01_Introduction of Hybrid System

1. System and Basic Automata

Definition of System

$$x(\sigma) = \phi(\tau, \sigma, x(\tau), u)$$

- τ : initial time
- σ: current time
- u: input function (over [τ, σ])
- ϕ: transition map

Classification

- Continuous-state (speed, position)/Discrete-state (#customers in the queue)
- Continuous-time/Discrete-time
- Time-driven/Event-Driven

Combination of them we call "**hybrid**"

Models for Time-Driven Systems

Continuous-time time-driven systems:

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ y(t) &= g(x(t), u(t)) \end{aligned}$$

Discrete-time (or sampled) time-driven systems:

$$\begin{aligned} x(k+1) &= f(x(k), u(k)) \\ y(k) &= g(x(k), u(k)) \end{aligned}$$

Models For Event-driven Systems

Definition: Automaton

Automaton is defined by triple $\Sigma = (\mathscr{Q}, \mathscr{U}, \phi)$ with

- $\mathscr{Q}$: **finite or countable** set of discrete states
- $\mathscr{U}$: **finite or countable set** of discrete inputs ("input alphabet")
- ϕ: $\mathscr{Q} \times \mathscr{U} \to P(\mathscr{Q})$: partial transition function.
    - where $P(\mathscr{Q})$ is power set of $\mathscr{Q}$ (set of all subsets)

Definition: Finite Automaton

Finite automaton: $\mathscr{Q}$ and $\mathscr{U}$ finite

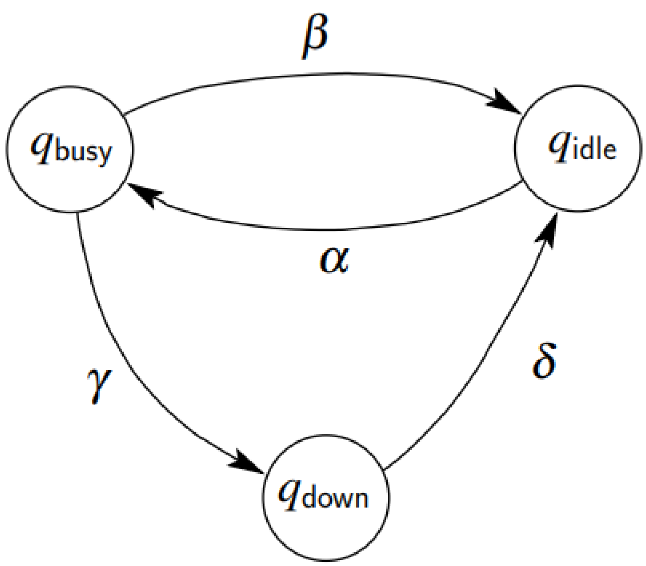
Definition: Countable

Countable means we can assign **order** on the elements

Evolution of Automaton

Deterministic Automaton

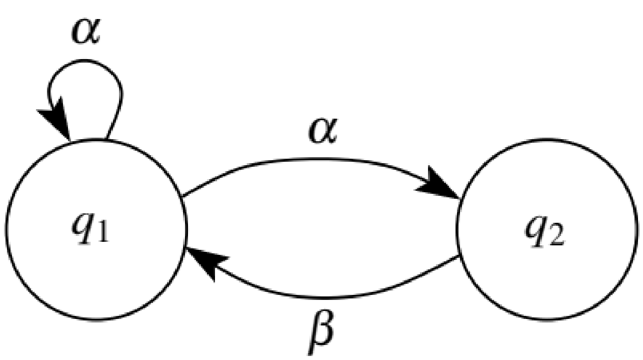
If each set of next states has 0 or 1 element



$$\phi(q_{\text{busy}}, \beta) = \{q_{\text{idle}}\} \qquad \phi(q_{\text{idle}}, \alpha) = \{q_{\text{busy}}\}$$
$$\phi(q_{\text{busy}}, \gamma) = \{q_{\text{down}}\} \qquad \phi(q_{\text{down}}, \delta) = \{q_{\text{idle}}\}$$

Non-Deterministic Automaton

If some set of next states has more than 1 element



$$\phi(q_1, \alpha) = \{q_1, q_2\} \qquad \phi(q_2, \beta) = \{q_1\}$$

Hybrid System

System can be in **one of several modes**

- In each mode: behavior described by system of difference or differential equations
- Mode switches due to occurrence of "**events**"
  - external control signal
  - internal control signal
  - dynamics of system itself

2. Computation Complexity

Undecidable Problems

No algorithm at all can be given for solving the problem in general, i.e., **finite termination cannot be guaranteed**

Decision Problem

Solution is either "yes" or "no". Like "does there exits?"

> e.g., traveling salesman decision problem:
> Given a network of cities, intercity distances, and a number B, does there exist a tour with length $\leqslant B$?

Search Problem

> e.g., traveling salesman problem:
> Given a network of cities, intercity distances, what is the shortest tour?

Time Complexity Function $T(n)$

Largest amount of time needed to solve problem instance of size n **(worst case!)**

Polynomial Time Algorithm

Polynomial Time Algorithm

$T(n) \leqslant |p(n)|$ for some polynomial p

P Problem

solvable by polynomial time algorithm

Nondeterministically Polynomial problems

NP Class

Time complexity of **checking stage** is polynomial

Note:

NP class means we **can check quickly.**

- Nondeterministic computer:
  - guessing stage (tour)
  - checking stage (compute length of tour + compare it with B)

- Each problem in NP can be solved in exponential time: $T(n) \leq 2^{n^k}$

<u>NP-complete Decision Problems:</u>

the “most difficult” class in NP

- any NP-complete problem solvable in polynomial time $\Rightarrow$ every problem in NP solvable in polynomial time
- any problem in NP intractable $\Rightarrow$ NP-complete problems also intractable

NP-hard Problem

<u>NP-hard Problems</u>

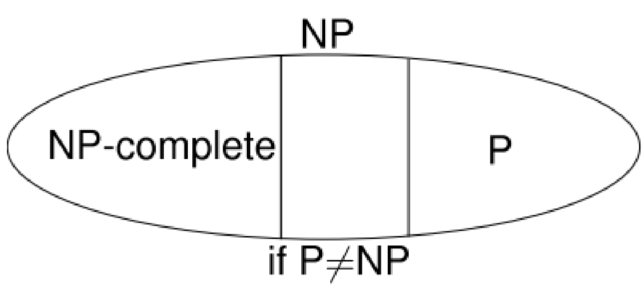
At least as hard as NP-complete problems. Cannot decide whether solution is correct or not in polynomial time.

Notes:

- Decision problem is NP-complete $\Rightarrow$ corresponding search problem is NP-hard
  - For TSDP: guess tour $\rightarrow$ check with B
  - For TSP: generate tour $\rightarrow$ check whether shortest

– NP-complete (decision problem)
  $\rightarrow$ solvable in polynomial time *if and only if* P = NP
– NP-hard (search problem)
  $\rightarrow$ cannot be solved in polynomial time *unless* P = NP

Relations



3. Hybrid Automaton

Definition

Hybrid automaton H is collection $H=(Q, X, f, Init, \operatorname{lnv}, E, G, R)$ where

- $Q=\{q_1, \ldots, q_N\}$ is finite set of **discrete states or modes**
- $X=\mathbb{R}^n$ is set of **continuous states**
- $f: Q \times X \rightarrow X$ is **vector field**
- $Init \subseteq Q \times X$ is set of **initial states**
- $Inv: Q \rightarrow P(X)$ describes **invariants**
- $E \subseteq Q \times Q$ is set of **edges or transitions**
- $G: E \rightarrow P(X)$ is guard condition
- $R: E \rightarrow P(X \times X)$ is **reset map**

illustration

- Hybrid state: (q, x)
- **Evolution** of continuous state in mode $q: \dot{x}=f(q, x)$
- Invariant Inv: describes conditions that **continuous state has to satisfy in given mode**
- Guard G: specifies subset of state space where certain **transition is enabled**
- Reset map R : specifies how **new continuous states are related to previous continuous states**

Evolution of Hybrid Automaton

- **Initial** hybrid state $(q_0, x_0) \in \operatorname{Init}$
- **Continuous state** x evolves according to

$$\dot{x}=f\left(q_0, x\right) \quad \text { with } \quad x(0)=x_0$$

- **discrete state** q remains constant: $q(t)=q_0$
- Continuous evolution **can go on as long as** $x \in \operatorname{lnv}\left(q_0\right)$
- If at some point state x **reaches guard** $G\left(q_0, q_1\right)$, then
  - transition $q_0 \rightarrow q_1$ is **enabled**
  - discrete state **may change** to q_1, continuous state then **jumps** from current value x^{-} to new value x^{+} with $\left(x^{-}, x^{+}\right) \in R\left(q_0, q_1\right)$
- Next, continuous evolution resumes and whole process is repeated

4. Zeno Behavior

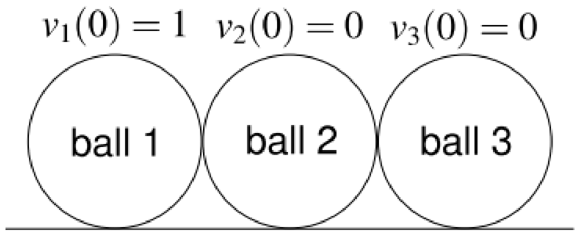
Introduction

infinitely many mode **switches** in **finite** time interval

- **Live-lock** is a case of Zeno Behavior

Bouncing Ball Example

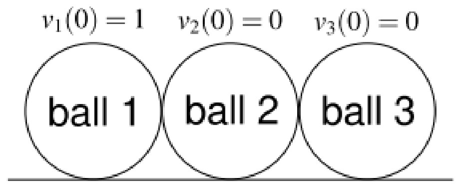
3.4 Three-balls example



- System consisting of three balls
- Inelastic impacts modeled by successions of simple impacts
- Suppose unit masses, touching at time 0, and $v_1(0) = 1$, $v_2(0) = v_3(0) = 0$
- We model all impacts separately $\rightarrow$
  - first, inelastic collision between balls 1 and 2, resulting in $v_1(0+) = v_2(0+) = 0.5$, $v_3(0+) = 0$

hs_intro.50

3.4 Three-balls example (continued)



- next, ball 2 hits ball 3, resulting in $v_1(0{+}{+}) = \frac{1}{2}$, $v_2(0{+}{+}) = v_3(0{+}{+}) = \frac{1}{4}$
- next, ball 1 hits ball 2 again, etc.

$\rightarrow$ sequence of resets:
$$\begin{array}{llllllll} v_1: & 1 & \frac{1}{2} & \frac{1}{2} & \frac{3}{8} & \frac{3}{8} & \frac{11}{32} & \ldots \\ v_2: & 0 & \frac{1}{2} & \frac{1}{4} & \frac{3}{8} & \frac{5}{16} & \frac{11}{32} & \ldots \\ v_3: & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{5}{16} & \frac{5}{16} & \ldots \end{array}$$

converges to $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})^\top$

- Afterwards, smooth continuation is possible with constant and equal velocity for all balls
- Infinite number of events (resets) at one time instant, sometimes called *live-lock* $\rightarrow$ another special case of Zeno behavior

Summary

4. Summary

- Definition and examples of hybrid systems
- Hybrid automaton
- Complexity issues: modeling power vs decision power
- Zeno behavior