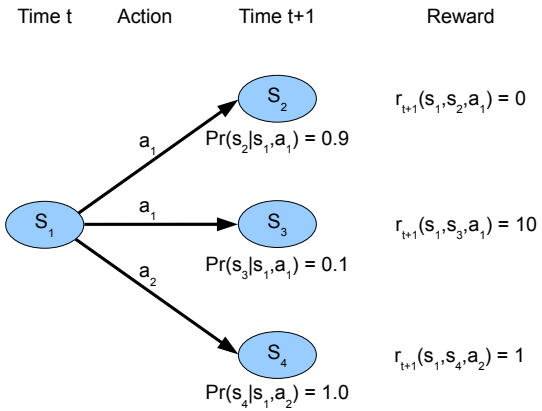


Planning in Continuous Domains with MDPs

Neil Bacon, Supervisor: Scott Sanner

August 11, 2008

Markov Decision Process Example



MDP Formalism

A Markov Decision Process (MDP) is characterised by

$$M = \langle S, A, \mathcal{P}_{s,s'}^a, \mathcal{R}_{s,s'}^a \rangle$$

where

- S is the state space,
- A is the action space,
- $\mathcal{P}_{s,s'}^a = \Pr(s_{t+1} = s' | s_t = s, a_t = a)$ is the stochastic Markov state transition model and
- $\mathcal{R}_{s,s'}^a = \mathbb{E}(r_{t+1} | s_t = s, a_t = a, s_{t+1} = s')$ is the expected value of the next reward.

Planning with MDPs

A policy $\pi(s)$ is a function from states to actions.

An optimal policy $\pi^*(s)$ is a policy that maximises some definition of long term reward, e.g. sum of discounted future rewards.

Many AI planning problems can be formulated as a MDP [3].

The goal of a MDP planning problem is to find the optimal policy $\pi^*(s)$.

Goal

Solve MDP planning problems with continuous state and action spaces and continuous transition models.

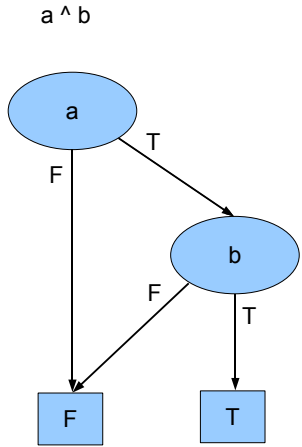
Solve using value iteration on Extended Algebraic Decision Diagrams (ADDs).

Simplifying assumption: state variables conditionally independent given previous state.

Apply to problems in the traffic controller domain and if time permits the Mars rover domain.

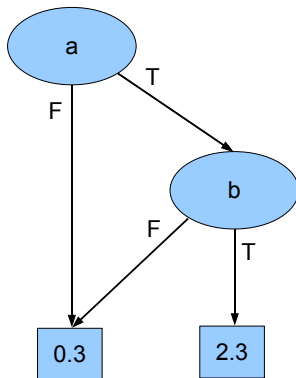
Binary Decision Diagram

Tree representing a function $\mathbb{B}^n \rightarrow \mathbb{B}$



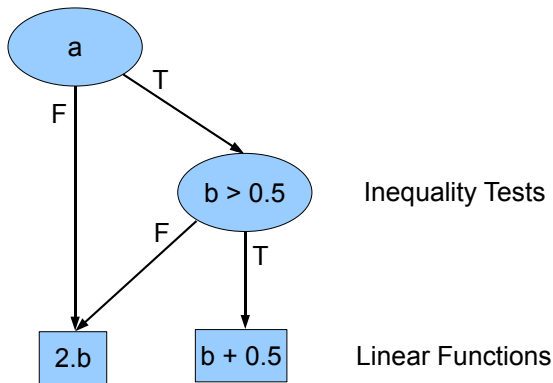
Algebraic Decision Diagram

Tree representing a function $\mathbb{B}^n \rightarrow \mathbb{R}$ [1]



Extended Algebraic Decision Diagram

New work: A compact representation of a function $\langle \mathbb{B}^n, \mathbb{R}^m \rangle \rightarrow \mathbb{R}$, amenable to computation.



Value Iteration with ADDs

Perform value iteration on Extended ADDs to find the optimal value function. At each iteration:

- the next value function and policy are computed exactly from current ADDs;
- computation involves a convolution - result has higher polynomial order [2];
- approximate result with a piece-wise linear function on a hyper-rectangular partition, represented by an Extended ADD.

Traffic Control Domain

Planning in
Continuous
Domains with
MDPs

Neil Bacon,
Supervisor:
Scott Sanner

Planning with
MDPs

Goal

Using ADDs

Traffic Control
Domain

Control Model

Traffic Model

Reward and
Objective

What you can
do

The initial application domain is a traffic controller for a single intersection.

The intersection model will be kept as simple as possible. The state model $\langle s, \vec{q} \rangle$ consists of

- the cycle state $s \in \{s_1, \dots, s_m\}$ determines which directions get a green light and which get a red,
- the number of cars in each of n lane queues $\vec{q} = \langle q_1, \dots, q_n \rangle$ where each $q_i \in [0, \infty]$ (assume fully observable).

Traffic Control Domain - Extensions

Planning in
Continuous
Domains with
MDPs

Neil Bacon,
Supervisor:
Scott Sanner

Planning with
MDPs

Goal

Using ADDs

Traffic Control
Domain

Control Model

Traffic Model

Reward and
Objective

What you can
do

Time permitting, the model can be extended to include:

- multiple intersections (including simple models of traffic as they travel between intersections),
- pedestrian push buttons,
- skipping a cycle if the queue is short in order to clear a longer queue on another road (as long as fairness is ensured).

Control Model

We assume the only control action at any stage in the cycle is how long to remain in that cycle stage. Thus, we model the action as a continuous waiting time t on the interval $t \in [MinCycleTime, MaxCycleTime]$.

Traffic Arrival Model

The traffic model updates the queue lengths at each stage as the difference between an *arrival model* and a *clearance model*.

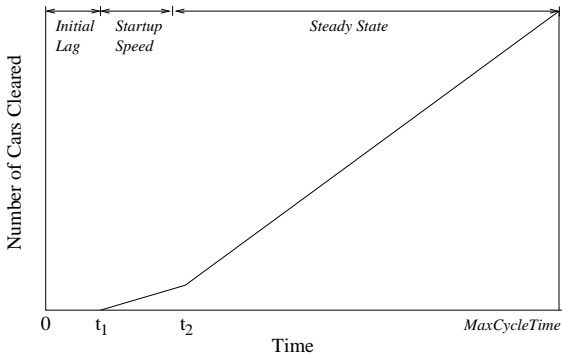
The proposed arrival model is Poisson, with the probability of k cars arriving in the interval t given by:

$$\Pr(k|t) = \frac{(\lambda \frac{t}{I})^k e^{-\lambda \frac{t}{I}}}{k!}$$

where λ is the mean number of arrivals in the interval I .

Traffic Clearance Model

The proposed clearance model for the queue whose light is green is represented as the following piecewise linear function of the action duration:



Reward and Objective

The immediate reward is negative sum of the queue lengths:

$$r(s, \vec{q}) = \sum_i -q_i$$

and the discounted long term reward that we seek to maximise, is defined at time step t as:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

where $0 < \gamma < 1$ is the discount rate.

What you can do

Wish me luck!

Best wishes for your own projects.



Jesse Hoey, Robert St-aubin, Alan Hu, and Craig Boutilier.
SPUDD: Stochastic Planning Using Decision Diagrams.
*In Proceedings of the Fifteenth Conference on Uncertainty
in Artificial Intelligence*, pages 279–288. Morgan
Kaufmann, 1999.



Lihong Li and Michael L. Littman.
Lazy approximation for solving continuous finite-horizon
MDPs.
In National Conference on Artificial Intelligence, 2005.



Richard S. Sutton and Andrew G. Barto.
Reinforcement Learning: An Introduction.
MIT Press, 1998.