Distributed and cyber physical systems: formal tools and research challenges

Laura Bussi

University of Pisa

05/03/2021

Outline

- Introduction
- Main challenges
- Existing tools
- Possible research directions
- Conclusions

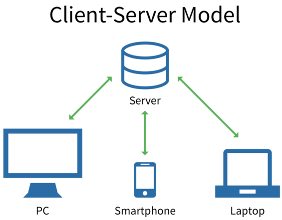
Distributed systems

Distributed systems

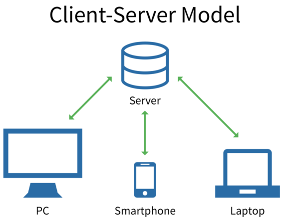
- Multiple components, located on different networked machines.
- Communicate via message passing.
- Coordinate in order to appear as a single, coherent system.
- Various size, architectures...

Examples?

Examples?



Examples?



Cyber physical systems (from the NIST website)

Cyber physical systems (from the NIST website)

- Cyber-Physical Systems (CPS) comprise interacting digital, analog, physical, and human components engineered for function through integrated physics and logic.

Cyber physical systems (from the NIST website)

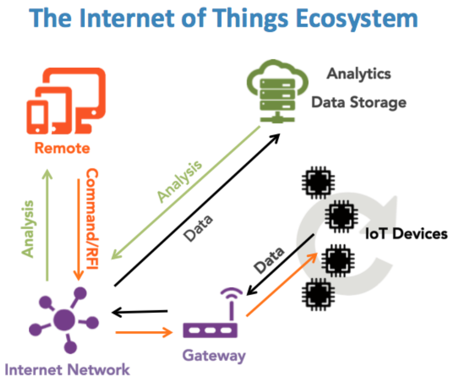
- Cyber-Physical Systems (CPS) comprise interacting digital, analog, physical, and human components engineered for function through integrated physics and logic.
- Other phrases that you might hear when discussing these and related CPS technologies include:
  - Internet of Things (IoT)
  - Industrial Internet
  - Smart Cities
  - Smart Grid

Cyber physical systems (from the NIST website)

- Cyber-Physical Systems (CPS) comprise interacting digital, analog, physical, and human components engineered for function through integrated physics and logic.
- Other phrases that you might hear when discussing these and related CPS technologies include:
  - Internet of Things (IoT)
  - Industrial Internet
  - Smart Cities
  - Smart Grid
  - "Smart" Anything

Examples?

Examples?


The Internet of Things Ecosystem
Remote
Analytics
Data Storage
Analysis
Command/RFI
Analysis
Data
IoT Devices
Data
Internet Network
Gateway

Why should we use formal methods?

Why should we use formal methods?

- Needed to ensure trustworthiness of systems:
  - safety, security...

Why should we use formal methods?

- Needed to ensure trustworthiness of systems:
  - safety, security...
- Using formal methods *guarantees* that certain properties hold on a given system

Why should we use formal methods?

- Needed to ensure trustworthiness of systems:
  - safety, security...
- Using formal methods *guarantees* that certain properties hold on a given system
- Problems?
  - efficiency, usability

But...

Engineers use TLA+ to prevent serious but subtle bugs from reaching production.

BY CHRIS NEWCOMBE, TIM RATH, FAN ZHANG, BOGDAN MUNTEANU, MARC BROOKER, AND MICHAEL DEARDEUFF

How Amazon Web Services Uses Formal Methods

S3 is just one of many AWS services that store and process data our customers have entrusted to us. To safeguard that data, the core of each service relies on fault-tolerant distributed algorithms for replication, consistency, concurrency control, auto-scaling, load balancing, and other coordination tasks. There are many such algorithms in the literature, but combining them into a cohesive system is a challenge, as the algorithms must usually be modified to interact properly in a real-world system. In addition, we have found it necessary to invent algorithms of our own. We work hard to avoid unnecessary complexity, but the essential complexity of the task remains high.

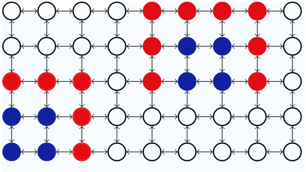
Complexity increases the probability of human error in design, code, and operations. Errors in the core of the system could cause loss or corruption of data, or violate other interface contracts on which our customers depend. So, before launching a service, we need to reach extremely high confidence that the core of the system is correct. We have found the standard

The magical recipe

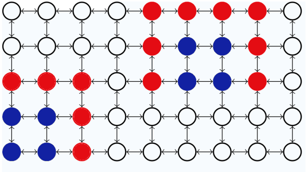
The magical recipe

- Provide good theory
- Support theory with efficient and usable tools

Spatio-temporal properties

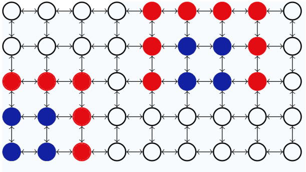


Spatio-temporal properties



- *blue near red*

Spatio-temporal properties



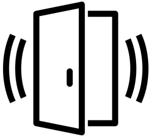
- *blue near red*
- *eventually, at least a blue point will become green*

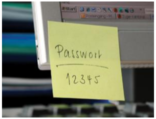
Dealing with physical phenomena

Dealing with physical phenomena

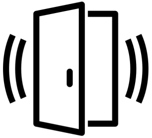
- Discrete vs continuous
- 3D (e.g. buildings) vs 2D (e.g. links)

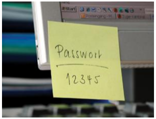
3D vs 2D





3D vs 2D





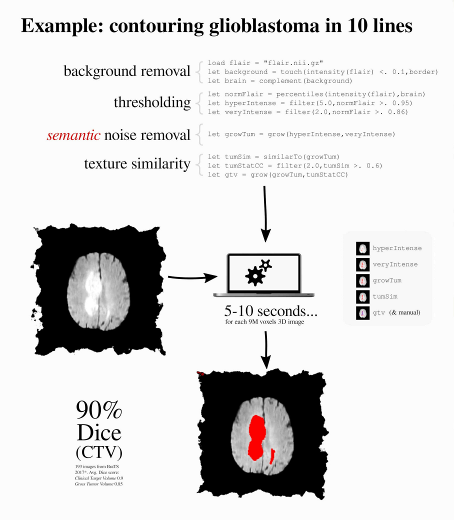
- *What could possibly go wrong?*

SLCS

SLCS

- Spatial Logic for Closure Spaces
- Used to specify and verify properties over graphs
- Includes "one step" operators, as well as more complex ones (e.g. Reachability)

Example


Example: contouring glioblastoma in 10 lines
background removal
load flair = "flair.nii.gz"
let background = touch(intensity(flair) <. 0.1,border)
let brain = complement(background)
thresholding
let normFlair = percentiles(intensity(flair),brain)
let hyperIntense = filter(5.0,normFlair >. 0.95)
let veryIntense = filter(2.0,normFlair >. 0.86)
semantic noise removal
let growTum = grow(hyperIntense,veryIntense)
texture similarity
let tumSim = similarTo(growTum)
let tumStatCC = filter(2.0,tumSim >. 0.6)
let gtv = grow(growTum,tumStatCC)
hyperIntense
veryIntense
growTum
tumSim
gtv (& manual)
5-10 seconds...
for each 9M voxels 3D image
90%
Dice
(CTV)
193 images from BraTS 2017*. Avg. Dice score:
Clinical Target Volume 0.9
Gross Tumor Volume 0.85

VoxLogicA

- Voxel based image analyser
- Interprets ImgQL commands
- Model checks properties over images

TLA+

TLA+

- Temporal Logics of Actions
- Used to design, model, document, and verify concurrent and distributed systems
- Include set operators (safety) and temporal logics operators (liveness)

Example

All possible actions

$$
\begin{aligned}
Next \triangleq\ & \\
& \lor\ CloseDoor \\
& \lor\ LightOn \\
& \lor\ OpenDraw \\
& \lor\ ClosePanel
\end{aligned}
$$

Specification of the entire system

$$Spec \triangleq Init \land \Box[Next]_{\langle panel,\, light,\, draw,\, door,\, state \rangle}$$

Specification never violates the type invariance

THEOREM $Spec \Rightarrow \Box TypeInv$

The panel and door are never both unlocked in the same time

$$
\begin{aligned}
Inv \triangleq\ & \\
& \lor\ panel = \text{“unlocked”} \Rightarrow door = \text{“locked”} \\
& \lor\ door = \text{“unlocked”} \Rightarrow panel = \text{“locked”}
\end{aligned}
$$

The TLA Toolbox

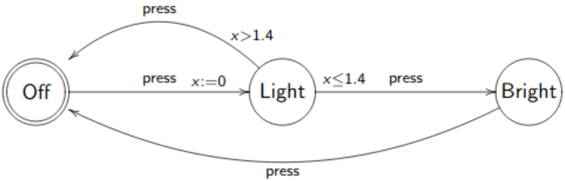
- IDE (built on top of Eclipse)
- Model checker
- Proof system

Timed automata

Timed automata

- Finite-state automata extended with a finite set of real-valued clocks.
- Time is considered a continuous quantity.
- Allows for modeling of real-time systems.

Example



UPPAAL

- Specification and verification of real-time systems.
- These are modeled as networks of timed automata.

Spatio-temporal properties

- Idea: extend SLCS to specify spatio-temporal properties
- This would allow to deal with dynamic graphs
- How?

Spatio-temporal properties

- Idea: extend SLCS to specify spatio-temporal properties
- This would allow to deal with dynamic graphs
- How?
  - By mean of existential quantifier and temporal operators
  - This implies the need for new tools
  - Possibly exploit parallel devices (e.g. GPUs ☺)

3D vs 2D

- Idea: define a "hybrid space"
- Give the sintax and semantics of a specification language.

What about automata?

- Timed automata are an interesting subclass of *hybrid automata*
- Hybrid automata are very expressive...
- ...then they are not decidable (of course!)
- Are there other interesting subclasses of hybrid automata?

Conclusions

Conclusions

- There are many possibilities to explore.

Conclusions

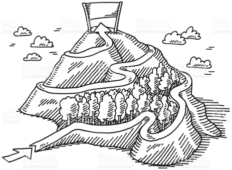
- There are many possibilities to explore.
- Design and verification of distributed and cyber physical systems need to be supported by formal tools.

Conclusions

- There are many possibilities to explore.
- Design and verification of distributed and cyber physical systems need to be supported by formal tools.
- This can be done by providing good theory and taking a leaf from working tools.

Conclusions

- There are many possibilities to explore.
- Design and verification of distributed and cyber physical systems need to be supported by formal tools.
- This can be done by providing good theory and taking a leaf from working tools.



References

- *https://www.nist.gov/el/cyber-physical-systems*
- C. Newcombe et al. - *How Amazon Web Services uses formal methods*
- V. Ciancia et al. - *VoxLogicA: a Spatial Model Checker for Declarative Image Analysis*
- *https://lamport.azurewebsites.net/tla/tla.html*
- R. Alur - *Timed automata*
- *https://uppaal.org/*

Contacts

- laura.bussi@phd.unipi.it
- laura.bussi@isti.cnr.it
- Room 344, Dept. of CS