A Spatio-Temporal Logic for the Specification and Refinement of Mobile Systems

Martin Wirsing
LMU Munich

(with Stephan Merz, INRIA Lorraine and Júlia Zappe, LMU Munich)

Motivation

Formal description of systems with mobile code

- WAN computing, agent-based systems
- correctness non-obvious, including security issues

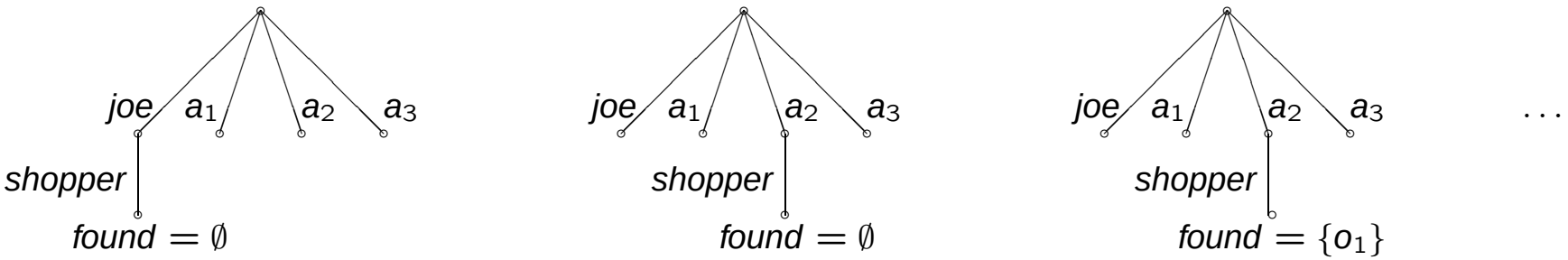
Existing formalisms for mobile systems

- mostly based on “operational” calculi
- some have associated logics: Ambient logic, μ-calculus for Klaim
- “intensional” semantics, reflecting structural equivalence
- no good notions of refinement

Reactive systems

- transition system semantics (next-state relation + fairness)
- well-established refinement notions
- stuttering equivalence: TLA

Basic Idea



Configurations (t, λ)

- t finite tree, edges labelled by unique names
- λ assigns local states to nodes

Computations $\sigma = (t_0, \lambda_0), (t_1, \lambda_1), \ldots$

Formulas

$shopper\langle found = \emptyset\rangle$	location *shopper* exists without found goods
$joe.shopper \gg a_2.shopper$	*shopper* moves from location *joe* to location a_2

MTLA (Mobile Temporal Logic of Actions)

TLA

- Linear Temporal Logic with formulas $\Box[\mathcal{A}]_v$
- Important feature: invariance under finite stuttering

+ Spatial Formulas

- Explicit name references $n[F]$
  - F holds at n ... provided n exists
  - NB: n may be arbitrarily far down the tree
- Structural modification of trees $\alpha.n \gg \beta.n$
  - subtree at αn before transition equals subtree at βn after transition
  - local state at moving subtree preserved

Refinement of mobile systems

Operation refinement

- decompose high-level operations
- represented in TLA by implication, thanks to stuttering invariance

Spatial decomposition

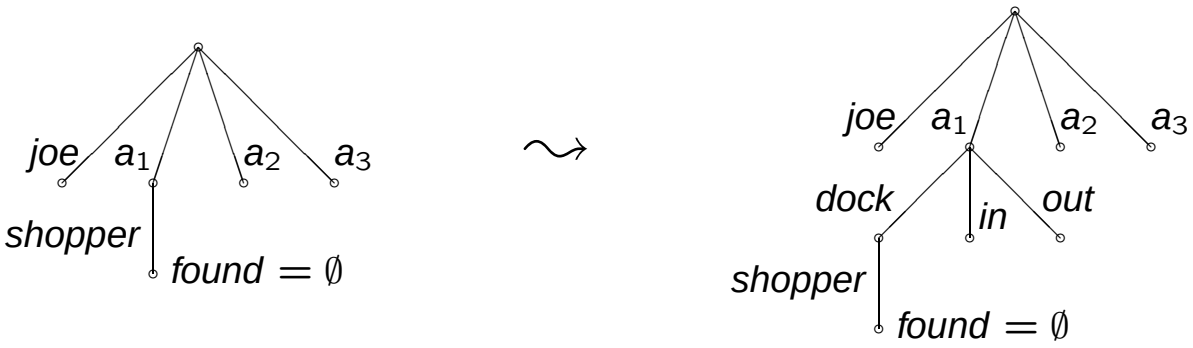
- refine high-level location *n* into a tree (with root named *n*)
- in general also distribute local state of *n*

Virtualisation of locations

- implement high-level location *n* by structurally different hierarchy
- preserve external behavior : *n* hidden from high-level interface

Spatial decomposition

Suppose visiting agents are kept in a “dock” location



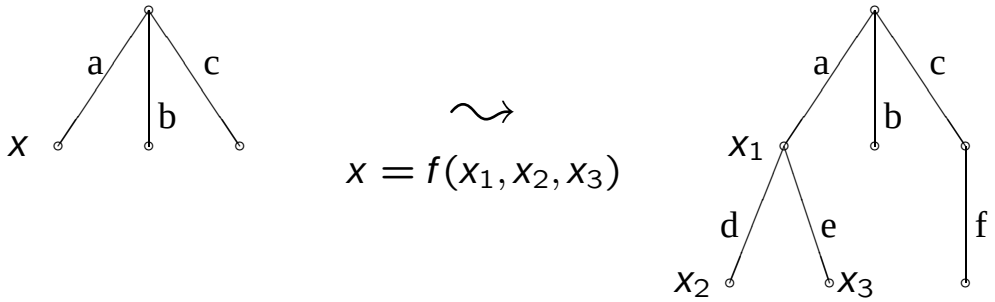
Still conforms to the original specification

- formula *Shopper* doesn’t mention locations *dock*, *in*, *out*
- location *shopper* is still below location a_1

Refinement is expressed as $\quad Impl \;\Rightarrow\; Spec$

Spatial decomposition: general case

Usually, decomposition requires distribution of state



Refinement is then expressed as $\quad Impl \;\Rightarrow\; \exists\, a.x : Spec$

local state variable x hidden from high-level interface

Virtualisation of locations

Hide entire locations, not just local state

External behavior preserved except for location *n*

- formally expressed by quantification over locations
- spatial refinement mappings

Refinement is expressed as $\quad Impl \;\Rightarrow\; \exists\!\!\!\!\!\!\;\; n : Spec$

Summing up

TLA'ish logic for specification of mobile systems

- add (few) spatial operators to describe topology
- concise description of system structure and its evolution

Refinement concepts represented as implication

- stuttering invariance supports operation refinement, as in TLA
- "deep" spatial operators support spatial decomposition

Future work: axiomatization, decidability, model checking