A Traffic Sensor that Issues Tickets

November 7, 2005, Cupertino, CA – SpeedInfo Inc., a private company based in Cupertino, Calif., has deployed an additional 50 DVSS-100 lightweight, solar powered radar sensors in the San Francisco Bay metro area. The DVSS-100 Speed Sensor is a fully self-contained, roadside-mounted, traffic measurement sensor. (www.speedinfo.com)

Consider a prototype model with a camera that not only measures traffic data, but also detects speeders and issues speeding tickets.

Performance Measure?

Environment?
Fully observable (partially/not observable):
Deterministic (stochastic):
Episodic (sequential):
Static (dynamic):
Discrete (continuous):
Singe agent (multi agent):

Actuators? – Issue ticket
Sensors? – Radar, camera

Buckets of water

Google interview question (2005): You have a 5-gallon, a 3-gallon bucket, and a faucet. How do you measure exactly 4 gallons of water?

Generally: An agent has a set of buckets B1, …, Bn, all of different integer sizes (in gallons). The largest bucket, Bn, holds cmax gallons. The agent must fill this bucket Bn with exactly g gallons of water in it, where g < cmax, while minimizing wasted water?

Performance Measure?

Environment?
Fully observable (partially/not observable):
Deterministic (stochastic):
Episodic (sequential):
Static (dynamic):
Discrete (continuous):
Singe agent (multi agent):

Actuators?

Sensors?

Formulating "Buckets of Water" as a Search Problem

Problem Definition

An agent has a set of buckets B1, …, Bn, all of different integer sizes (in gallons). The largest bucket, Bn, holds cmax gallons. The agent must fill this bucket Bn with exactly g gallons of water in it, where g < cmax, while minimizing wasted water?

State space: For a bucket B_i, let c_i be its capacity and w_i be the amount of water currently contained within it. Then, we can describe a state by specifying w_i `for all i from 1 to n`.

Actions: $\texttt{Empty}(B_i)$, $\texttt{Fill}(B_i)$, $\texttt{Pour}(B_i, B_j)$

Initial state: $w_i = 0$ for all i

Goal test: $w_n = g$

Successor function: Informally, the successor function captures what actions are allowed in a state and what new state results from performing each action. Formally, a successor function maps from a state to a set of (action, state) pairs.

$$\text{Successor}(w1, \ldots, wn) =$$
$$\{(\text{Empty}(B_i),\ w_i' = 0\ \&\ w_k' = w_k \text{ for } k \neq i) \text{ for all } i\} \cup$$
$$\{(\text{Fill}(B_i),\ w_i' = c_i\ \&\ w_k' = w_k \text{ for } k \neq i) \text{ for all } i\} \cup$$
$$\{(\text{Pour}(B_i, B_j),\ w_i' = \max(0, w_i - (c_j - w_j))\ \&$$
$$w_j' = \min(c_j, w_i + w_j)\ \&\ w_k' = w_k \text{ for } k \neq I, j) \text{ for all } i, j\}$$

Cost function: Informally, we can assign cost only to filling buckets from the tap. Formally, we have:

$$\text{Given state S characterized by } \{w_1, \ldots, w_n\},$$
$$\text{Cost}(\text{Fill}(B_i)) = c_i - w_i$$
$$\text{Cost}(\text{Empty}(B_i)) = 0$$
$$\text{Cost}(\text{Pour}(B_i, B_j)) = 0$$

The path cost is the sum of the costs of all actions in the path.