Model Checking

A short introduction

Alberto Molinari
Università degli Studi di Udine,
Dipartimento di Matematica, Informatica e Fisica

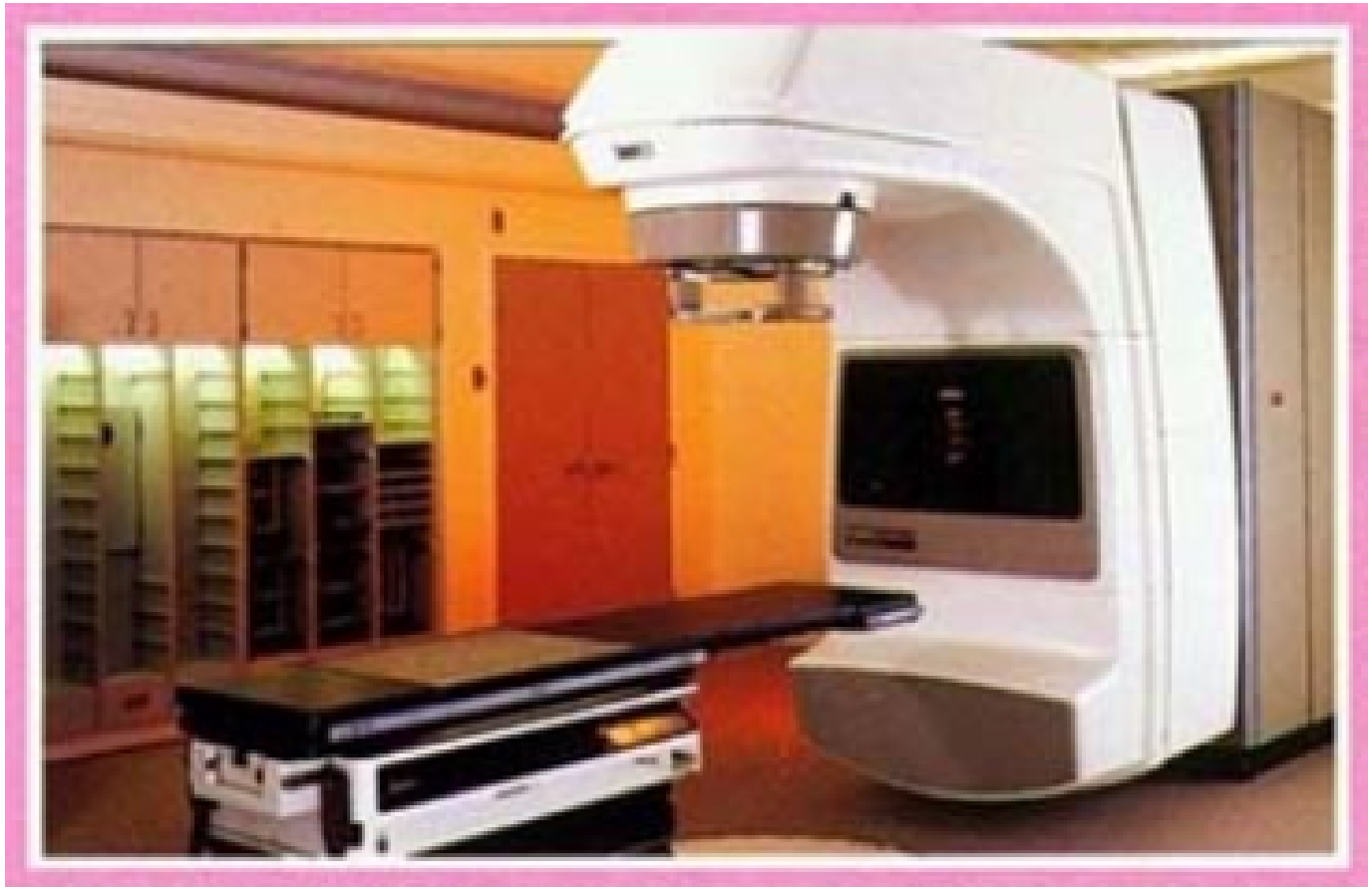
Jan 13, 2017

IT systems everyday

- Our reliance on IT systems (Information and Communication Technology) is growing rapidly
- Smart cards, smartphones, smart TVs, telephone switches/networks, Internet, bio-medical systems/devices, traffic control systems, ...
- however...

Examples of bugs and failures—Therac-25

- Software flaw in the control part of the radiation therapy machine Therac-25 ⇒ death of 6 cancer patients between 1985 and 1987 by overdose of radiation



Examples of bugs and failures—Pentium

- The bug in Intel's Pentium floating-point division unit (1994) ⇒ loss of about $ 475 million to replace faulty processors

Examples of bugs and failures—Denver airport

- Software error in a baggage handling system postponed the opening of Denver airport for 9 months (1990) ⇒ loss of $ 1.1 million per day

Examples of bugs and failures—Schiapparelli Mars Lander

From www.esa.int/Our_Activities/Space_Science/ExoMars/Schiaparelli_landing_investigation_makes_progress

[...] saturation [...] of the Inertial Measurement Unit had occurred shortly after the parachute deployment. [...] When merged into the navigation system, the erroneous information generated an estimated altitude that was negative—that is, below ground level. This in turn successively triggered a premature release of the parachute [...], a brief firing of the braking thrusters and finally activation of the on-ground systems as if Schiaparelli had already landed. In reality, the vehicle was still at an altitude of around 3.7 km.

System verification

- The reliability of IT systems is a key issue...
- ...starting from the system design process!
- System verification is used to establish that the design or product under consideration possesses certain properties

System verification

- The reliability of IT systems is a key issue...
- ...starting from the system design process!
- System verification is used to establish that the design or product under consideration possesses certain properties

System verification

- The reliability of IT systems is a key issue...
- ...starting from the system design process!
- System verification is used to establish that the design or product under consideration possesses certain properties

Testing

- Testing is a dynamic technique that actually runs the software.
- Correctness determined by forcing the software to traverse a set of execution paths
- As exhaustive testing of all execution paths is practically infeasible: testing can thus never be complete, i.e., it can only show the presence of errors, not their absence

Testing

- Testing is a dynamic technique that actually runs the software.
- Correctness determined by forcing the software to traverse a set of execution paths
- As exhaustive testing of all execution paths is practically infeasible: testing can thus never be complete, i.e., it can only show the presence of errors, not their absence

Formal methods

- Formal methods: “the applied mathematics for modeling and analyzing IT systems”
- Aim: to establish system correctness with mathematical rigor

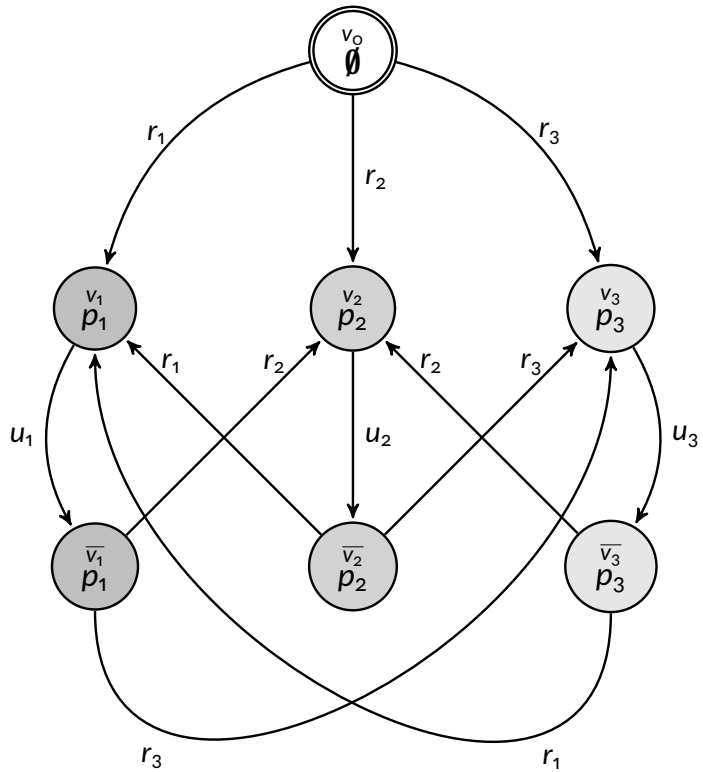
Formal methods

- Formal methods: “the applied mathematics for modeling and analyzing IT systems”
- Aim: to establish system correctness with mathematical rigor

Model checking

Model checking: the desired properties of a system are checked against a model of the system

- the model is a (finite) state-transition graph: how the system behaves
- system properties are specified by a temporal logic: what the system should do, and what it should not do



Model checking

Distinctive features of model checking:

- exhaustive verification of all the possible behaviours
- fully automatic process
- a counterexample is produced for a violated property: modeling error/design error?
- state spaces $\sim 10^{30}$ (up to even 10^{476} states can be handled for specific problems)

Typical properties

- functional correctness (does the system do what it is supposed to do?)
- safety ("something bad never happens")
- liveness ("something good will eventually happen")
- fairness (does, under certain conditions, an event occur repeatedly?)
- absence of deadlock states
- real-time properties (is the system acting in time?)
- ...

Strengths of model checking

- It is complete
- It provides diagnostic information in case a property is invalidated: useful for debugging purposes
- General verification approach
- Supports partial verification, i.e., no complete requirement specification is needed

Weaknesses of model checking

- It verifies a system model, and not the actual system! ⇒ any obtained result is thus as good as the system model ⇒ complementary techniques, such as testing, are needed to find fabrication faults (for hardware) or coding errors (for software)
- Mainly appropriate to control-intensive applications, less suited for data-intensive applications
- State-space explosion problem
- Requires some expertise in finding appropriate abstractions