Temporal Logic

Natasha Alechina Brian Logan

Utrecht University

n.a.alechina@uu.nl b.s.logan@uu.nl

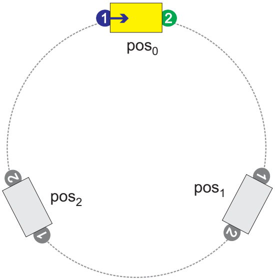
Reasoning about Time

- this lecture is about temporal logics (in general)
- temporal logic formulas describe how the system evolves
- interpreted over state transition systems

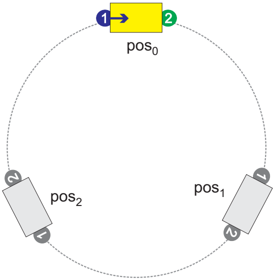
State transition systems

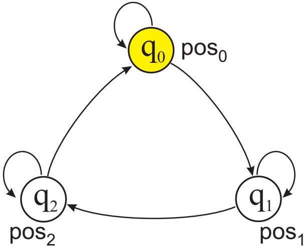
- propositional logic formulas are evaluated with respect to assignments
- first order logic formulas are evaluated with respect to relational structures
- temporal logic formulas are evaluated with respect to state transition systems
- states correspond to particular values of system variables, transitions to changes in these values (due to some actions or events)
- propositional temporal logic describes properties of states using boolean propositions

Example: a system represented as a state transition system



Example: a system represented as a state transition system





Definition of a state transition system

Definition (State Transition System)

An state transition system is a triple

$$\langle St, \longrightarrow, \mathcal{V} \rangle$$

where:

- St is a non-empty set of states,
- $\longrightarrow \subseteq St \times St$ is a transition relation
- $\mathcal{V} : \mathcal{PV} \rightarrow 2^{St}$ is a valuation function that assigns to each proposition a set of states where it is true
  ($\mathcal{PV}$ is the set of propositions).

Temporal Operators

typical temporal operators:

$X\varphi$	φ is true in the next moment in time
$G\varphi$	φ is true in all future moments
$F\varphi$	φ is true in some future moment
$\varphi\, U\, \psi$	φ is true until the moment when ψ becomes true

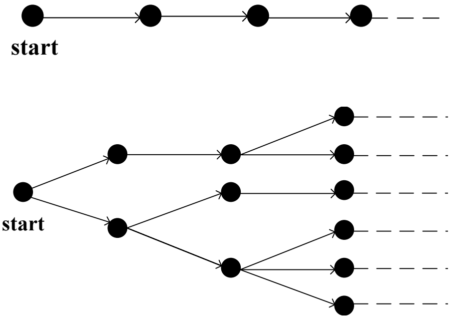
example formulas:

$G((\neg \text{passport} \vee \neg \text{ticket}) \;\rightarrow\; X\neg \text{board\_flight})$

$\text{send}(\text{msg}, \text{rcvr}) \;\rightarrow\; F\text{receive}(\text{msg}, \text{rcvr})$

Models of Time

- transition relation represents time
- models of time: linear vs. branching



Paths

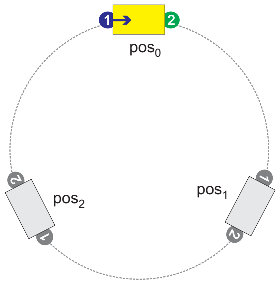
Definition (Paths in a transition system)

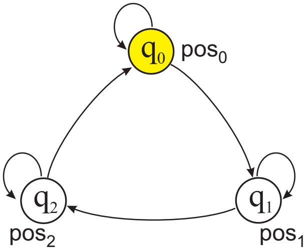
A path λ in $\langle St, \longrightarrow, \mathcal{V}\rangle$ is a sequence of states from St, $q_0 q_1 q_2 \ldots$, such that for each q_i and q_{i+1}, $q_i \longrightarrow q_{i+1}$.

A path must be full, i.e. either infinite, or ending in a state with no outgoing transition.

In this course, unless stated otherwise, each state has an outgoing transition and the paths are infinite
in the coursework, we introduce a finite path version of one of the logics

Example: Robot and Carriage



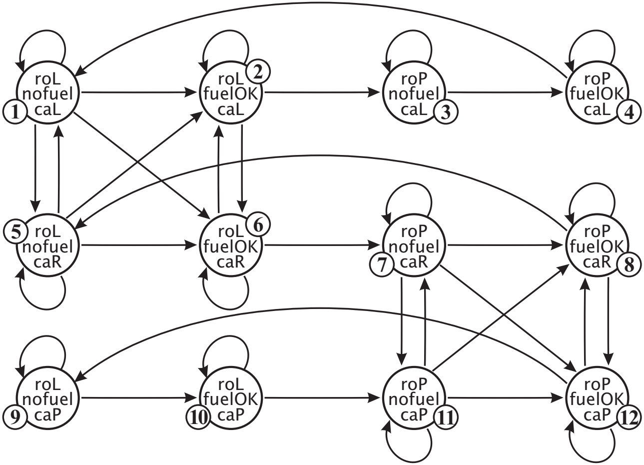


A possible path: $q_0 q_0 q_1 q_2 q_2 q_2 \ldots$

Example: Rocket and Cargo

- a rocket and a cargo,
- the rocket can be moved between London (proposition roL) and Paris (proposition roP),
- the cargo can be in London (caL), Paris (caP), or inside the rocket (caR),
- the rocket can be moved only if it has its fuel tank full (fuelOK),
- when it moves, it consumes fuel, and nofuel holds after each flight.

Example: Rocket and Cargo



Reasoning about Time: Specification Templates

temporal logic was originally developed in order to represent tense in natural language

in Computer Science it became widely used for the formal specification and verification of concurrent and distributed systems

useful concepts can be formally, and concisely, specified using temporal logics, e.g.:

- safety properties
- liveness properties
- fairness properties

Reasoning about Time: Safety Properties

Safety / maintenance:

"something bad will not happen"
"something good will always hold"

typical examples:

$G\,\neg\text{deadlock}$

$G\,(\text{fuelOK} \vee X\,\text{fuelOK})$

usually: $G\,\neg \ldots$

Reasoning about Time: Liveness Properties

Liveness / achievement:

"something good will happen"

typical examples:

$F\,$goal

$FG\,$retired

requested $\rightarrow F\,$granted

usually: $F \ldots$

Reasoning about Time: Fairness Properties

Fairness / service:

> *"Whenever something is attempted/requested, then it will be successful/granted"*

typical examples:

$G\,F\,\text{rain}$

$G\,(\text{attempt} \rightarrow F\,\text{success})$

$(G\,F\,\text{attempt}) \rightarrow (G\,F\,\text{success})$

usually: $G\,F\ldots$

Fairness properties:

- useful when scheduling processes, etc.
- specifying properties of the environment

Logical Problems

- Decision problem: given representation of an instance, decide whether it belongs to the set of "good" instances
- Typical logical problems: validity, satisfiability, and model checking

Logical Problems

Validity: given formula φ, determine if φ is valid (true in every model)

usually: have an axiomatic theory (e.g. ethical rules for robots) and ask whether some safety property is provable in this theory/valid in all models of the theory

for example: $G\neg\text{kills\_people}$

Logical Problems: Satisfiability

Satisfiability: given formula φ, determine if φ is satisfiable (true in some model)

usually: is an axiomatic theory consistent (is the conjunction of axioms satisfiable in some model)?

or: does Theory logically entail φ? This can be checked by asking instead: is Theory $\wedge \neg\varphi$ satisfiable?

Logical Problems: Model Checking

Model checking: given formula φ and model M, determine if φ is true in M

this is what we will mostly study in this course

Decision Problems

Decision problem is a Yes/No question. Given an instance of the problem (e.g. input formula, model) return Yes/No.

When we want some object returned (e.g. a witness or a counterexample): function form of a decision problem.

Complexity classes: sets of decision problems that require the same computational resources for solving them

Some Complexity Classes

- P (polynomial time): problems solvable in polynomial time by a deterministic Turing machine
- NP (polynomial nondeterministic time): problems solvable in polynomial time by a non-deterministic Turing machine
- co-NP: problems whose complement (swap yes/no answer) is in NP
- PSPACE (polynomial space): problems solvable by a Turing machine with a polynomially bounded tape (tape can be reused!)
- EXPTIME (exponential time): problems solvable in exponential time by a deterministic Turing machine