Modeling & Control of Hybrid Systems

Chapter 7 — Model Checking and Timed Automata

Overview

1. Introduction

- *Model checking* = process of automatically analyzing properties of systems by exploring their state space
- Finite state systems → properties can be investigated by systematically exploring states
  E.g., check whether particular set of states will be reached
- Not possible for hybrid systems since number of states is infinite
- However, for some hybrid systems one can find "equivalent" finite state system by partitioning state space into finite number of sets such that any two states in set exhibit similar behavior
  → analyze hybrid system by working with sets of partition
- Generation and analysis of finite partition can be carried out by computer

2. Transition systems

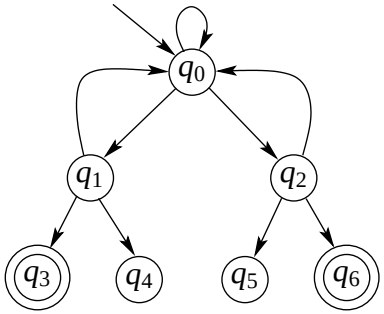
Transition system $T = (S, \delta, S_0, S_F)$ consists of

- set of states S (finite or infinite)
- transition relation $\delta : S \to P(S)$
- set of initial states $S_0 \subseteq S$
- set of final states $S_F \subseteq S$

Trajectory of transition system is (in)finite sequence of states $\{s_i\}_{i=0}^{N}$ such that

- $s_0 \in S_0$
- $s_{i+1} \in \delta(s_i)$ for all i

Example of finite state transition system



- states: $S = \{q_0, \ldots, q_6\}$;
- transition relation: $\delta(q_0) = \{q_0, q_1, q_2\}$, $\delta(q_1) = \{q_0, q_3, q_4\}$, $\delta(q_2) = \{q_0, q_5, q_6\}$, $\delta(q_3) = \delta(q_4) = \delta(q_5) = \delta(q_6) = \varnothing$
- initial states: $S_0 = \{q_0\}$
- final states: $S_F = \{q_3, q_6\}$ (indicated by double circles)

Transition system of hybrid automaton

- Hybrid automaton can be transformed into transition system by abstracting away time
  we do not care how long it takes to get from s to s', we only care whether it is possible to get there eventually

$\rightarrow$ transition system captures sequence of events that hybrid system may experience, but *not* timing of these events

Reachability

- Transition system is *reachable* if there exists trajectory such that $s_i \in S_F$ for some i

3. Bisimulation

- Turn *infinite* state system into *finite* state system by grouping together states that have "similar" behavior $\rightarrow$ partition
- Yields so-called quotient transition system
  finite number of states $\rightarrow$ can be analyzed more easily
- Problem: for most partitions properties of quotient transition system do not allow to draw any useful conclusions about properties of original system
- However, special type of partition for which quotient system $\hat{T}$ is "equivalent" to original transition system T: *bisimulation*

Important property

If partition $\{S_i\}_{i \in I}$ is bisimulation of transition system T and $\hat{T}$ is quotient transition system, then S_F is reachable by T if and only if corresponding final state $\hat{S}_F$ in $\hat{T}$ is reachable by $\hat{T}$

- For finite state systems $\rightarrow$ computational efficiency
  Study reachability in quotient system instead of original system (quotient system usually much smaller than original)
- For infinite state systems:
  Even if original transition system has infinite number of states, sometimes bisimulation consisting of finite number of sets
  $\rightarrow$ answer reachability questions for infinite state system by studying equivalent finite state system

Bisimulation algorithm

- For timed automata we can always find *finite* bisimulation
- Bisimulation algorithm (see lecture notes):
  - For finite state systems bisimulation algorithm will always terminate
    Problem: it may be more work to find bisimulation than to investigate reachability of the original system
  - For infinite state systems: sometimes, algorithm may never terminate (reason: not all infinite state transition systems have finite bisimulations)

Bisimulation algorithm (continued)

- For *timed automata*: bisimulation algorithm terminates in finite number of steps

  Disadvantage: total number of states in the quotient transition system grows very quickly (exponentially) as number of timers n increases