1 Curse of Dimensionality

1.1 State space explosion!

1.1.1 Example 1: Traveling salesman problem

See Winstons book [1].

1.1.2 Example 2: Fishery:

A town must decide how many bass to catch and sell in a year. For selling x bass in year t, the revenue generated is $r_t(x)$. The cost of catching x bass is $c(x, b)$ where b is the number of bass in the lake at the beginning of the year. Bass reproduce and the number of bass in the lake at the beginning of a year is 20% more that the number left at the end of the previous year. Assume that there are 10000 bass in the lake at the beginning of year 1. Formulate a DP recursion to maximize revenue over T years. [1]

2 Mitigating Curse of Dimensionality

1) State space aggregation or gridding , 2) interpolation and 3) Value Function approximation (OR 774)

2.1 State space aggregation or gridding

This also called coarse discretization of the state space. In the fisheries problem impose a condition that you may catch bass only in multiples of 1000. In example 3 above, cash can be spent only in multiples of 100 and bushels can be sold or bought in multiples of 100. Coarse gridding reduces the resolution which in turn reduces the problem's solution quality. To improve the solution quality, solve the problem first with coarse gridding. Find the states that are optimal. Then, finely discretize states that are around the optimal states (a process known a grid refinement). Solve the problem again. Continue until you are satisfied with the resolution of the answer or until your computer chokes, which ever happens first! Its a hope that refinement will increase the solution quality but sometimes the optimal path may lie outside the refined area.

In summary, disadvantage of aggregation: Two different states S_1 and S_2 may have the same value function. It may happen then, that we make the same decision in both states eventhough the difference in state S_1 and S_2 may call for different decisions.

2.2 Interpolation after gridding

In problems where price of an asset or return on investments is a state variable then there may not be an exact match between the next state to the set of descritized states on the grid. In this case one can round of the next state to one of the nearest state (grid point) or add the state as a new state (new grid point) and interpolate its value function f using the nearest grid points.

2.3 Value Function Approximation (in OR 774)

In general, the regression VFA is written as

$$\bar{V}^{n+1}(S^{n+1}|\theta) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(S^{n+1}), \tag{1}$$

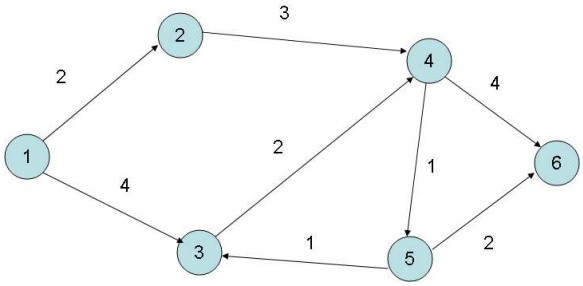

Figure 1: Cyclic network

where $\bar{V}^{n+1}(S^{n+1}|\theta)$ is the estimated value function in state S^{n+1}, θ_f is a parameter vector, ϕ_f is a set of basis functions, and $f \in \mathcal{F}$ is called feature, which is an integer that denotes the number of terms in the regression model. In ADP, the above regression value function approximation can reduce large state-spaces and their values into a small number of features f, which avoids the storage of millions of individual states and their values.

3 Cycles in network

Shortest path Algorithm for cyclic networks:

In acyclic networks, backward recursion is

$$f_t(i) = \min_j [c_{ij} + f_{t+1}(j)],\ i < j \tag{2}$$

or forward reaching is

$$f_{t+1}(j) = \min_i [c_{ij} + f_t(i)],\ i < j. \tag{3}$$

In both equations, $i < j$ is an important requirement. The value of f is calculated in a descending (backward recursion) or an ascending order (forward reaching). However, in network with cycles this is not possible.

For network with cycles only reaching works in which $f_t(j)$ is computed in a non-decreasing f-value sequence.

Example 4: See figure 1

Step 1: set f_1=0. Set $f_j = c_{1j}, \forall j = 2, \cdots, N$ and set $T = \{2, \cdots, N\}$. If node 1 is not connected to any j then set c_{1j} to a large number.

$$f_1 = 0 \tag{4}$$
$$f_2 = c_{12} = 2 \tag{5}$$
$$f_3 = c_{13} = 4 \tag{6}$$
$$f_4 = f_5 = f_6 = \infty \tag{7}$$

Set $T = \{2, \cdots, 6\}$.

Step 2: Find a node i in T for which

$$f_i = \min[f_j | j \in T] \tag{8}$$

So the minimum of $f_2, \cdots, f_6$ is $f_2 = 2$. So $i = 2$.

Step 3: Delete $i = 2$ from T. Stop if T is empty, else go to Step 4.
Step 4: $T = \{3, \cdots, 6\}$. For each $j \in T$, update

$$f_j = \min[f_j, f_i + c_{ij}] \tag{9}$$

$$\begin{aligned} f_3 &= \min[f_3, f_2 + c_{23}] = \min[4, \infty] = 4 && (10)\\ f_4 &= \min[f_4, f_2 + c_{24}] = \min[\infty, 5] = 5 && (11)\\ f_5 &= \min[f_5, f_2 + c_{25}] = \min[\infty, 2 + \infty] = \infty && (12)\\ f_6 &= \min[f_6, f_2 + c_{26}] = \min[\infty, 2 + \infty] = \infty && (13) \end{aligned}$$

Step 5: go to step 2.
Solving the problem further: The minimum of $f_3, \cdots, f_6$ is $f_3 = 4$. So $i = 3$. Delete $i = 3$ from T. $T = \{4, \cdots, 6\}$.

$$\begin{aligned} f_4 &= \min[f_4, f_3 + c_{34}] = \min[5, 6] = 5 && (14)\\ f_5 &= \min[f_5, f_3 + c_{35}] = \min[\infty, 4 + \infty] = \infty && (15)\\ f_6 &= \min[f_6, f_3 + c_{36}] = \min[\infty, 4 + \infty] = \infty && (16) \end{aligned}$$

The minimum of $f_4, \cdots, f_6$ is $f_4 = 5$. So $i = 4$. Delete $i = 4$ from T. $T = \{5, \cdots, 6\}$.

$$\begin{aligned} f_5 &= \min[f_5, f_4 + c_{45}] = \min[\infty, 5 + 1] = 6 && (17)\\ f_6 &= \min[f_6, f_4 + c_{46}] = \min[\infty, 5 + 4] = 9 && (18) \end{aligned}$$

The minimum of $f_5, \cdots, f_6$ is $f_5 = 6$. So $i = 5$. Delete $i = 5$ from T. $T = \{6\}$.

$$f_6 = \min[f_6, f_5 + c_{56}] = \min[9, 6 + 2] = 8 \tag{19}$$

The minimum of f_6 is $f_6 = 8$. So $i = 6$. Delete $i = 6$ from T. $T = \{\phi\}$.
Answer is 8. Tracing the path backwards gives 6-5-4-2-1

4 Examples in Deterministic DP- Infinite Horizon

No longer a shortest path algorithm.

4.1 Machine replacement problem

How long to wait until the machine is replaced . These systems are called regenerative systems because the system regenerates as a new system. Example: When would you replace your car? Renewal systems will renew after a maintenance but do not restart as fresh.

Suppose there are $k = 1, 2, \cdots, N$ alternatives at which the machine can be replaced. This means at t if you pick a value for k then replace after $t+k$ time units. Let R_k be the cost of alternative k at the start of the regeneration period (new purchase+maintenance-salvage). For a given k, R_k is a constant. If β is the time value of money $0 < \beta < 1$ and f^n is the present value of regeneration at iteration n (we drop the use of t because t goes to infinity and we are still using backward recursion, hence, we start using n) then

$$f^{n+1} = \min_k [\beta^k f^n + R_k] \tag{20}$$

The above is same as

$$f_t = \min_k [\beta^k f_{t+k} + R_k] \tag{21}$$

In backward recursion,

$$n = 0, time = t + k(at\ infinity) \tag{22}$$
$$n = 1, time = t \tag{23}$$
$$n = 2, time = t - k \tag{24}$$
$$n = 3, time = t - 2k \tag{25}$$
$$\tag{26}$$

Over an unbounded horizon using the same policy k, the present value of regeneration can be analytically solved as

$$f^{n+1} = \min_k [\frac{R_k}{1 - \beta^k}] \tag{27}$$

because of the geometric progression

$$R_k + \beta^k R_k + \beta^{2k} + \cdots = \frac{R_k}{1 - \beta^k} \tag{28}$$

Since analytical methods are not always available, value iteration is used to find the answer.

4.2 Steps in value iteration

1. Choose f^0 to be any number.
2. Find f^1, f^2, $\cdots$
3. Terminate when $|f^{n+1} - f^n| < \epsilon$, where ϵ is a very small number (such as 0.01).

Example 1:
Given $R_1 = \$260$, $R_2 = \$540$, and $R_3 = \$760$ and $\beta = 0.9$ Find the best replacement policy analytically and using value iteration.
Analytical solution:

References

[1] W. L. Winston. *Introduction to Mathematical Programming, Vol 1.* Thompson, 2003.