# Model checking for trains

Benjamin Bordais, Thomas Mari, Julie Parreaux
supervised by
Nathalie Bertrand, Loïc Hélouët, Ocan Sankur

ENS Rennes

28 janvier 2018

# Introduction

Every day, millions of people take the train !

- Unpredictable events may cause some delays, we need to catch them up.
- This is done by adapting the speed profile of each train.
- Regulation policy : automatic tool that gives instructions in real time when changes occur in the traffic.

# Introduction

- Need to check effectiveness of regulation policies
- Model checkers : help in the evaluation of regulation policies
- They need a formal model

# Sommaire

# Sommaire

# Formalism

We need a formal model to represent rail networks.

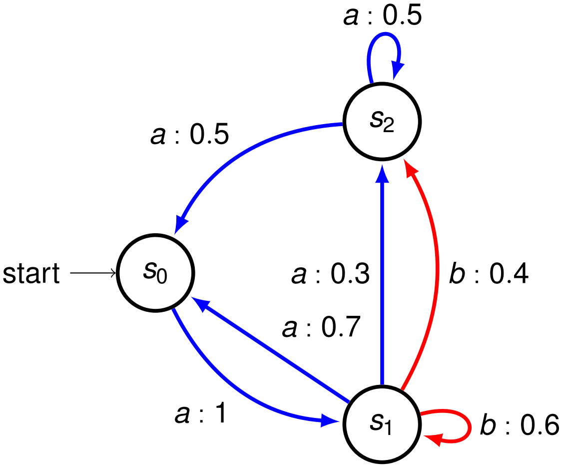
### Need for randomness

Delays are unpredictable and conveniently represented through probabilities.

### Need for nondeterminism

Trains can accelerate or slow down. These changes of speed need to be depicted by our model.

# Model : Markov Decision Process (MDP)



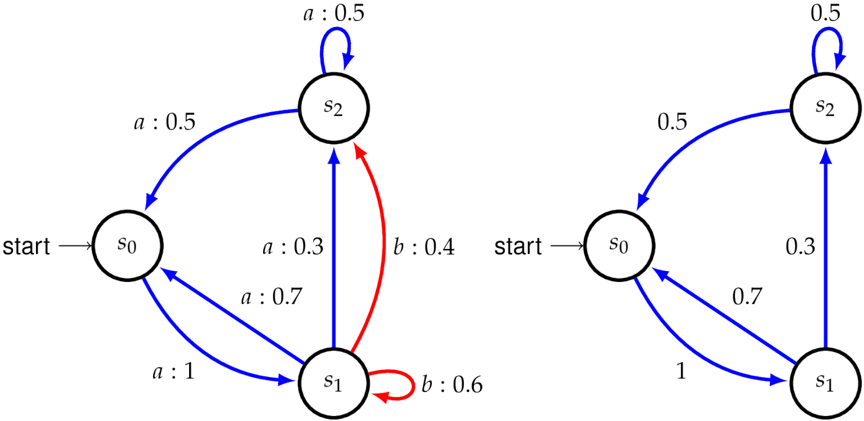

An example of an MDP

# Regulation policy

Regulation policy : choose the speed of train according to the state of the system.

- A regulation policy resolves the nondeterminism of an MDP
- The model induced by an MDP and a policy is a Discrete Time Markov Chain (DMTC)

# Discrete Time Markov Chain (DTMC)




MDP
+ regulation policy
(e. g. always choose a)

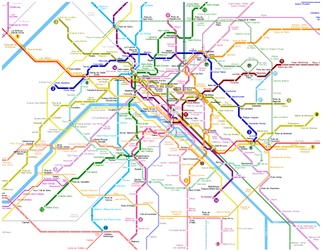
=

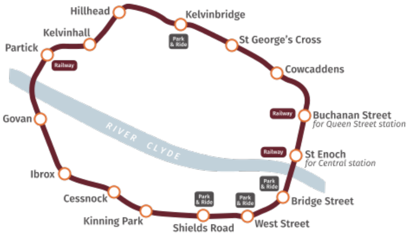
DTMC

# Sommaire

# Topology

- Real system are often too complex for formalization
- The simpler the system, the simpler the model !
- First, we study a ring system



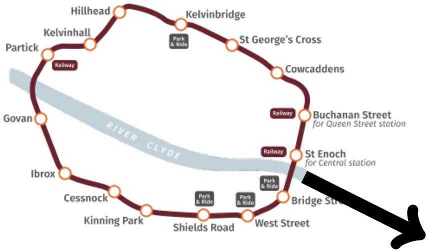


# Space and Time Discretization

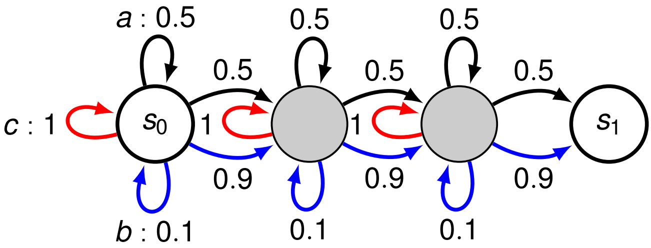
Real system $=$ continuous $\neq$ discrete $=$ model.
Time, space are discretized.

- Step by step evolution : at each time step a transition is taken
- States $=$ stations.
  Distance between two stations $=$ number of intermediate points

# A toy example




- Action $a$ : medium speed
- Action $b$ : higher speed
- Action $c$ : dwell



# Sommaire

# PCTL logic

The PCTL [1] [2] logic uses sevral connectors :

- The usual connectors of propositional logic
- Temporal connectors :
  - Next : $X\ \phi$
  - Eventually : $F\ \phi$
  - Bounded eventually : $F^{\leq n}\ \phi$
- A probabilistic connector : $P_{\alpha\ p}$ with $\alpha \in \{\leq, <, \geq, >\}$ and $p \in [0, 1]$

1. A logic for reasoning about time and reliability, Hanson et al., 1994
2. Automatic Verification of Finite-state Concurrent Systems Using Temporal Logic Specifications, Clarke et al., 1986

# PCTL logic

- Two trains never collide :

$$\phi = P_{\leq 0}(F\ \phi_{collision})$$

- If a train has some delays it will catch it up within 10 steps with a high probability :

$$\phi = \phi_{delay} \Rightarrow P_{\geq 0.9}(F^{\leq 10}\ \neg\phi_{delay})$$

# Model checking

Automatic tools : prove that a model with its regulation policy satisfies requirements.
Model-checkers PRISM [3] and Storm [4] :

- Check that a model verifies some properties
- Synthesize a regulation policy that meets some properties

3. PRISM 4.0 : Verification of Probabilistic Real-time Systems, Kwiatkowska et al. 2011
4. A storm is Coming : A Modern Probabilistic Model Checker, Dehnert et al., 2017

# Abstraction

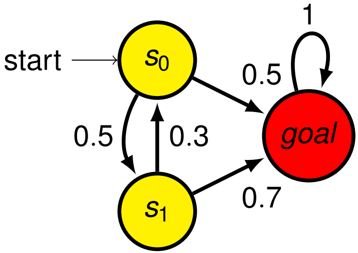
Model-checkers can only work on model of reasonable size.
Principle : reduce the size of models, possibly with a loss of precision
We need to use an abstraction technique such as :

- Three-valued abstraction [5]
- Game-based abstraction [6]

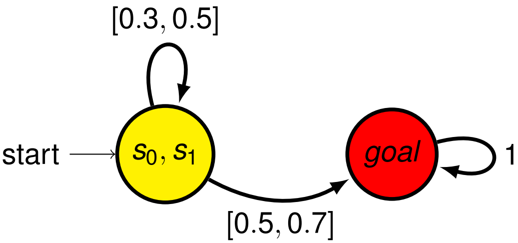
---

5. On Abstraction of Probabilistic Systems, Dehnert et al., 2012
6. Game-based Abstraction for Markov Decision Processes, Kwiatkowska et al., 2006

# The three-valued abstraction




A very simple MDP




An abstraction of the MDP

Does the formula $P_{\leq 0.6}(X\ goal)$ hold in this MDP ?

# Future Work

- Adapt probabilities for a more accurate modeling of the real system
- Trade off discretization // accuracy
- Use an existing abstraction or design a new one
- Synthesize a policy with a model checker

- Consider a more complex topology