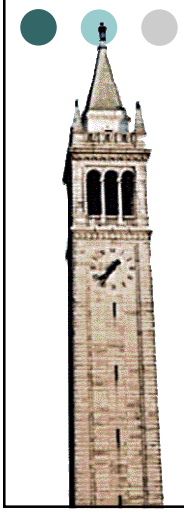
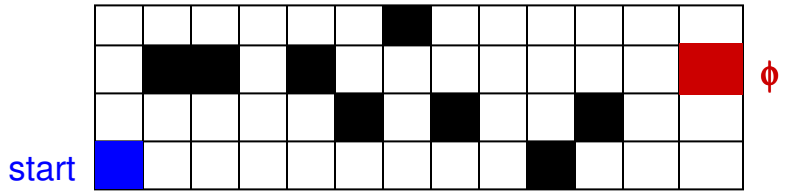


Introduction to Embedded Systems

Edward A. Lee & Sanjit A. Seshia
UC Berkeley
EECS 124
Spring 2008

Lecture 15: Controller Synthesis

Recap: A Robot delivery service, with moving obstacles



ϕ = destination for robot

At any time step:

Robot can move Left, Right, Up, Down, Stay Put

Environment can move one obstacle Up or Down or Stay Put

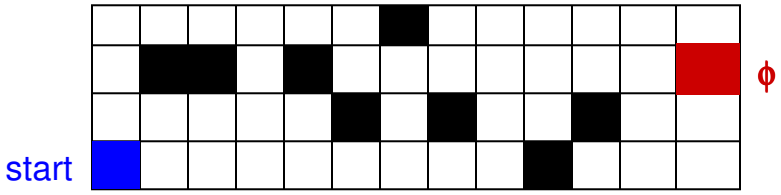
→ But only total of 5 times / 2 times

Can model Robot and Env as FSMs

→ Robot state = its position,

→ Env state = positions of obstacles and counts

Recap of topics covered last time



1. Stated the goal in temporal logic: **F** ϕ
   - The problem is a "reachability problem"
2. Gave an algorithm to solve a version of the reachability problem
3. Considered some alternative goals:

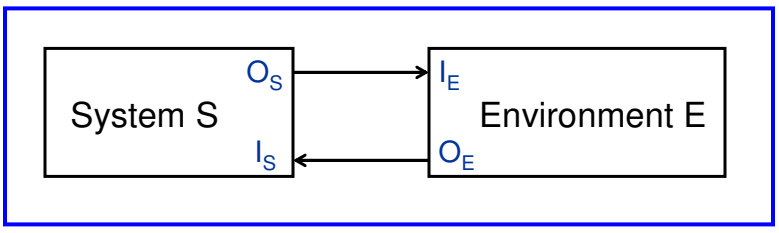
   F $\phi_1 \wedge$ **F** $\phi_2 \wedge \ldots \wedge$ **F** ϕ_n

   F($\phi_1 \wedge$ **F (** $\phi_2 \wedge \ldots \wedge$ **F** ϕ_n **))**

Synthesizing the Robot's Strategy

- Construct FSMs for Robot and Environment
- Compose FSMs to form a new FSM
- Check whether the goal state is reachable from the start state

Under what kind of environment: benign or adversarial?



Reachability Analysis, Revisited

The reachability problem:

Given an FSM M = (Q, δ, Q_0), and a state ϕ,
is *s* reachable from some $q_0 \in Q_0$ by following δ ?

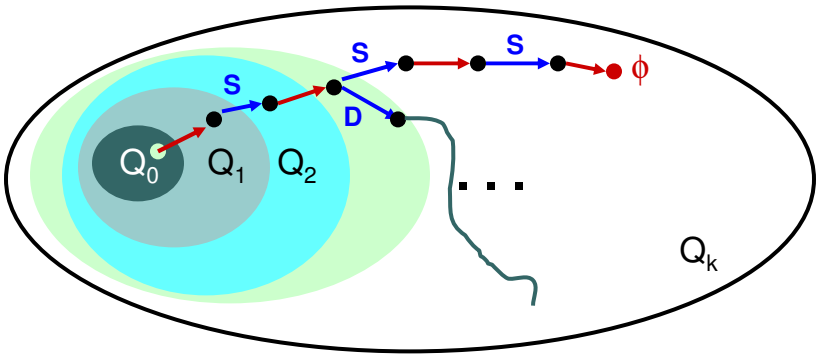
System evolution according to δ is a sequence:
robot step, env step, robot step, env step, …

ϕ is reachable if there is some sequence of robot and env steps from q_0 to ϕ

→ This seq need not have the worst-case env steps!

→ It's optimistic – assumes helpful environment/static obstacles

Visualizing 'Optimistic' Controller Synthesis



S – obstacles "stay put"

D – obstacle "moves down"

Synthesis with an Adversarial Environment

Suppose at every step, an adversary picks the worst possible action to stop the robot's progress

This is a game between the system and its adversarial environment

We want to modify the search performed by the reachability algorithm to handle this worst-case behavior.

Any ideas?

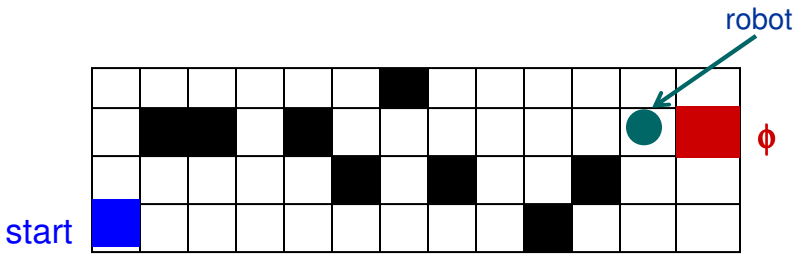
Controllable States

Idea:
Compute the set of states from which, no matter what the environment does, the robot can reach the red square ϕ.

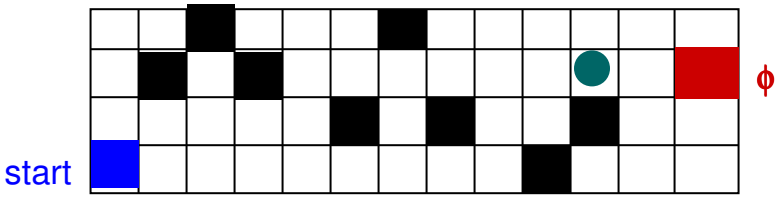
Such states are called controllable states.

What are some examples of controllable states for our robot example?

Examples of Controllable States

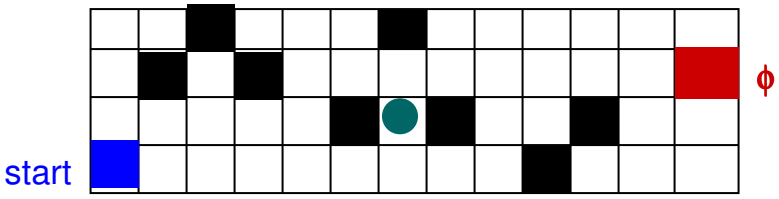


Examples of Controllable States



Env_Moves = MAX_MOVES

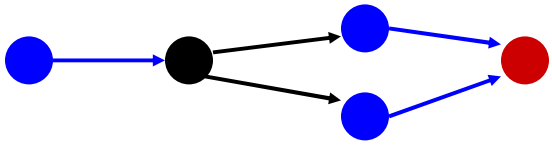
Examples of Controllable States



Env_Moves = MAX_MOVES

Synthesis Algorithm

1. Start with the set of trivially controllable states S_0
2. Add all states from which the robot can reach S_0 in one *FSM step* (robot step, env step), no matter what the environment does



3. Repeat until no new states added
4. Check if this set contains a start state.
   If yes, then we found a strategy.
   If no, then no strategy exists against the worst-case environment (adversary).

Synthesis Algorithm: Formal Description

Input: Description of M: (Q_0, δ), ϕ

Each state $q = (q_1, q_2)$, δ_1 updates q_1, δ_2 updates q_2

Output: Does Q_0 contain a controllable state?

Init: $S := S_{new} := \phi$;

while (S_{new} != $\emptyset$) {

if ($S_{new} \cap Q_0$!= $\emptyset$) return YES;

$S' := \{ q \mid \forall p_1 \in \delta_1(q) \exists p_2 \in \delta_2(p_1) \text{ s.t. } p_2 \in S \}$

$\cup$ S

$S_{new} := S' \setminus S$; S := S';

}

return NO;

S is the set of controllable states

Controller Synthesis for **G** *p*

Suppose we want the system to always satisfy property p

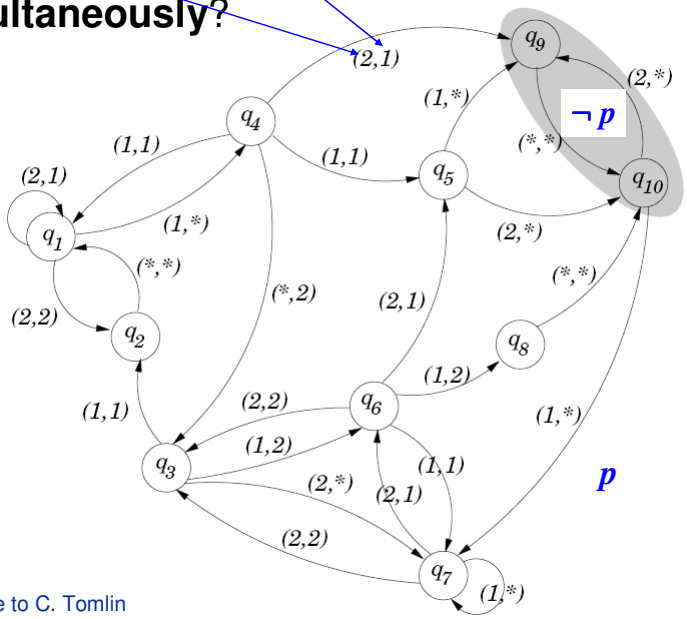
→ alternatively, we want the system to never satisfy ¬ p

E.g. robot should never hit an obstacle; aircraft should never collide; etc.

How can we use the previous algorithm to synthesize a control strategy for **G** p?

[Hint: switch the roles of the environment and the system (robot) – now the environment is trying to reach a goal state]

What if System & Environment Take Steps **Simultaneously**?



Example due to C. Tomlin

Handling other kinds of Temporal Logic Goals

G F p

Example:

The iRobot must visit the charging station infinitely often

- Consider the FSM formed by composing the iRobot FSM with its Environment FSM
- Visualize this FSM as a directed graph
- Suppose that "visiting the charging station" is a state p in this graph

  What graph property corresponds to visiting the state p infinitely often?