Quantitative Verification of Adaptive IT Systems

Radu Calinescu

Aston University

Joint work with Marta Kwiatkowska, Carlo Ghezzi, Raffaela Mirandola, Giordano Tamburrelli and Lars Grunske


sigmaspectrum
ult General Care
Amiodarone
0.5
mg/min
GHI 3
JKL 4
MNO 5
PQR 6
STU 7
VWX 8
YZ 9

The New York Times

Business Day

F.D.A. Steps Up Oversight of Infusion Pumps

(23 April 2010)

"Over the last five years, the agency says it has received reports of 710 patient deaths linked to problems with the devices [...]

Some of those deaths involved patients who suffered drug overdoses [...] because the device's software malfunctioned.

Manufacturers [...] issued 79 recalls, among the highest for any medical device."

"Pump producers now typically conduct 'simulated' testing of its devices by users to identify bugs."

freepatentsonline
all the inventions of mankind

SEARCH:

FIND

GO TO ADVANCED SEARCH

HOME | SEARCH PATENTS | CHEMICAL SEARCH

SAFETY LIMITS FOR CLOSED-LOOP INFUSION PUMP CONTROL

United States Patent Application 20110034909 **Kind Code:** A1

A system and process for providing safety limits on the delivery of an infusion formulation by an infusion pump system in response to a sensed biological state. The safety limits may comprise user-initiated event signals corresponding to events that may significantly affect the biological state. The safety limits may further comprise user-initiated event ranking signals for respective events which specify a degree, quantity, or measure for the respective event. The user-initiated event and event ranking signals may be communicated to a computing element associated with

Two stringent requirements for IT systems

IT systems are increasingly used in safety- and business-critical applications

- they must operate in predictable ways & achieve high availability

IT systems are increasingly used in applications characterised by continual change in state, environment and requirements

- they must adapt to change

Motivation

Predictability/high availability & *adaptiveness* have typically been studied in isolation, by two research communities

- formal methods
- autonomic/context-aware/self-managing computing

Motivation

Predictability/high availability & *adaptiveness* have typically been studied in isolation, by two research communities

- formal methods
- autonomic/context-aware/self-managing computing

Quantitative verification of adaptive IT systems integrates techniques from both research areas

- quantitative modelling, analysis and verification used **on-line**, as part of the adaptation process

Outline

Quantitative verification

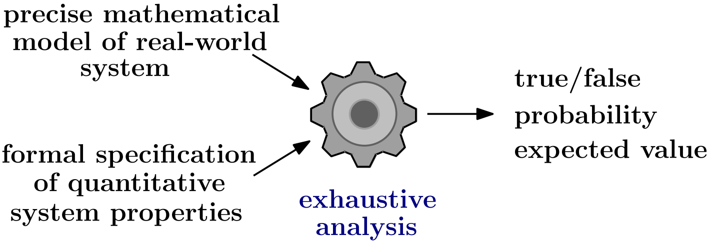
Formal technique for establishing quantitative properties of systems that exhibit probabilistic or real-time behaviour

- probability of system being up $\geq 99.9\%$ of the time
- expected length of request queue for a service ≤ 10

Quantitative verification

Formal technique for establishing quantitative properties of systems that exhibit probabilistic or real-time behaviour

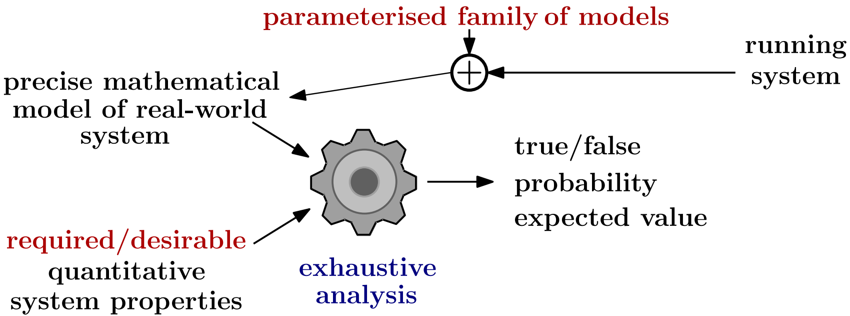
- probability of system being up $\geq 99.9\%$ of the time
- expected length of request queue for a service ≤ 10



On-line quantitative verification: violation detection

Verification of required/desirable quantitative properties is performed at runtime

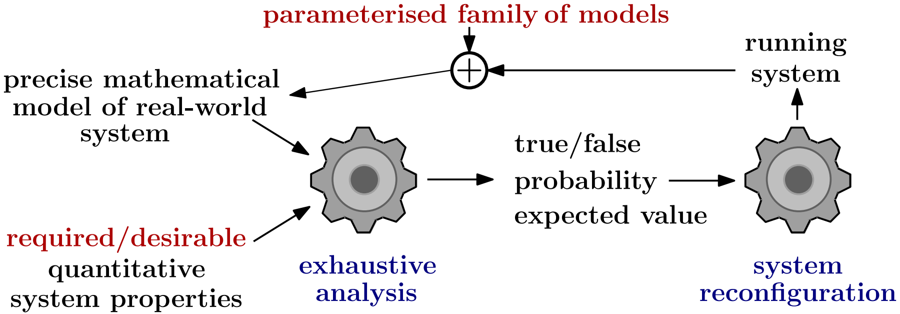
- analysed model selected based on actual system state
-



On-line quantitative verification: adaptation

Verification of required/desirable quantitative properties is performed at runtime

- analysed model selected based on actual system state
- verification results used to adjust system configuration



Realisation of the approach

The approach was applied in several application domains by using

- models: Markov chains, Markov decision processes
- properties: expressed in probabilistic variants of temporal logic
- verification: probabilistic model checker PRISM
- development platform: GPAC (General-Purpose Autonomic Computing)
- applications: see later slides

Discrete-/continuous-time Markov chains

$$\mathrm{DTMC} = (S, s_{\mathrm{init}}, \mathbf{P}, L)$$

labelling function, $L : S \rightarrow 2^{AP}$

transition probability matrix, $\mathbf{P} : S \times S \rightarrow [0, 1]$

initial state, $s_{\mathrm{init}} \in S$

finite set of states

Discrete-/continuous-time Markov chains

$$\mathrm{DTMC} = (S, s_{\mathrm{init}}, \mathbf{P}, L)$$

labelling function, $L : S \rightarrow 2^{AP}$

transition probability matrix, $\mathbf{P} : S \times S \rightarrow [0, 1]$

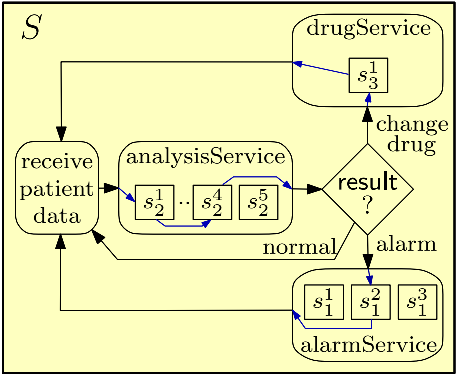
initial state, $s_{\mathrm{init}} \in S$

finite set of states

$$\mathrm{CTMC} = (S, s_{\mathrm{init}}, \mathbf{R}, L)$$

transition rate matrix, $\mathbf{R} : S \times S \rightarrow R_{+}$

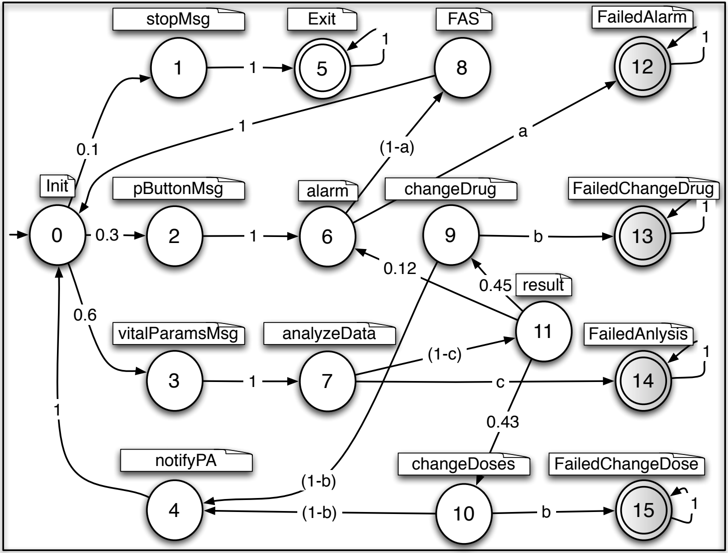
Example: patient monitoring workflow



Each operation can be carried out by one or a combination of several functionally-equivalent third-party services

- services are characterised by different (and potentially changing) success rates, response times, costs, etc.

Example: patient monitoring workflow



Quantitative property specification

PCTL—Probabilistic Computational Tree Logic for DTMCs

$$\phi ::= \textit{true} \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \mathrm{P}_{\bowtie p}[\mathrm{X}\phi] \mid \mathrm{P}_{\bowtie p}[\phi \mathrm{U}^{\leq k} \phi] \mid \mathrm{P}_{\bowtie p}[\phi \mathrm{U} \phi]$$

CSL—Continuous Stochastic Logic for CTMCs

$$\phi ::= \textit{true} \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \mathrm{P}_{\bowtie p}[\mathrm{X}\phi] \mid \mathrm{P}_{\bowtie p}[\phi \mathrm{U}^{\mathrm{I}} \phi] \mid \mathrm{S}_{\bowtie p}[\phi]$$

- P is the *probabilistic operator* and S is the *steady-state operator*
- $a \in AP$, $p \in [0, 1]$ is a probability, $\bowtie \in \{<, >, \leq, \geq\}$ is a relational operator, $k \in \mathbb{N}$ and $\mathrm{I} \subseteq R_{\geq 0}$ is a time interval

Example: patient monitoring workflow

Reliability requirements:

- $alarm \rightarrow \mathrm{P}_{\leq 0.005}[\mathrm{X}\ failedAlarm]$: *The probability that an attempt to raise the alarm fails is less than* 0.005.
- $\mathrm{P}_{\leq 0.14}[\mathrm{true}\ \mathrm{U}\ failure]$: *The probability that a service failure ever occurs during the lifetime of the system is less than* 0.14.
- …

Performability requirements:

- $\mathrm{P}_{\leq 0.2}[q/Q_{max} > 0.75]$: *The probability that the number of pending changeDrug requests exceeds 75% of the request queue capacity for service DrugService*1 *in the steady state is less than* 0.2.
- …

MAPE control loop

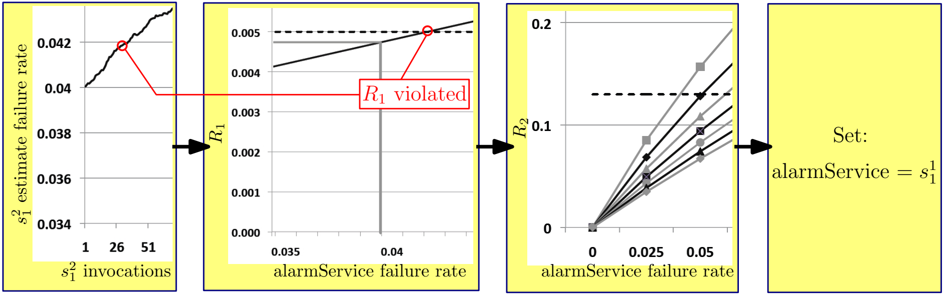
Monitor → Analyse → Plan → Execute

Monitor	Analyse	Plan	Execute
update model	re-verify properties	analyse possible configurations	select suitable configuration

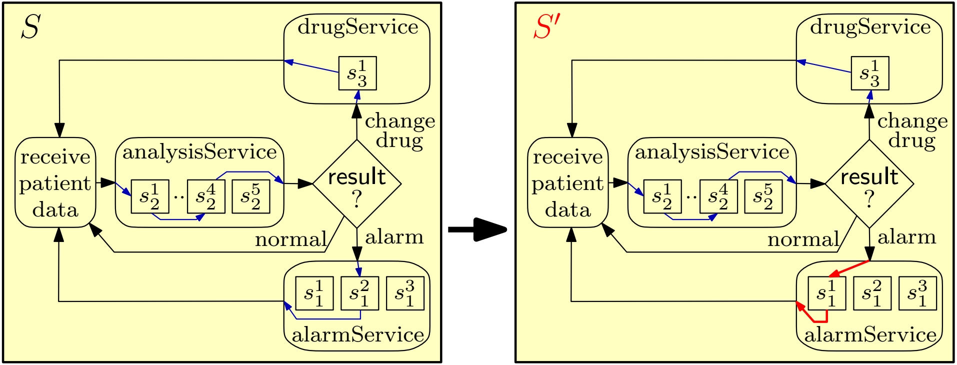
Example: patient monitoring workflow

Monitor → Analyse → Plan → Execute

update model | re-verify properties | analyse possible configurations | select suitable configuration



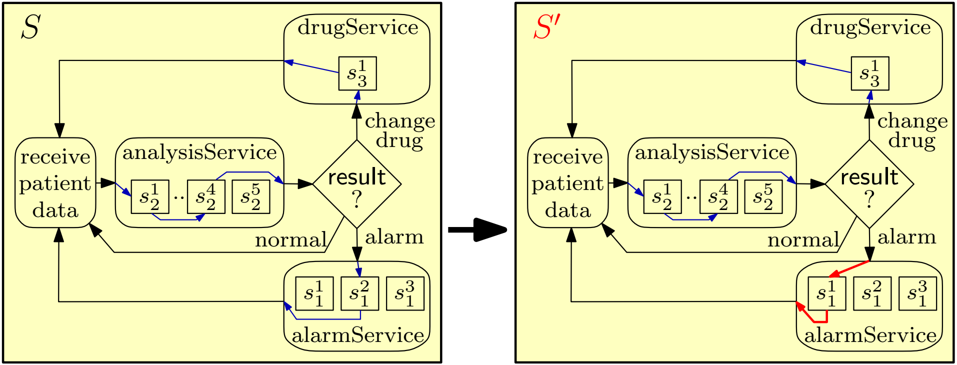
Example: patient monitoring workflow



Scalability

- up to a few seconds, acceptable for many of today's workflows
- verification time grows linearly with the number of requirements, and exponentially with the number of operations and services

Example: patient monitoring workflow



For details, see Calinescu, Grunske, Kwiatkowska, Mirandola & Tamburrelli, Dynamic QoS Management and Optimisation in Service-Based Systems, *IEEE Trans. Software Eng.*, 2010

Some other applications

- Adaptive allocation of data-center servers to clusters to achieve SLA-specified availability in the presence of failures and variable workloads (Calinescu & Kwiatkowska, FASE 2009)
- Adaptive power management of disk drives (Calinescu & Kwiatkowska, ICSE 2009)
- Other uses of formal methods to support predictable adaptation in IT systems (Calinescu, Kikuchi & Kwiatkowska, 2011)

Research challenges

Challenges

Expert knowledge required to produce "good" models

- more models should be built as part of the system development process

Challenges

Expert knowledge required to produce "good" models

- more models should be built as part of the system development process

Mastering of temporal logics needed to get requirements right

- requirements should be specifiable in close-to-natural language

Challenges

State-space explosion

- techniques needed to reduce analysis time & overheads

Challenges

State-space explosion

- techniques needed to reduce analysis time & overheads

E.g., *incremental verification*

- changes are often incremental: use verification results for one model to speed up the analysis of the next

Challenges

State-space explosion

- techniques needed to reduce analysis time & overheads

E.g., *assume-guarantee*

- fits well system-of-systems nature of targeted adaptive systems

Challenges

Tools not intended for runtime use

- use command-line interfaces (lower-level APIs better/faster)

Local optima (unless all possible configurations verified)

- off-line assessment to ensure solution is effective

Monitoring/learning require to update/refine system model

- on-line machine learning techniques combining a priori knowledge with observed system behaviour

Summary

Increasing need for computer systems to adapt in predictable, dependable ways to changes in their state, goals & environment

Runtime quantitative verification can contribute to achieving such adaptiveness in certain scenarios

- so far for small/medium-sized computer systems
- promising results that larger systems can be handled started to emerge

Interesting work required to address research challenges

- exploring and improving scalability
- identifying new applications
- developing high-level language(s) for expressing system goals

Thank you

Questions?