Spatio-temporal modelling and simulation with the FCPP Aggregate Programming framework

III - Logics

Volker Stolz,[†] Ferruccio Damiani,[*] Giorgio Audrito[*]

[*]University of Turin, Italy
[†]Western Norway University of Applied Sciences, Bergen

June 20, 2022

Outline

- Runtime Verification
  - What is it?
  - Runtime Verification for the IoT
- Temporal logic
- Spatial logic

What is it?

- properties of a computing system (safety/liveliness/correctness. . .)
- whenever proving them is too hard. . .
- you can monitor their failure instead!
- automatic synthesis of monitors from specifications

Runtime Verification for the IoT

Several requirements need to be met:

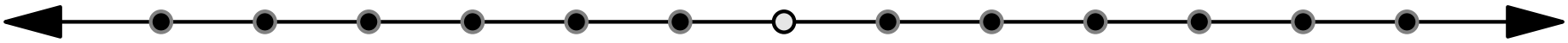
- fully distributed monitors *(multi-hop networks)*
- monitors integrated within the IoT system
- dynamic devices and monitors *(may fail, join, move)*
- low resource consumption *(limited device capabilities)*

Outline

- Runtime Verification
- Temporal logic
    - Linear Time Logics (LTL)
    - Branching Time Logics (CTL)
    - Past-CTL on Event Structures
    - Sample Applications
    - Monitoring Past-CTL in Field Calculus
- Spatial logic

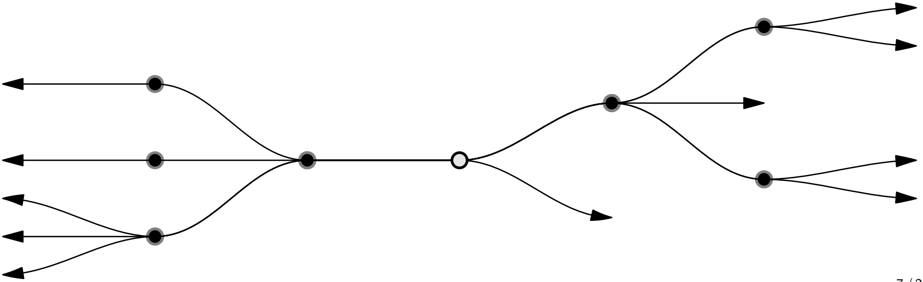
Linear Time Logics (LTL)

$$
\begin{array}{lll}
\phi ::= & \bot \mid \top \mid q \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \Rightarrow \phi) \mid (\phi \Leftrightarrow \phi) & \text{logical op.} \\
& \mid (\mathrm{X}\,\phi) \mid (\phi\,\mathrm{U}\,\phi) \mid (\mathrm{F}\,\phi) \mid (\mathrm{G}\,\phi) & \text{future op.} \\
& \mid (\mathrm{Y}\,\phi) \mid (\phi\,\mathrm{S}\,\phi) \mid (\mathrm{P}\,\phi) \mid (\mathrm{H}\,\phi) & \text{past op.}
\end{array}
$$



Branching Time Logics (CTL)

$$\begin{array}{rll} \phi ::= & \bot \mid \top \mid q \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \Rightarrow \phi) \mid (\phi \Leftrightarrow \phi) & \text{logical op.} \\ & \mid (\mathrm{X}\,\phi) \mid (\mathrm{AX}\,\phi) \mid (\mathrm{EX}\,\phi) \mid (\phi\,\mathrm{U}\,\phi) \mid (\phi\,\mathrm{AU}\,\phi) \mid (\phi\,\mathrm{EU}\,\phi) & \text{future op.} \\ & \mid (\mathrm{F}\,\phi) \mid (\mathrm{AF}\,\phi) \mid (\mathrm{EF}\,\phi) \mid (\mathrm{G}\,\phi) \mid (\mathrm{AG}\,\phi) \mid (\mathrm{EG}\,\phi) & \\ & \mid (\mathrm{Y}\,\phi) \mid (\mathrm{AY}\,\phi) \mid (\mathrm{EY}\,\phi) \mid (\phi\,\mathrm{S}\,\phi) \mid (\phi\,\mathrm{AS}\,\phi) \mid (\phi\,\mathrm{ES}\,\phi) & \text{past op.} \\ & \mid (\mathrm{P}\,\phi) \mid (\mathrm{AP}\,\phi) \mid (\mathrm{EP}\,\phi) \mid (\mathrm{H}\,\phi) \mid (\mathrm{AH}\,\phi) \mid (\mathrm{EH}\,\phi) & \end{array}$$



Past-CTL on Event Structures

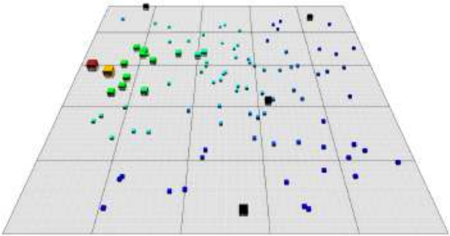
- $\mathbf{E}[\![\top]\!](\epsilon) = \top$, and $\mathbf{E}[\![q]\!](\epsilon)$ is $\Phi_q(\epsilon)$
- $\mathbf{E}[\![\neg\phi]\!](\epsilon) = \neg\mathbf{E}[\![\phi]\!](\epsilon)$ and $\mathbf{E}[\![\phi_1 \vee \phi_2]\!](\epsilon) = \mathbf{E}[\![\phi_1]\!](\epsilon) \vee \mathbf{E}[\![\phi_2]\!](\epsilon)$
- $\mathbf{E}[\![\mathrm{Y}\,\phi]\!](\epsilon) = \mathbf{E}[\![\phi]\!](\epsilon')$ where ϵ' is the event preceding ϵ on the same device (if it exists, $\mathbf{E}[\![\mathrm{Y}\,\phi]\!](\epsilon) = \bot$ otherwise)
- $\mathbf{E}[\![\mathrm{AY}\,\phi]\!](\epsilon) = \bigwedge_{\epsilon' \rightsquigarrow \epsilon} \mathbf{E}[\![\phi]\!](\epsilon')$ (ϕ is true in each preceding event ϵ')
- $\mathbf{E}[\![\phi_1\,\mathrm{AS}\,\phi_2]\!](\epsilon)$ holds iff for every path $\epsilon_1 \rightsquigarrow \ldots \rightsquigarrow \epsilon_n = \epsilon$ such that ϵ_1 has no neighbours, $\exists i.\ \mathbf{E}[\![\phi_2]\!](\epsilon_i)$ and $\forall j > i.\ \mathbf{E}[\![\phi_1]\!](\epsilon_j)$
- $\mathbf{E}[\![\phi_1\,\mathrm{ES}\,\phi_2]\!](\epsilon)$ holds iff it exists a path $\epsilon_1 \rightsquigarrow \ldots \rightsquigarrow \epsilon_n = \epsilon$ such that $\mathbf{E}[\![\phi_2]\!](\epsilon_1)$ holds and $\mathbf{E}[\![\phi_1]\!](\epsilon_i)$ holds for $i = 2 \ldots n$
- $\mathbf{E}[\![\phi_1\,\mathrm{S}\,\phi_2]\!](\epsilon)$ holds iff it exists a path $\epsilon_1 \rightsquigarrow \ldots \rightsquigarrow \epsilon_n = \epsilon$ of events all on $\delta = d(\epsilon)$, such that $\mathbf{E}[\![\phi_2]\!](\epsilon_1)$ holds and $\mathbf{E}[\![\phi_1]\!](\epsilon_i)$ holds for $i = 2 \ldots n$

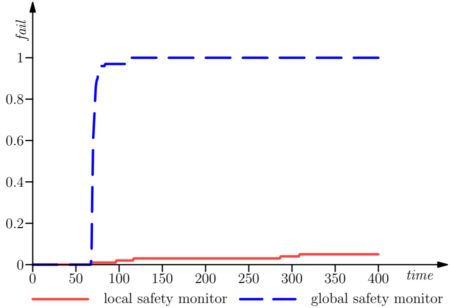
Sample Applications

Crowd Safety

Whenever in safety during an alert, we do not regress:

$$\mathrm{AH}(\mathrm{Y}(\mathit{safe} \wedge \mathit{alert}) \rightarrow (\mathit{safe} \vee \neg \mathit{alert}))$$



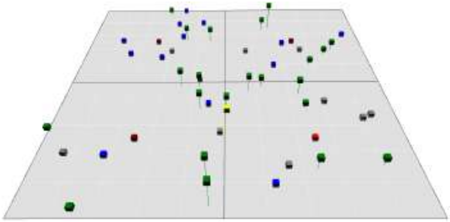


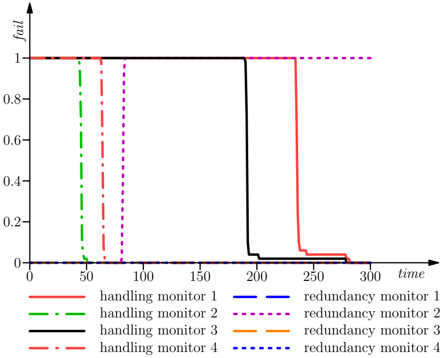
Sample Applications

Drones Recognition

- *Area i gets eventually handled by a drone (liveness):* $\mathrm{EP}\,\mathsf{done}_i$
- *No drone is handling an area that knows to be already handled (safety):*

$$\mathrm{AH}\,\neg(\mathsf{done}_i \wedge \mathrm{EY}(\mathrm{EP}\,\mathsf{done}_i))$$



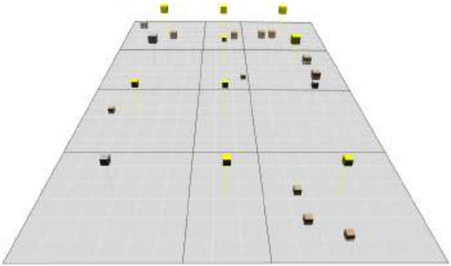


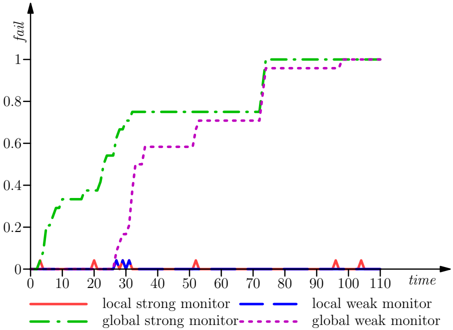
Sample Applications

Smart Home

Lights are on iff some people have been in the immediate vicinity in the close past:

$$\text{light} \to (\text{people} \wedge \mathrm{Y}\,\text{people} \to \text{on}) \wedge (\neg\text{people} \wedge \mathrm{Y}\,\neg\text{people} \to \neg\text{on}).$$





Monitoring Past-CTL in FCPP

<table>
<tr><td>$\top$</td><td><code>true</code></td><td>$\phi_1 \lor \phi_2$</td><td><code>F1 | F2</code></td></tr>
<tr><td>$\bot$</td><td><code>false</code></td><td>$\phi_1 \land \phi_2$</td><td><code>F1 & F2</code></td></tr>
<tr><td>q</td><td><code>q()</code></td><td>$\phi_1 \Rightarrow \phi_2$</td><td><code>F1 <= F2</code></td></tr>
<tr><td>$\neg\phi$</td><td><code>!F</code></td><td>$\phi_1 \Leftrightarrow \phi_2$</td><td><code>F1 == F2</code></td></tr>
<tr><td>$\mathrm{Y}\,\phi$</td><td colspan="3"><code>old(CALL, false, F)</code></td></tr>
<tr><td>$\mathrm{AY}\,\phi$</td><td colspan="3"><code>all_hood(CALL, nbr(CALL, true, F))</code></td></tr>
<tr><td>$\mathrm{EY}\,\phi$</td><td colspan="3"><code>any_hood(CALL, nbr(CALL, false, F))</code></td></tr>
<tr><td>$\phi_1 \mathrm{S}\, \phi_2$</td><td colspan="3"><code>old(CALL, false, [&](bool o){return F2 | (F1 & o);})</code></td></tr>
<tr><td>$\phi_1 \mathrm{AS}\, \phi_2$</td><td colspan="3"><code>nbr(CALL,false,[&](field<bool> o){return F2|(F1&all_hood(CALL,o));})</code></td></tr>
<tr><td>$\phi_1 \mathrm{ES}\, \phi_2$</td><td colspan="3"><code>nbr(CALL,false,[&](field<bool> o){return F2|(F1&any_hood(CALL,o));})</code></td></tr>
<tr><td>$\mathrm{P}\,\phi$</td><td colspan="3"><code>old(CALL, false, [&](bool o){return F | o;})</code></td></tr>
<tr><td>$\mathrm{AP}\,\phi$</td><td colspan="3"><code>nbr(CALL, false, [&](field<bool> o){return F | all_hood(CALL, o);})</code></td></tr>
<tr><td>$\mathrm{EP}\,\phi$</td><td colspan="3"><code>nbr(CALL, false, [&](field<bool> o){return F | any_hood(CALL, o);})</code></td></tr>
<tr><td>$\mathrm{H}\,\phi$</td><td colspan="3"><code>old(CALL, true, [&](bool o){return F & o;})</code></td></tr>
<tr><td>$\mathrm{AH}\,\phi$</td><td colspan="3"><code>nbr(CALL, true, [&](field<bool> o){return F & all_hood(CALL, o);})</code></td></tr>
<tr><td>$\mathrm{EH}\,\phi$</td><td colspan="3"><code>nbr(CALL, true, [&](field<bool> o){return F & any_hood(CALL, o);})</code></td></tr>
</table>

Outline

- Runtime Verification
- Temporal logic
- Spatial logic
  - Monitoring Spatial Properties
  - Spatial Logic of Closure Spaces
  - Sample Applications
  - SLCS in Field Calculus

Monitoring Spatial Properties

- introduce the spatial logic of closure spaces (SLCS) for expressing properties of situated, distributed systems to be monitored
- provide sample applications in smart home and emergency settings
- outline a translation of SLCS formulas into a field calculus monitor for them

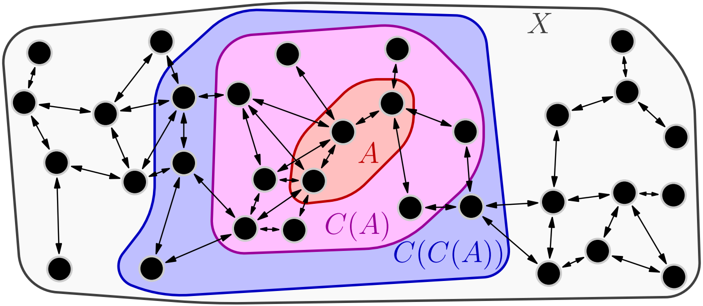
$\neg D \vee (O \Leftrightarrow \mathcal{F}\, P)$

```
!D || (O == somewhere(P))
```

Closure Spaces

Definition

A closure space is a set X with a *closure operator* $C : 2^X \to 2^X$ such that:
$C(\emptyset) = \emptyset$, $A \subseteq C(A)$, $C(A \cup B) = C(A) \cup C(B)$



- generalizes topological spaces by not demanding $C(C(A)) = C(A)$
- contains quasi-discrete spaces for which $C(A) = \bigcup_{x \in A} C(\{x\})$
  $\longrightarrow$ characterised as graphs where $C(v)$ are the neighbours of v

Spatial Logic of Closure Spaces (SLCS)

$$\phi ::= \top \mid q \mid (\neg\phi) \mid (\phi \vee \phi) \mid (\Diamond\,\phi) \mid (\phi\,\mathcal{R}\,\phi)$$

fundamental op.

$\Box\,\phi \triangleq \neg(\Diamond(\neg\phi))$ $\quad \partial\,\phi \triangleq (\Diamond\,\phi) \wedge \neg(\Box\,\phi)$ $\quad \partial^{-}\,\phi \triangleq \phi \wedge \neg(\Box\,\phi)$ $\quad \partial^{+}\,\phi \triangleq (\Diamond\,\phi) \wedge \neg\phi$

$\phi\,\mathcal{T}\,\psi \triangleq \phi\,\mathcal{R}(\Diamond\,\psi)$ $\quad \phi\,\mathcal{U}\,\psi \triangleq \phi \wedge \Box\neg(\neg\psi\,\mathcal{R}\,\neg\phi)$ $\quad \mathcal{F}\,\phi \triangleq \top\,\mathcal{R}\,\phi$ $\quad \mathcal{G}\,\phi \triangleq \neg\,\mathcal{F}\,\neg\phi$

Local modalities

- $\Diamond\,\phi$ (closure) holds at points with some neighbour satisfying ϕ...

Global modalities

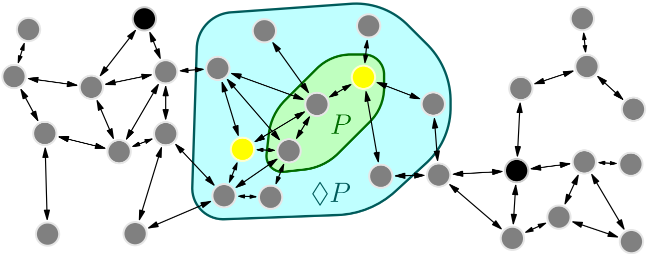
- $\phi\,\mathcal{R}\,\psi$ (reaches) holds at the start of paths satisfying ϕ ending in ψ...

two modalities are fundamental, the rest is derived
($\mathcal{R}$ chosen for presentation convenience)

Sample Smart-Home Applications

Electrical devices are on when people is present

- P true on points who are sensing people
- D true on points which are electrical devices, O true if they are on

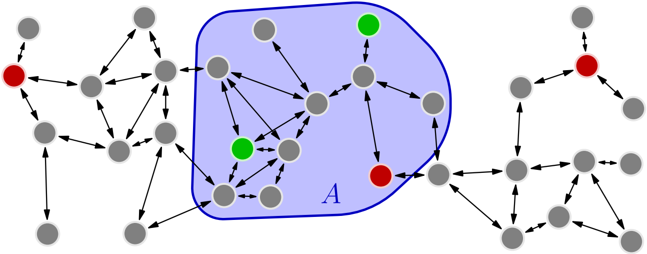


- nearby people only: $D \Rightarrow (O \Leftrightarrow \Diamond P)$
  $\longrightarrow$ if I am an electrical device, I should be on iff a neighbour senses people
- farther away people: $D \Rightarrow (O \Leftrightarrow \mathcal{F}\, P)$
  $\longrightarrow$ if I am an electrical device, I should be on iff anybody senses people

Sample Smart-Home Applications

High stereo level result in agreement on lowering

- L true on stereos with high level
- A true on points agreeing on lowering the volume

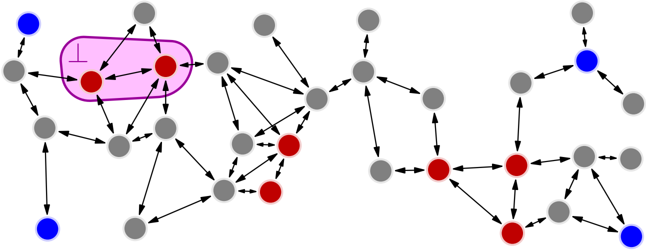


- nearby people only: $L \Rightarrow (\Box A)$
  $\longrightarrow$ if I am an high level stereo, every neighbour agrees on lowering
- farther away people: $L \Rightarrow (\mathcal{G} A)$
  $\longrightarrow$ if I am an high level stereo, everybody agrees on lowering

Sample Emergency Applications

Can reach the internet through non-busy devices: $\neg B \mathcal{R} I$

- B true on busy devices
- I true if device has internet connection



Dangerous areas are surrounded by devices who can reach safely a base:

$$D \Rightarrow (D \mathcal{U} (\neg D \mathcal{R} B))$$

- D true on dangerous areas
- B true if device is on a base

SLCS Translation in Field Calculus

$\top$	`true`	$\phi_1 \vee \phi_2$	`F1 | F2`
$\bot$	`false`	$\phi_1 \wedge \phi_2$	`F1 & F2`
q	`q()`	$\phi_1 \Rightarrow \phi_2$	`F1 <= F2`
$\neg\phi$	`!F`	$\phi_1 \Leftrightarrow \phi_2$	`F1 == F2`
$\Box\,\phi$	`all_hood(CALL, nbr(CALL, F))`		
$\Diamond\,\phi$	`any_hood(CALL, nbr(CALL, F))`		
$\phi_1\,\mathcal{R}\,\phi_2$	`F1 ? somewhere(CALL, F2) : false`		

- `somewhere(F)` if `F` holds in some reachable device computing the function

```
FUN bool somewhere(ARGS, bool F) { CODE return distanceTo(CALL, F) < D; }
```

  - `D` is the network diameter $\longrightarrow$ if closest `F` is farther, it doesn't exist
  - computes distance from closest device where `F` holds
    $\longrightarrow$ optimal strategy but not exact: cannot know things instantaneously

SLCS Translation in FCPP

The translation `P` of a formula ϕ is:

- efficient: resources linear in (formula length) $\times$ (neighbourhood size)
- self-stabilising: if network stops changing, converges to the interpretation of ϕ
- reactive: follows changes with optimal speed
  $\longrightarrow$ provided that the diameter estimate is correct

$\neg D \vee (O \Leftrightarrow \mathcal{F}\, P)$ $\longrightarrow$ `!D || (O == somewhere(P))`