Online Monitoring of Distributed Systems with a Five-Valued Linear Temporal Logic

Prof. Dr. Holger Schlingloff
Fraunhofer FOKUS & Humboldt Universität

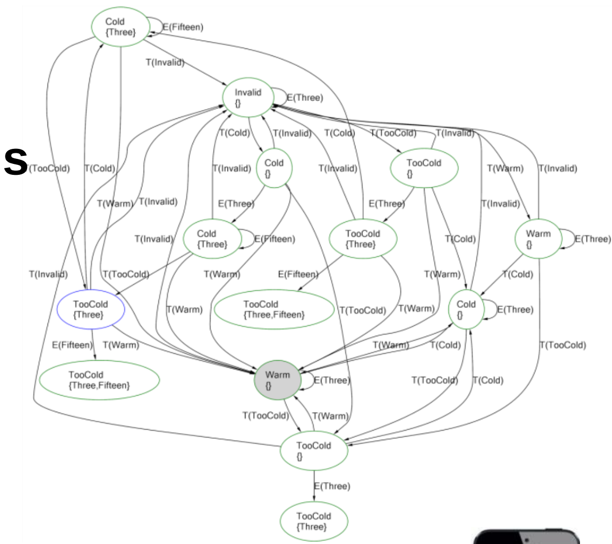
holger.schlingloff@fokus.fraunhofer.de

Joint work with Ming Chai, HU Berlin

IFIP WG1.3 Foundations of System Specification

Previous Talks

- **Specification and modelling of embedded systems**
  - formalization of natural-language specifications
  - “revision” operation for formulas and models



- **Model-based testing of software product lines**
  - feature modelling, domain engineering
  - enhancement of models, reuse of test-cases
  - three-valued test assignment

Today

- **Monitoring (aka “passive testing”)**
  - observing instead of influencing

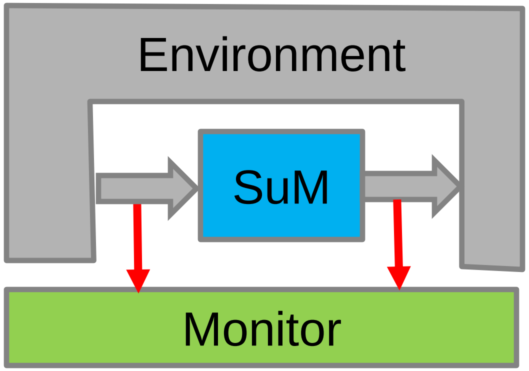
Structure of this Talk

1. Monitoring
2. Dimensions of uncertainty
3. Multi-valued logic
4. Monitoring algorithm
5. Example: RBC/RBC Handover

1. Runtime monitoring

- Observe rather than influence the behaviour of a system
  - useful for watchdog mechanisms, supervision, portmortem-diagnosis, ...
  - in particular interesting for multi-core technology
    (one core is working, the other one is watching)

- **Difference to verification:** no model of the system required
  (observing the actual system)

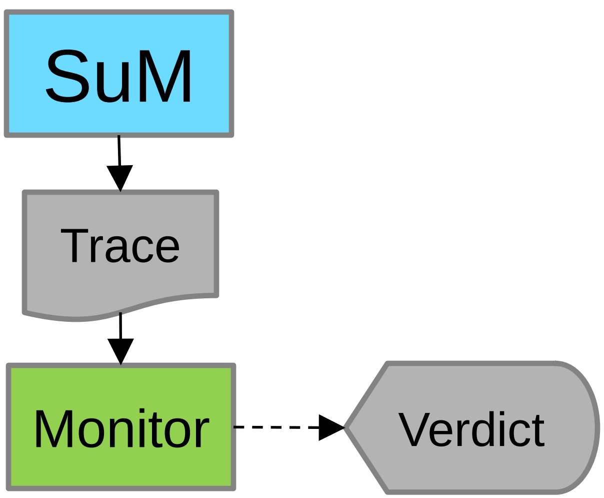


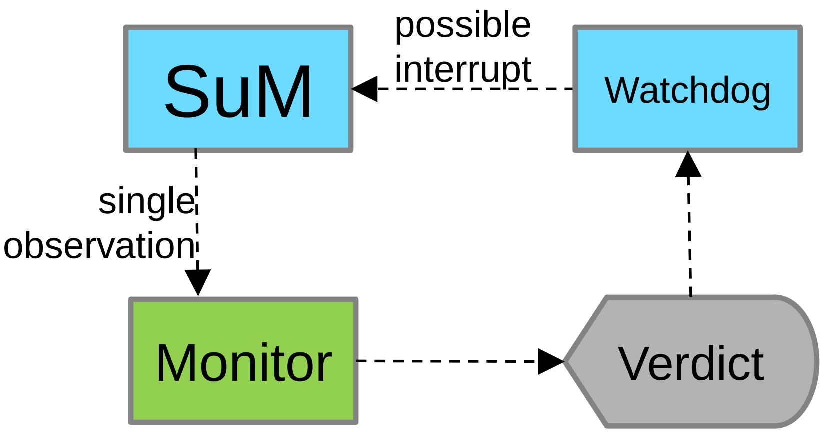
- **Difference to testing:** no artificial stimulation
  (observing the system in its actual production environment)

- Disregard Heisenberg's uncertainty principle
  (observation does *not* change the system's behaviour)
  exception: interrupt / terminate the system

Offline and Online Monitoring

- *Offline version:* given a trace (e.g., a sequence of events) and a spec (e.g., a finite automaton): Does the trace conform to the specification?
    - well-known word problem of finite automata
    - infinite executions?
- *Online version:* given a system producing the trace, solve the same problem
    - “online algorithms” for predictable worst-case deviation from optimum
    - here: no “optimum”, but statement about conformance





Fraunhofer FOKUS

Conformance

How to specify properties to be observed?

- temporal logic
- process algebra
- automata & transition systems
- UML models, ...

➔ Here: classical LTL, (metric LTL for real time constraints)

When does the behaviour of a system conform to its LTL specification?

- Safety (“nothing bad ever happens”)
  - as soon as it’s violated, the answer is “no”
  - up to then the answer is “don’t know”
- Liveness (“eventually something good will happen”)
  - if all obligations are satisfied, the answer is “yes”
  - up to then, the answer is “don’t know”

2. Dimensions of Uncertainty

- Uncertain future
  - if not the whole trace is available (online)
- Uncertain timing
  - if the parallel interleaving cannot be observed exactly
- Uncertain state
  - if the internal state of the system is unknown, i.e., system has observationally equivalent states; can lead to mode confusion
- Uncertain history
  - if monitoring a system which is already running, i.e., not from the start
- Other uncertainties

Subsequently, we deal with the first two of these dimensions

Uncertain Future

- LTL models are infinite (or finite/infinite) sequences $\tau = (\tau_o\ \tau_1\ \tau_2\ ...)$
- Truth value at point τ_i depends on some points τ_j with $j \geq i$
- Want to issue an "intermediate" verdict at point τ_i
- Bauer/Leucker/Schallhart (2011): three-valued LTL
  - "?" denotes "unknown"
  - Kleene's three-valued truth tables

$\vee$	Y	?	N
Y	Y	Y	Y
?	Y	?	?
N	Y	?	N

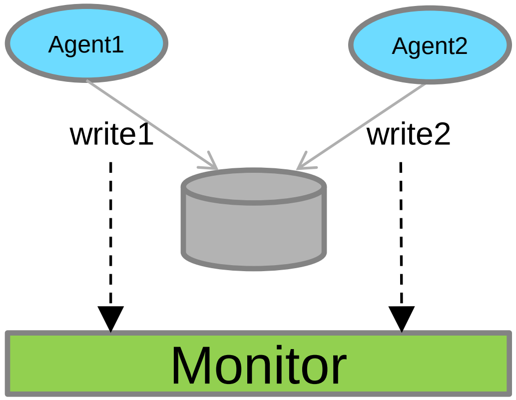
$\neg$	
Y	N
?	?
N	Y

- Example
  - Observation = (open, read, write, write, close)
  - Property = (open → **F** close)
  - Verdict = (?, ?, ?, ?, T)
- Extension: four-valued logic (tt, ff, pt, pf)
  - Verdict = (pf, pf, pf, pf, tt)

Uncertain Timing

If the system under monitoring is distributed and consists of several communicating subsystems, timing may not be accurately observable

Example:



What is the truth value of „write1 is executed before write2“?

(is this the same „unknown“ as before?)

Timestamping will only help if a global clock is available

3. Multiple Truth Values

In monitoring, a truth value can be regarded as an answer to a question
(does the system satisfy the property? => Y, N)

Several answers possible ➔ powerset of truth values
i.e., ? = {Y, N}
Empty set disallowed: No answer is not an answer
yields three truth values (Y,N,?)

Evolution of knowledge ➔ pairing of truth values
e.g., pt = (Y ~> ?)
Monotonicity assumption: Increased knowledge leads to more choices
yields five truth values

More Formally

Assume standard LTL with operators $\bot, \rightarrow, \ldots, \mathbf{X}, \mathbf{U}$.
The validation function assigns to each ω-sequence τ and formula φ a unique truth value $[[\tau \models \varphi]] \in \{Y, N\} = \mathbf{B}_2$

Let $\mathcal{T}$ be a set of ω-sequences.
Define $[[\mathcal{T} \models \varphi]] = \bigcup_{\tau \in \mathcal{T}} [[\tau \models \varphi]]$

clearly, $[[\mathcal{T} \models \varphi]] \in \{\{Y\}, \{N\}, \{Y, N\}\} = \mathbf{B}_3$

Let $\mathcal{T}$ be a set of *finite* sequences.
Define $[[\mathcal{T} \models \varphi]] \subseteq (\mathbf{B}_3 \times \mathbf{B}_3)$ by

$$[[\mathcal{T} \models \varphi]] = (A \rightsquigarrow B) \quad \text{iff}$$

$$[[\mathcal{T} \circ \{\varepsilon^\omega\} \models \varphi]] = A \text{ and}$$

$$[[\mathcal{T} \circ \Sigma^\omega \models \varphi]] = B$$

Fraunhofer FOKUS

Five Truth Values

From this, $[[\mathcal{T} \models \varphi]] = (A \leadsto B)$ implies $A \subseteq B$

Therefore, we have 5 truth values:

true: $(\{Y\} \leadsto \{Y\})$

false: $(\{N\} \leadsto \{N\})$

possibly true: $(\{Y\} \leadsto \{Y, N\})$

possibly false: $(\{Y\} \leadsto \{Y, N\})$

unknown: $(\{Y, N\} \leadsto \{Y, N\})$

For example, $[[\tau \models \varphi]] = (\{Y\} \leadsto \{Y, N\})$ iff

- $\tau \circ \varepsilon^{\omega} \models \varphi$ and
- there exists some $\tau' \in \Sigma^{\omega}$ such that $\tau \circ \tau' \not\models \varphi$

Uncertain Time Events

Events with an approximative time stamp: $ue = (e, t, \Delta t)$

intuition: event e has occured at time $t \pm \Delta t$

Each set of uncertain time events $\mathcal{B} = \{ue_1, ..., ue_n\}$ gives rise to a set of traces: Trace $\tau = (\tau_1 \tau_2 ... \tau_n)$ is consistent with $\mathcal{B}$ iff there exists a permutation ρ such that

- τ consists of the events $\mathcal{B}$ ordered according to ρ
- if $\rho_i < \rho_j$ then $t_i < t_j$ or the intervals ue_i, ue_j overlap

E.g. if $\mathcal{B} = \{(a, 0, 3), (b, 2, 3), (c, 4, 3)\}$

then $\mathcal{T} = \{abc, bac, acb\}$

Fraunhofer FOKUS

E.g. if $\mathcal{B} = \{(a,0,3),(b,2,3),(c,4,3)\}$

then $\mathcal{T} = \{abc, bac, acb\}$

Examples:

$$[[\mathcal{T} \models (a \rightarrow \mathbf{F}\ b)]] = \mathit{true}$$

$$[[\mathcal{T} \models (\mathbf{G}\ (a \vee b)]] = \mathit{false}$$

$$[[\mathcal{T} \models \mathbf{G}\ \mathbf{F}(b \vee c)]] = \mathit{possibly\ true}$$

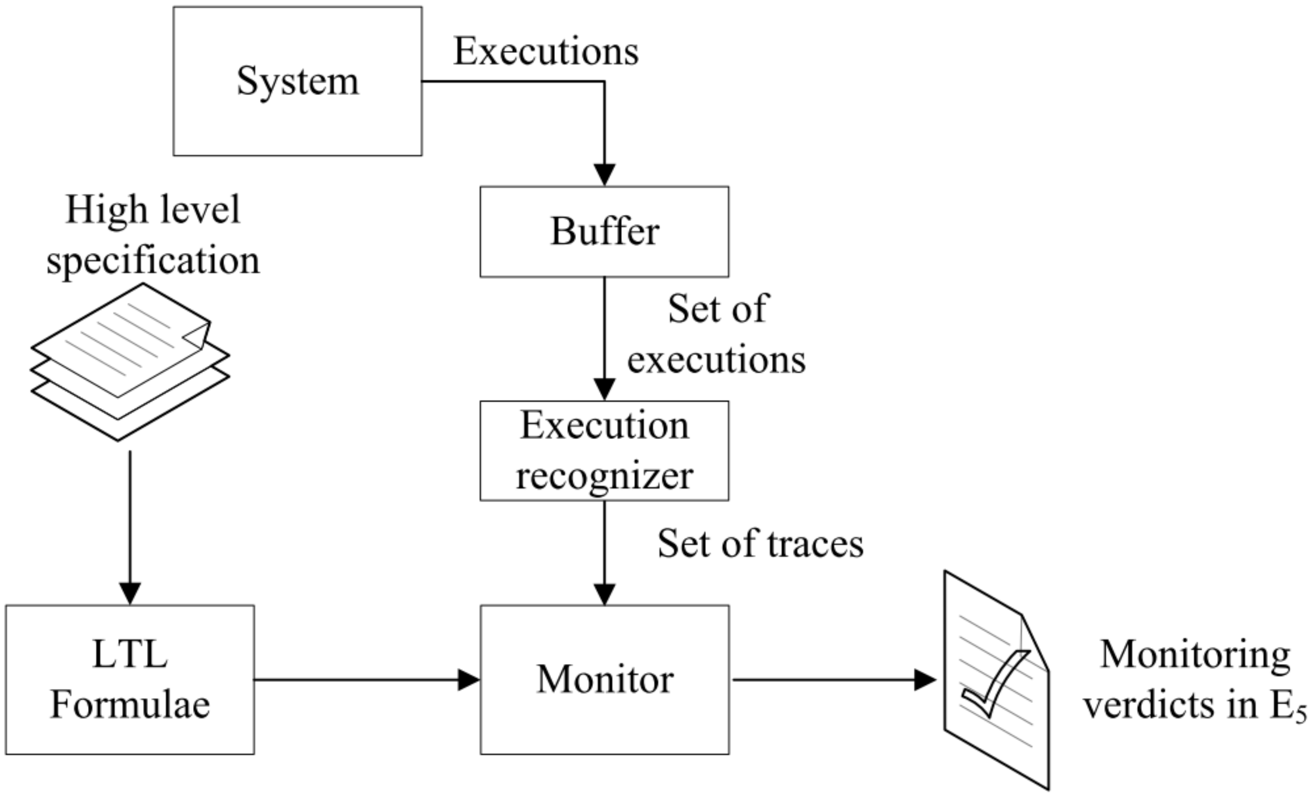
$$[[\mathcal{T} \models \mathbf{G}(c \rightarrow \mathbf{F}\ (a))]] = \mathit{possibly\ false}$$

$$[[\mathcal{T} \models ((a\ \mathbf{U}\ b)]] = \mathit{unknown}$$

Safety properties are *false*, *unknown* or *possibly true*

Liveness properties are *true*, *unknown* or *possibly false*

4. Monitoring Algorithm



Naive algorithm takes $2^{|\mathcal{B}|} \cdot 2^{|\varphi|}$ time

Naive algorithm takes $2^{|\mathcal{B}|} \cdot 2^{|\varphi|}$ time

Fortunately, executions usually consist of "a string of diamonds"

Forward rewriting algorithm:

Decompose formula into safety- and liveness-part

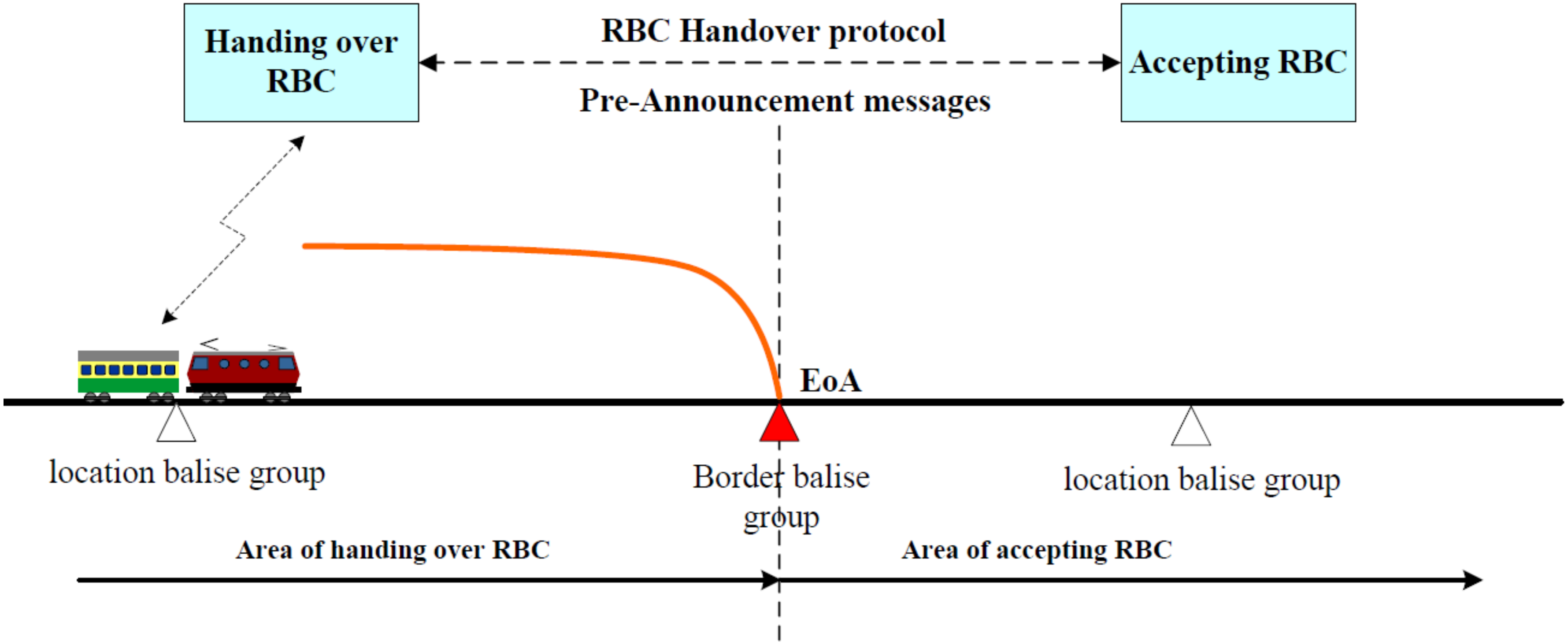
Complexity $(2^{|\mathcal{B}_1|} + \ldots + 2^{|\mathcal{B}_n|}) \cdot 2^{|\varphi|}$

Further improvements may be possible

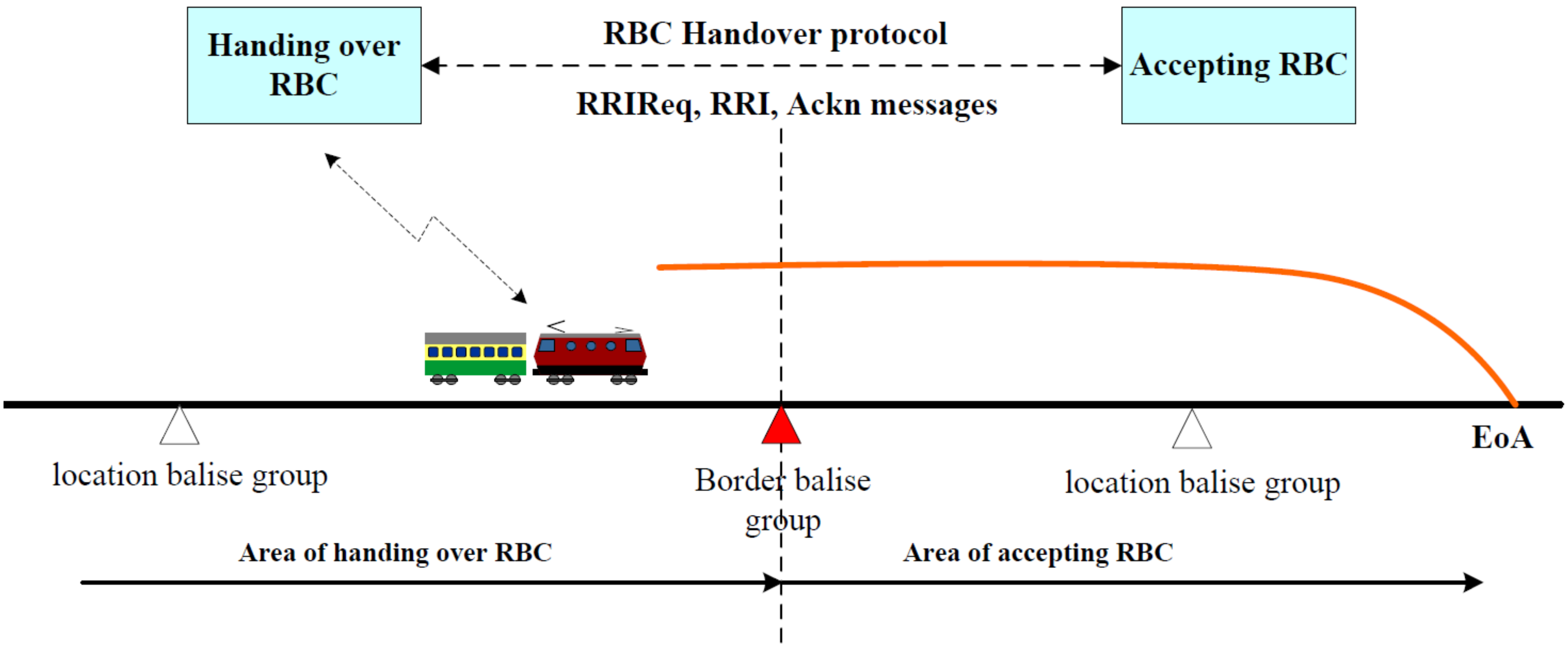
Experimental implementation in Maude

Function Five-valued LTL checking ($\mathcal{T}$, φ)
/* initialization of the checking process */
for j = 1 **to** $|FList\ (\varphi)|$ **do** {
$\psi \leftarrow$ FList[j];
RewF $[j] \leftarrow$ Rewrite ($\mathcal{T}_1$, ψ); }
for $i = 2$ **to** $|\mathcal{T}|$ **do** {
If φ is a temporal operation free formula
then print "$[\mathcal{T}_i \models \varphi] =$" $[\mathcal{T}_1 \models \varphi]$;
else print "$[\mathcal{T}_i \models \varphi] =$" SubFC ($\varphi$, $\mathcal{T}_i$, RewF[1]);}
Function SubFC (φ, $\mathcal{T}_i$, RewF $[j]$)
/* rewriting algorithm for subformulae */
for j = 1 **to** $|FList\ (\varphi)|$ **do** {
$\psi \leftarrow$ FList [j];
case ψ is a propositional logic formula
RewF $[j] \leftarrow (\mathcal{T}_i \models \psi)$;
Eva $[j] \leftarrow [\mathcal{T}_i \models \psi]$;
case $\psi = \neg\psi_1$
RewF $[j] \leftarrow$ not $(\mathcal{T}_i \models \psi)$;
Eva $[j] \leftarrow \neg\ [\mathcal{T}_i \models \psi]$;
case $\psi = \psi_1\ \mathcal{U}\ \psi_2$
RewF $[j] \leftarrow$ RewF $[j]$ or
(Rewrite ($\mathcal{T}_i$, $\mathbf{G}\psi_1$);
Eva $[j] \leftarrow$ [RewF $[j]$] $\vee$
([Rewrite ($\mathcal{T}_i$, $\mathbf{G}\ \psi_1$)] $\wedge$ *pf*);
case $\psi = \mathbf{X}\ \psi_1$
if $|\mathcal{T}_i| > 1$ then RewF $[j] \leftarrow (\mathcal{T}_i \models \mathbf{X}\ \psi_1)$;
else Eva $[j] \leftarrow$ ***pf***;
RewF $[j] \leftarrow \psi_1$;}
return Eva $[|FList\ (\varphi)|]$;

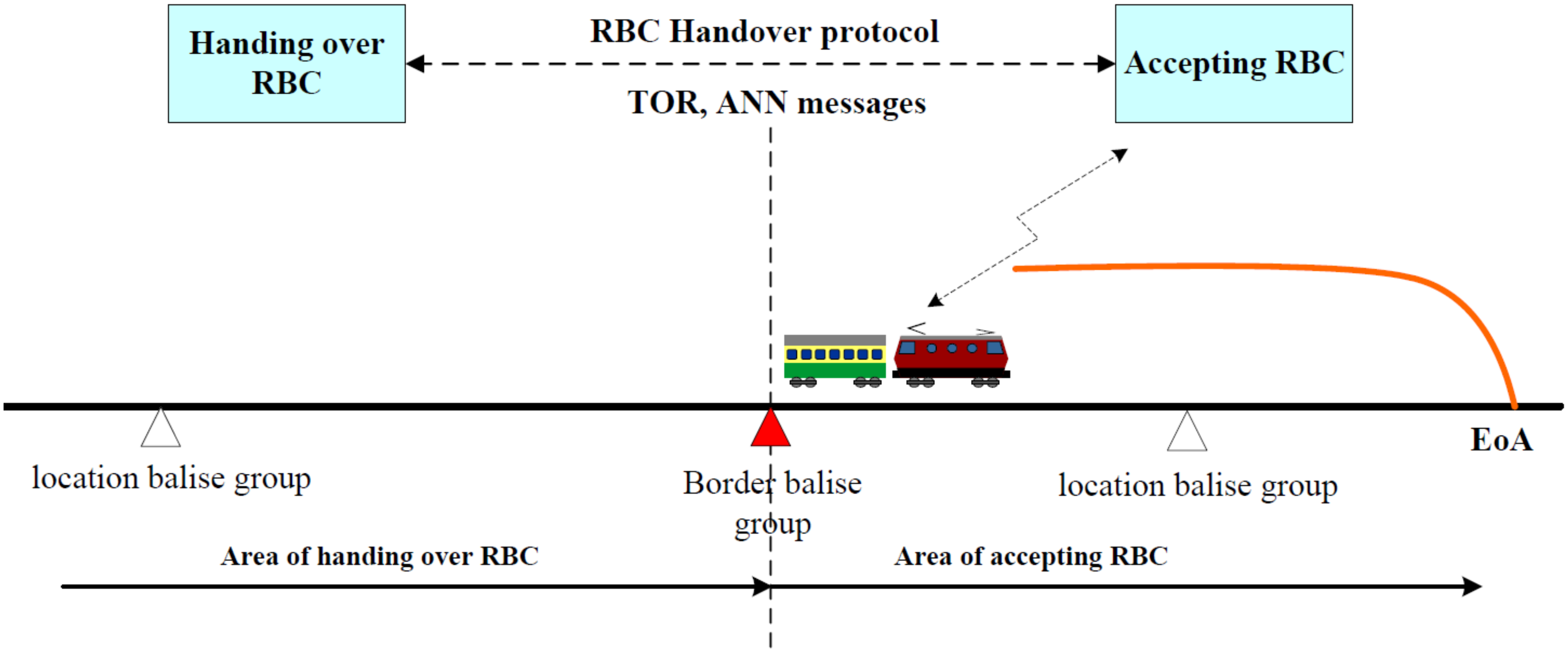
5. An Example: The RBC/RBC Handover Process



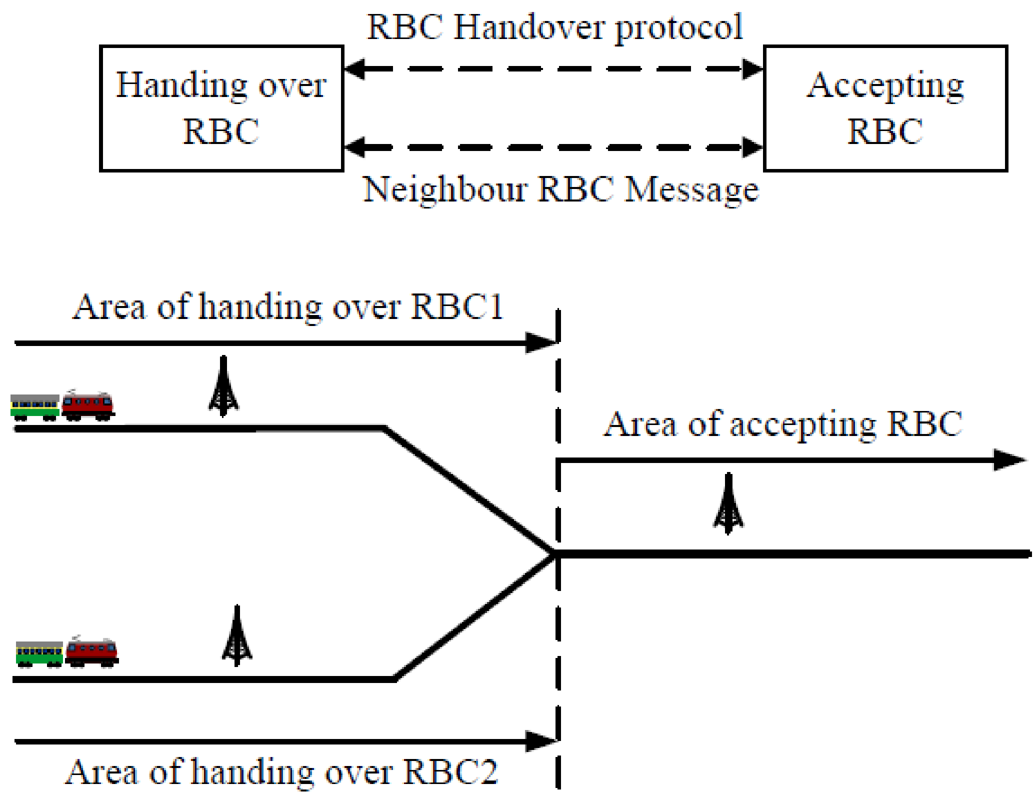
The RBC/RBC Handover Process



The RBC/RBC Handover Process



Monitoring Case Study: RBC/RBC Handover



Properties to be monitored

Consider the following properties:

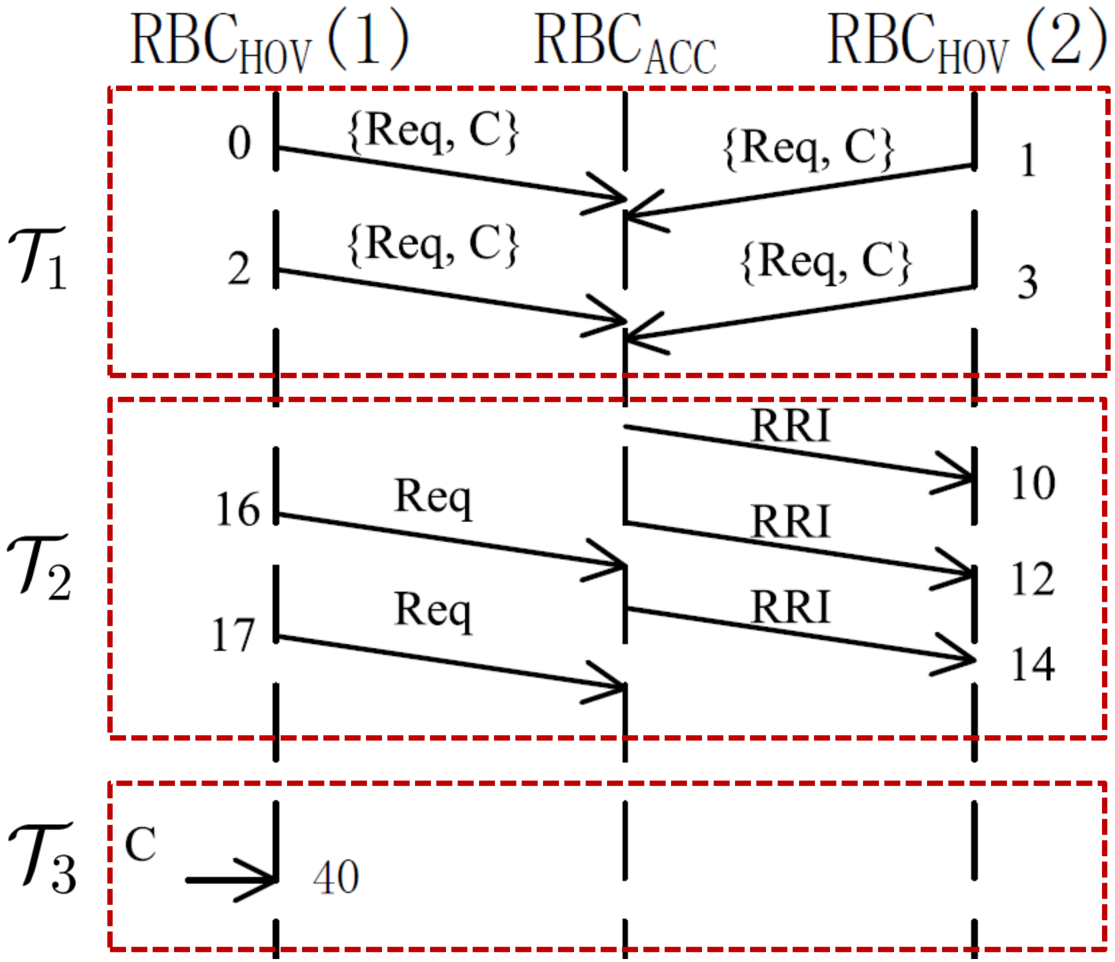
- An RBC_{HOV} sends a request to the RBC_{ACC}, then the RBC_{ACC} sends RRI to the RBC_{HOV}, and sets the route occupied.

$$\varphi_1 = (Req(i) \land C) \land \mathbf{F}\,(RRI(i) \land \neg C)$$

- If RBC_{ACC} sends an RRI to an RBC_{HOV}, it can not send it to another RBC_{HOV} until the route is clear

$$\varphi_2 = \mathbf{G}\,(RRI(i) \rightarrow (\neg RRI(i')\;\mathcal{U}\;C)), \text{ with } i \neq i'$$

Monitoring Example



Assume that maximal time delay of an event is 5 time units (i.e., Δt = 5).

$$\varphi_1 : (Req(i) \wedge C) \wedge \mathbf{F}\ (RRI(i) \wedge \neg C)$$

$$\varphi_2 : \mathbf{G}\ (RRI(i) \rightarrow (\neg RRI(i')\ \mathcal{U}\ C))$$

	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_3$
φ_1	*pf*	*uk*	*uk*
φ_2	*pt*	*pt*	*pt*

Summary

1. Monitoring

   online-monitoring as an interesting completion to verification and testing

2. Dimensions of uncertainty

   diffuse observations can / may / should give fuzzy results

3. Multi-valued logic

   union and product give five truth values

4. Monitoring algorithm

   complexity ”almost” linear in the number of observations

5. Example: RBC/RBC handover

   application is feasible, but more research is needed

THANK YOU FOR YOUR ATTENTION!