo Theory

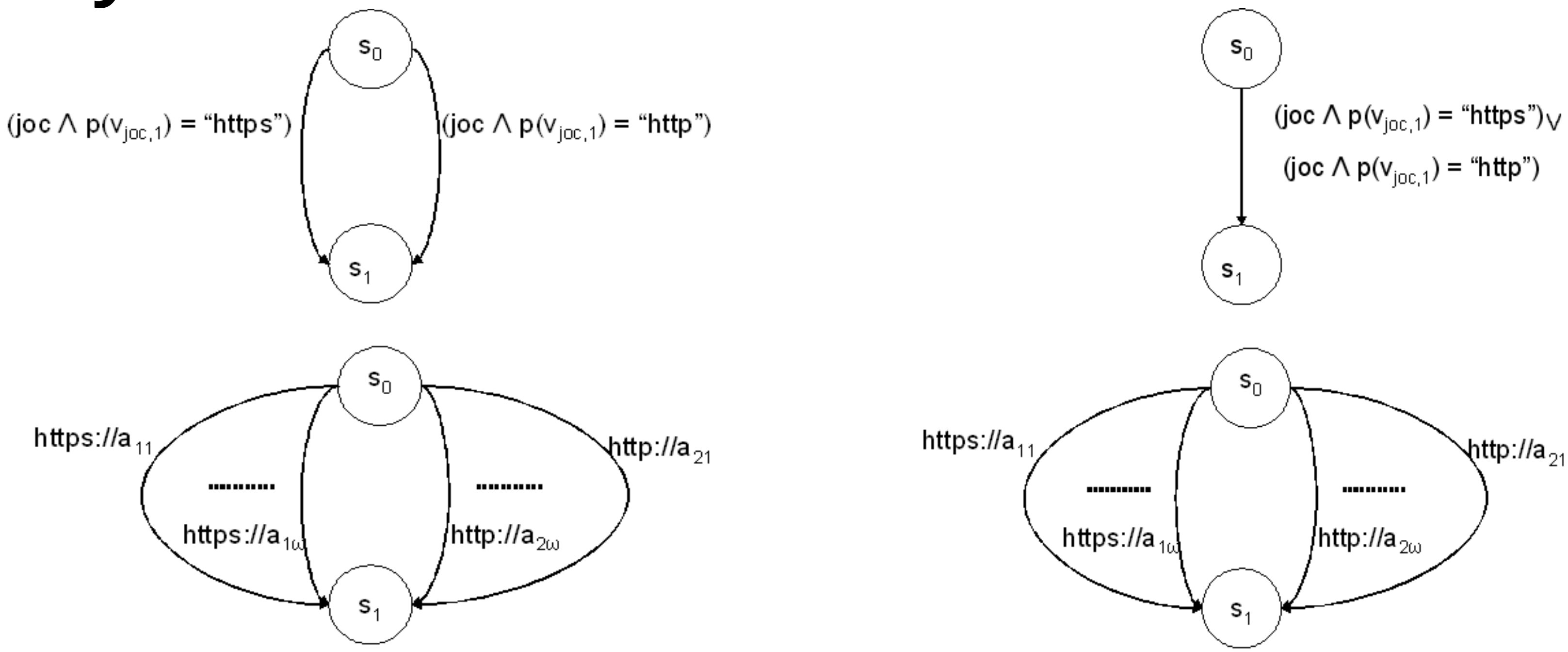
Matching as Simulation

- Matching = Simulation
  - Every APIs invoked by Contract can also be invoked by Policy.
    - Every behavior of Contract is also behavior of Policy.
  - Usually stronger than language inclusion
    - Policy allows midlet's Contract actions ”step-by-step”
- Compliance Game
  - Contract tries to make a concrete move and Policy follows accordingly to show that the Contract move is allowed.
  - IF expression of Contract implies expression of Policy is VALID (modulo theory)
  - THEN exists a move

Simulation as Game

- Winner of the game:
  - Contract cannot move: Policy wins.
  - Policy cannot move: Contract wins.
  - Otherwise, two infinite concrete runs *s* and *t* resp. of Contract and Policy:
    - *s* is an accepting concrete run and *t* is not an accepting concrete run: Contract wins.
    - Other cases: Policy wins.
- Failure of Matching
  - Policy cannot move => Contract is not compliant

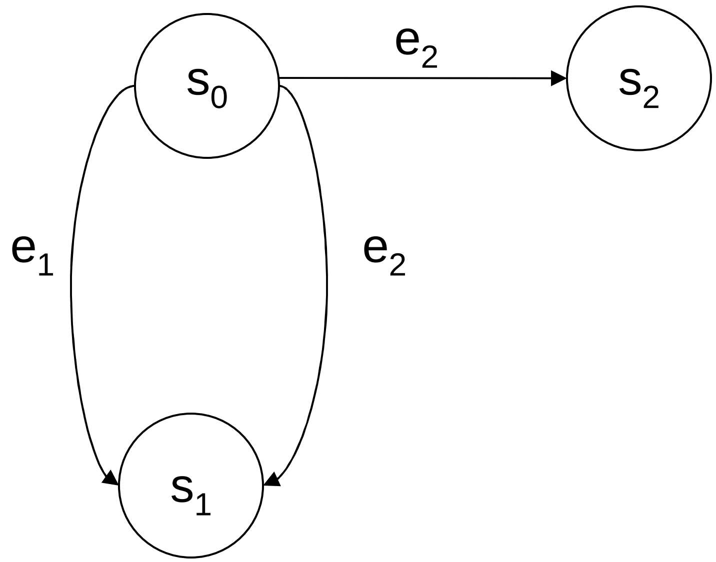
Symbolic vs Concrete Automaton

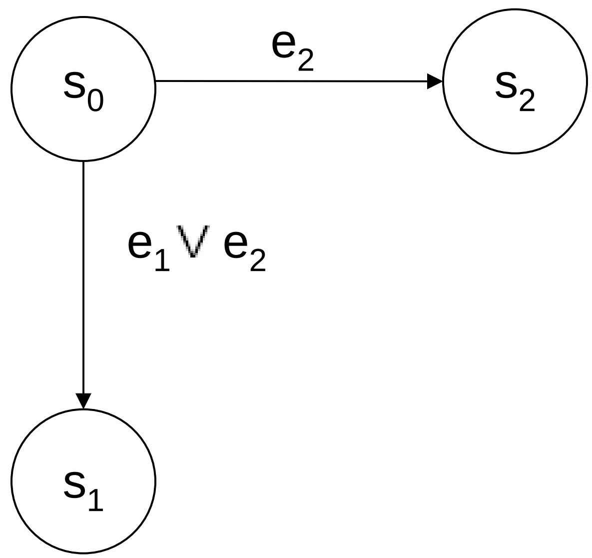


- If Ac ≤ Ap is an AMT fair simulation relation then Ac ⊑ Ap is a concrete fair simulation relation.
  - The converse does not hold in general.
  - Contrast to the simulation notions of (Hennessy and Lin 1995)
- AMT fair simulation is stronger than AMT language inclusion.

Normalized AMT

- For every q,q1 in set of states S there is at most one expression e1 in set of expressions E such that q1 in transition (q, e1).
  - Example from previous figure: Left is NOT normalized, Right is normalized
- Normalization is NOT always applicable:
  - disjunction of all expressions going to the same state as may change nondeterministic automata into deterministic automata (see figure below).
- IF automata are in normalized form
- THEN AMT fair simulation coincides with concrete fair simulation.





Simulation Policy-Contract Algorithm

- Finding counterexamples faster:
  - adapts Jurdzinski's algorithm on parity games (Jurdzinski 2000)
  - combine decision procedure for SMT (Cimatti et al.)
- Algorithm
  - Create compliance game graph G
  - $\mu(v) := 0$ for all v □ V
  - while $\mu(v) \neq \mu_{new}(\mu, v)$ for some v □ V do
    - $\mu := \mu new(\mu, v)$
  - IF $\mu < \infty$ THEN
    - Simulation exists

A bit Complexity for $\mathcal{AMT}$

- $\mathcal{AMT}$ runs
  - Concrete:
    - a sequence of states alternating with assignments (instantiation method args)
  - Symbolic:
    - a sequence of states alternating with expressions
- Criticality of fair simulation for matching
  - Jurdzinski's algorithm on parity games (Jurdzinski 2000)
- IF theory $\mathcal{T}$ decidable with oracle for SMT problem in complexity class $\mathcal{C}$ then:
  - Fair simulation of $\mathcal{AMT}_\mathcal{T}$ in *POL-TIME*$^{\mathcal{C}}$
  - Fair simulation of $\mathcal{AMT}_\mathcal{T}$ is $O(|S^c|.|S^p|.|\Delta^c+\Delta^p|)^{\mathcal{C}}$

Issues yet to be addressed

- Encoding of history dependent policies
  - allow certain strings we saw in the past
    - Eg connect only to url in the JAR manifest
    - Combine AMT with History Dependent Automata (Montanari & Pistore 1998)
    - Combine AMT with Extended Finite State Automata (Sekar et al. 2002)
- Infinite expressions
  - allow concrete run of infinite domains
    - Eg natural number not limited to some maximum length
    - Combine AMT with Finite Memory Automata (Kaminski & Francez 1994)

Conclusions

- Idea of Security-by-Contract
  - Always consider complete lifecycle monitoring is the end
  - Matching: be able to check that claimed security behavior of what you want to run is good for your security policy
- Concept of Automata Modulo Theory
  - we invented it for security policies of mobile code but…
  - usable for any security policy with a finite control structure but potentially infinite data
    - (secure workflows, protocol analysis, control-flow analysis etc.)
  - IF polynomial theory for deciding edges THEN Practical
  - Language Containment ≠ Simulation for AMT