Takehome Final

Michael Anderson

March 18, 2011

CS533

Prof. Fern

1

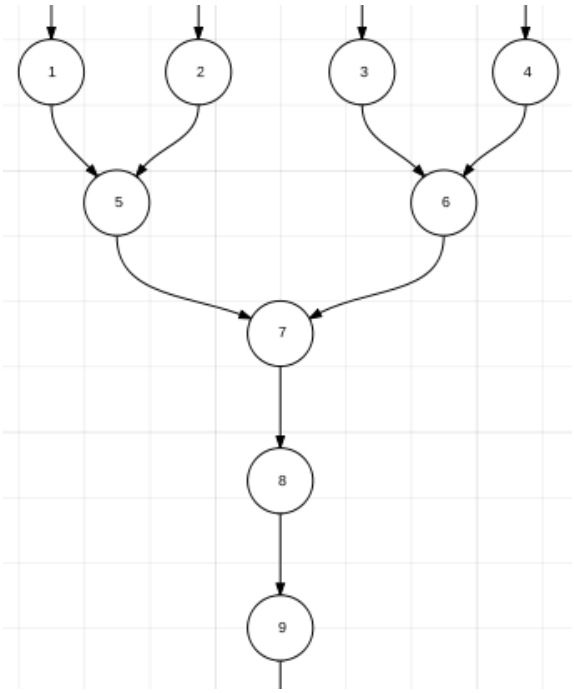
Imagine a k-order MDP in which all states have at most one parent state. In other words, for all s' that are states in the MDP, there exists at most one s such that $T(s, a, s') > 0$. Such a a k-order MDP would be directly convertible to a first-order MDP, because each state would only have one possible vector of ancestor states.

However, in an average k-order MDP where states can have multiple parents, a state could have a number of different ancestor vectors, and that information is not really accounted for in a normal first-order MDP.

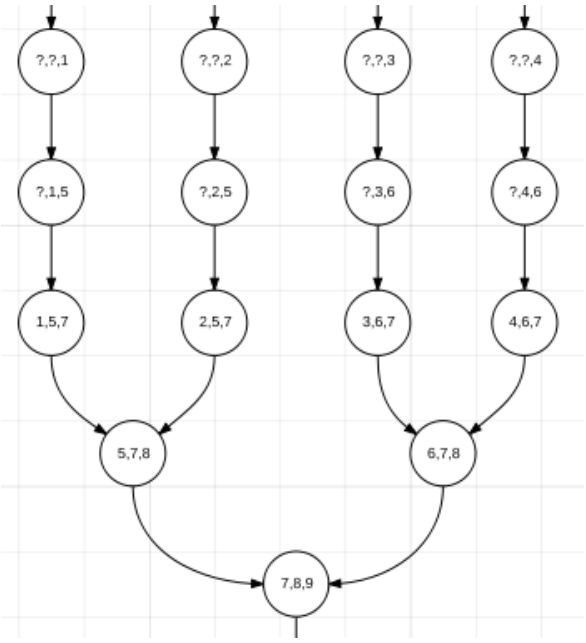
So, to convert a k-order MDP M to a first-order MDP M', traverse M cloning states, actions, rewards, and transitions over to M' exactly, except whenever a state s is found with multiple parents, make a set of states in M', $s_1, s_2, ...$, one state for each of the parents to transition to. Also duplicate all of the descendants of s, so that $s_1, s_2, ...$ each have their own copy. Finally, when copies of descendants are made in this way, check to see if they can be merged back together farther down the tree when they have the same ancestor vectors again (see picture).

The handling of A', R', and T' are straightforward. For states without multiple parents, they are identical to A, R, and T. But supposing that for some state s' in S there exists multiple s such that s is in S and $T(s, a, s') > 0$, then for each such s with a corresponding a that reaches s', make a a member of A' and make $T'(s, a, s_i) = T(s, a, s')$. For the rewards, make $R'(s_i) = R(s')$.

Below is an example of part of a 3-order MDP M and the corresponding part of its equivalent first-order MDP M'. This MDP is deterministic for simplicity, but without loss of generality. States are labeled 1 through 9 in M, and in M' states are labeled by their grandfather, their father, and then the state themselves. Notice that since 5 and 6 have two parents, they are each expanded into two states in M'. 7 is expanded into four states, because it has four combinations of grandfather/father pairs. 8 only has two such combinations and 9 only has one, so we see in the picture an example of the merging of descendant states mentioned above.



M



M'

2

Let $g(\theta_1, ..., \theta_n) = e^{Q_\theta(s,a)}$ and $h(\theta_1, ..., \theta_n) = \sum_{a'} e^{Q_\theta(s,a')}$. Then $\pi_\theta(s,a) = g/h$ and we want to compute:

$$\frac{\partial log(\frac{g}{h})}{\partial \theta_i}$$

Using $'$ as shorthand for the partial derivative in question, and applying the chain rule and quotient rule gives:

$$\frac{\partial log(\frac{g}{h})}{\partial \theta_i} = \frac{h}{g} \times \frac{\partial \frac{g}{h}}{\partial \theta_i} = \frac{h}{g} \times \frac{g'h - gh'}{h^2} = \frac{g'}{g} - \frac{h'}{h}$$

Need to apply the chain rule to differentiate g, and the hard part is differentiating the $Q_\theta(s,a)$ portion. Since all of the parts of the summation that defines Q_π are constant or held constant excepting θ_i, $Q'_\pi(s,a)$ is simply $f_i(s,a)$. So we get:

$$g' = \frac{\partial e^{Q_\theta(s,a)}}{\partial \theta_i} = f_i(s,a) e^{Q_\theta(s,a)}$$

Differentiating h is similar:

$$h' = \frac{\partial \sum_{a'} e^{Q_\theta(s,a')}}{\partial \theta_i} = \sum_{a'} \frac{\partial e^{Q_\theta(s,a')}}{\partial \theta_i} = \sum_{a'} f_i(s,a') e^{Q_\theta(s,a')}$$

Now we are ready to finish:

$$\frac{\partial log(\pi_\theta(s,a))}{\partial \theta_i} = \frac{g'}{g} - \frac{h'}{h} =$$

$$\frac{f_i(s,a) e^{Q_\theta(s,a)}}{e^{Q_\theta(s,a)}} - \frac{\sum_{a'} f_i(s,a') e^{Q_\theta(s,a')}}{\sum_{a'} e^{Q_\theta(s,a')}} =$$

$$\frac{f_i(s,a) e^{Q_\theta(s,a)}}{e^{Q_\theta(s,a)}} - \sum_{a'} \frac{e^{Q_\theta(s,a')}}{\sum_{a'} e^{Q_\theta(s,a')}} f_i(s,a') =$$

$$f_i(s,a) - \sum_{a'} \pi_\theta(s,a') f_i(s,a')$$

3

Assume the worst case, which is that the maximum reward R_{max} is obtained at each level of the tree. Then at search depth h, the reward is $\beta^h R_{max}$, because R_{max} is discounted by a factor of β at each level. The infinite horizon reward becomes an easily simplifiable geometric series:

$$Q_\pi(s,a) = \sum_{i=1}^{\infty} \beta^i R_{max} = \frac{1}{1-\beta} R_{max}$$

The finite horizon reward then, for a horizon of h, is

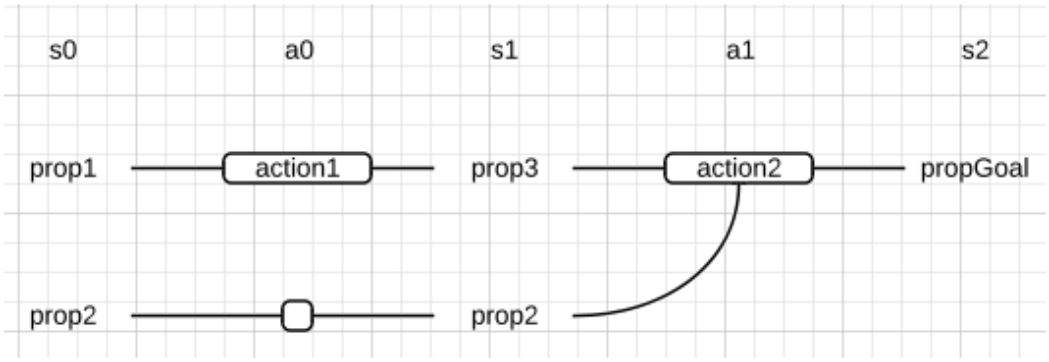
$$Q_\pi(s,a,h) = \sum_{i=1}^{h} \beta^i R_{max} = \frac{1-\beta^h}{1-\beta} R_{max}$$

Since this is the worst case, the difference between $Q_\pi(s,a)$ and $Q_\pi(s,a,h)$ is maximized by:

$$\frac{1}{1-\beta} R_{max} - \frac{1-\beta^h}{1-\beta} R_{max} = \frac{\beta^h}{1-\beta} R_{max}$$

4

(a) *No*, it is definitely not complete. If a proposition in state s_i that is not deleted by a_i does not persist to s_{i+1}, then the algorithm ends up with a restricted view of the environment, and may lose information that it needs to find a route to the goal. Consider the following planning graph:



In s0 prop1 and prop2 are asserted, in a0 action1 can be taken to add prop3 and delete its precondition prop1, then in s1 action2 (with preconditions prop3 and prop2) can be taken to reach propGoal, which we will assume is the only proposition of the goal state.

In this scenario, it is only possible to reach the goal if prop2 persists from s0 to s1. If that persistence action was not there, then GraphPlan would fail to find a path to the goal state, and is therefore incomplete.

(b) *No*, it is definitely not sound. The purpose of computing mutex relations is not only to prune the search tree, making GraphPlan more efficient, but also to ensure that it produces valid layered plans. For example, suppose at some level we have the necessary preconditions to take either of two actions, but one action deletes a precondition of the other action. Using mutexes will rule out these two actions as not valid in the same layer, but without using mutexes GraphPlan could place them in the same layer. Since one deletes a precondition of the other, the order that the actions were taken in would be significant, and a plan putting them in the same layer would be unsound.

5