Assignment 1

Siddharth Nayak EE16B073

21st August 2018

1 Question 1:

1.1 Brute force method:

For each state we have m actions to be taken. Thus for each of the n states we have m^n computations required. Now for N stages it will be $N.m^n$. Thus the brute force method is of $\mathcal{O}(N.m^n)$

1.2 Dynamic Programming

For dynamic programming we don't have to evaluate the computations which have been done already before all over again as it stores the computations. Thus we will require lesser number of computations. And thus dynamic programming algorithm is preferred. We have an nxn matrix for evaluating the transitions with action. So we have a tensor of nxnxm. For each stage we just have to calculate $m.n^2$ i.e for each edge between the nodes in the graph. Thus for N stages we require n^2mN computations. Thus the complexity of the the DP algorithm is $\mathcal{O}(n^2m.N)$.

1.3 Conclusion:

Clearly Dynamic programming algorithm is preferred over brute force as in brute force the complexity is an exponential one.

2 Question 2:

2.1 Part a:

For any $\Pi = \{\mu_0, \mu_1, ..., \mu_{N-1}\}$
Let $\Pi^k = \{\mu_k, \mu_{k+1}, ..., \mu_{N-1}\}$
Let J_k^* be the optimal cost to-go for the tail sub-problem with state x_k and stage k.
We need to show that $J_k^*(x_k) = J_k(x_k)$ for $k = 0, 1, ...N-1$
For k=N, $J_N^*(x_N) = g_N(x_N) = J_N(x_N) \forall x_N \in X$.
Assume: $J_{k+1}^*(x_{k+1}) = J_{k+1}(x_{k+1}) \forall x_k \in X$
$\Pi^k = \Big\{\mu_k, \Pi^{k+1}\Big\}$

$$J_k^*(x_k) = \min_{\{\mu_k, \Pi^{k+1}\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, \mu_k(x_i), x_{i+1}))\Big\}$$

$$J_k^*(x_k) = \min_{\{\mu_k, \Pi^{k+1}\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_N(x_N) + g_k(x_k, \mu_k(x_k), x_{k+1}) + \sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_k), x_{i+1}))\Big\}$$

$$J_k^*(x_k) = \min_{\{\mu_k, \Pi^{k+1}\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_k(x_k, \mu_k(x_k), x_{k+1})).exp(g_N(x_N)).exp(\sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_k), x_{i+1}))\Big\}$$

$$J_k^*(x_k) = \min_{\{\mu_k\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_k(x_k, \mu_k(x_k), x_{k+1})). \min_{\{\Pi^{k+1}\}} \big[exp(g_N(x_N)).exp(\sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_k), x_{i+1}))\big]\Big\}$$

$$J_k^*(x_k) = \min_{\{\mu_k\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_k(x_k, \mu_k(x_k), x_{k+1})). \min_{\{\Pi^{k+1}\}} \big[exp(g_N(x_N) + \sum_{i=k+1}^{N-1} g_i(x_i, \mu_i(x_k), x_{i+1}))\big]\Big\}$$

$$J_k^*(x_k) = \min_{\{\mu_k\}} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_k(x_k, \mu_k(x_k), x_{k+1})).J_{k+1}^*(x_{k+1})\Big\}$$

$$J_k^*(x_k) = \min_{a_k \in A(x_k)} \mathop{\mathbb{E}}_{x_{k+1}} \Big\{exp(g_k(x_k, \mu_k(x_k), x_{k+1})).J_{k+1}^*(x_{k+1})\Big\}$$

$\therefore J_k^*(x_k) = J_k(x_k)$

Thus the optimal cost and policy can be obtained.

2.2 Part b:

g_k depends only on x_k and a_k and does not depend on x_{k+1}:
$J_N(x_N) = exp(g_N(x_N))....(1)$ and,
$J_k(x_k) = \min_{a_k \in A(x_k)} \mathop{\mathbb{E}}_{x_{k+1}} \big(exp(g_k(x_k, a_k)J_{k+1}(x_{k+1}))\big)....(2)$
$V_k(x_k) = log(J_k(x_k))$
$\therefore$ taking log on both sides in equation(1), we get,

$log(J_N(x_N)) = g_N(x_N) = V_N(x_N)$.
Now taking log on both sides in equation(2), we get,
$log(J_k(x_k)) = v_k(x_k) = log\Big(\min_{a_k \in A(x_k)} \mathop{\mathbb{E}}_{x_{k+1}} \big(exp(g_k(x_k, a_k)J_{k+1}(x_{k+1}))\big)\Big)$
taking log inside min ,
$v_k(x_k) = \min_{a_k \in A(x_k)} log\Big(\mathop{\mathbb{E}}_{x_{k+1}} \big(exp(g_k(x_k, a_k)J_{k+1}(x_{k+1}))\big)\Big)$
Taking $exp(g_k(x_k, a_k))$ out of the expectation as it does not depend on x_{k+1}
$v_k(x_k) = \min_{a_k \in A(x_k)} log\Big(exp(g_k(x_k, a_k). \mathop{\mathbb{E}}_{x_{k+1}} \big(J_{k+1}(x_{k+1})\big)\big)\Big)$
$v_k(x_k) = \min_{a_k \in A(x_k)} \Big(g_k(x_k, a_k) + log\big(\mathop{\mathbb{E}}_{x_{k+1}} \big(J_{k+1}(x_{k+1})\big)\big)\Big)$
Also, $J_{k+1}(x_{k+1}) = exp(V_{k+1}(x_{k+1}))$
$\therefore v_k(x_k) = \min_{a_k \in A(x_k)} \Big[g_k(x_k, a_k) + log\big(\mathop{\mathbb{E}}_{x_{k+1}} \big(exp(V_{k+1}(x_{k+1}))\big)\big)\Big]$

3 Question 3:

3.1 Part a:

This problem is similar to the asset selling problem.
Stages $\rightarrow$ {1,2,3,.....k,k+1,....N}

$$\text{Actions} \rightarrow \begin{cases} a_1 : \text{Buy food} \\ a_2 : \text{Wait for better food} \end{cases}$$

$$\text{States} \rightarrow \begin{cases} T : \text{ if } x_k = T \text{ or if } x \neq T \text{ and } a_k = a_1 \\ \overline{T} : \text{otherwise} \end{cases}$$

In the end the person has to buy something.

$$\text{Cost} \rightarrow g_N(x_N) = \begin{cases} \frac{1}{1-p} : \text{ if } x_N \neq T \\ 0 : \text{ otherwise} \end{cases}$$

$$\text{Cost} \rightarrow g_k(x_k, a_k, x_{k+1}) = \begin{cases} (N-k) : \text{ if } x_k \neq T \text{ and } a_k = a_1 \\ 0 : \text{ otherwise} \end{cases}$$

$$J_N(\overline{T}) = C = \frac{1}{1-p}$$

3.2 Part b:

Writing the DP algorithm for the problem:
$J_k(x_k) = \min_{a_k \in \{a_1, a_2\}} \mathop{\mathbb{E}}_{x_{k+1}} \big((g_k(x_k, a_k, x_{k+1}) + J_{k+1}(x_{k+1})\big)\big)$

$$J_k(x_k) = \begin{cases} \min \Big(p.(N-k) + (1-p).J_{k+1}(x_{k+1}), J_{k+1}(x_{k+1})\Big) : \text{if } x_k \neq T \\ 0 : \text{otherwise} \end{cases}$$

Let $q = 1 - p$
$\therefore J_k(\overline{T}) = p \min[(N-k), J_{k+1}(\overline{T})] + qJ_{k+1}(\overline{T})$
Let $F_k = J_k(\overline{T})$
$\therefore$ the person should buy food if $N - k < F_{k+1}$
We have $F_k \leq F_{k+1}$ due to the min function
We also have:
$F_{k-1} = p \min[(N-k+1), J_k(\overline{T})] + qJ_k(\overline{T})$
Assume that buying at k-1 th shop is optimal.
$\therefore N - k + 1 < F_k$
$N - k < N - k + 1 < F_k \leq F_{k+1}$
$\therefore$ it is also optimal to buy at k th shop if the food is available and if it is optimal to buy at (k-1)th shop.
Now assume it is optimal to not buy food in (k+1)th shop.
$N - k + 1 > F_k$
$\therefore N - k + 1 > N - k > F_{k+1} > F_k$
$\therefore$ if it is optimal to not buy in (k+1)th shop then it is also optimal to not buy in kth shop.
$\therefore$ there exists some k^* where it is optimal to buy
$\therefore$ In particular k^* is the smallest integer which satisfies $N - k > F_{k+1}$
Since $F_N = \frac{1}{1-p} > F_k$ we know that $k^* < \infty$ exists.

Claim: $F_k = (N-k) + q^{N-k}.C - \dfrac{q}{p}(1 - q^{N-k})$ whenever $N - k < F_k$

Proof:
The proof follows by induction. for base case k=N-1:
$F_{N-1} = p. \min((N-(N-1)), F_{k+1}) + q.F_{k+1}$
$F_{N-1} = p. \min(1, C) + q.C$
$\therefore F_{N-1} = p + q.C$
The claim gives us: $F_{N-1} = (N-N+1) + q^{N-N+1}.C - \dfrac{q}{p}(1 - q^{N-N+1})$

$F_{N-1} = 1 + q.C - \dfrac{q}{p}(1-q)$
$F_{N-1} = q.C - \dfrac{p-q+q^2}{p}$
$\therefore F_{N-1} = p + q.C$

Assume the claim hold for $N-k+1 < F_k$. Then
$F_{k-1} = p.\min(N-k+1, F_k) + q.F_k$
$F_{k-1} = p.(N-k+1) + q.F_k$
Substituting F_k
$F_{k-1} = p.(N-k+1) + q.((N-k) + q^{N-k}.C - \dfrac{q}{p}(1-q^{N-k}))$
$F_{k-1} = (N-k+1) - q + q^{N-k+1}.C - \dfrac{q}{p}(q(1-q^{N-k})))$
$F_{k-1} = (N-k+1) + q^{N-k+1}.C - \dfrac{q}{p}(q(1-q^{N-k}) + p)$
$\therefore F_{k-1} = (N-k+1) + q^{N-k+1}.C - \dfrac{q}{p}(1-q^{N-k+1})$

Now since $N-k^*+1 < F_k$ we can determine k^* as the smallest integer satisfying:
$N-k \geq F_{k+1} = (N-k-1) + q^{N-k-1}.C - \dfrac{q}{p}(1-q^{N-k-1})$
$p \geq p.q^{N-k-1}.C - q.(1-q^{N-k-1})$
$1 \geq (p.C+q).q^{N-k-1}$
$\therefore (p.C+q)^{-1} \geq q^{N-k-1}$.

4 Question 4:

There are N jobs to be scheduled:
Let i and j be the kth and (k+1)st job in an optimally ordered list.
$L = \{i_0, i_1, i_2, ...i_{k-1}, i, j, i_{k+2}, ..., i_{N-1}\}$
$L^1 = \{i_0, i_1, i_2, ...i_{k-1}, j, i, i_{k+2}, ..., i_{N-1}\}$
$\mathbb{E}\{\text{reward of } L\} = \mathbb{E}\left\{\text{reward of}\{i_0, ...i_{k-1}\}\right\}$
$\quad + p_{i0}.pi_1....p_{i_k-1}(p_i\beta_i.Z_i + p_i.p_j.\beta_i.Z_i.\beta_j.Z_j)$
$\quad + p_{i_0}.p_{i_1}.....p_ik - 1.p_i.p_j.\mathbb{E}\left\{\text{reward of}\{i_{k+2}, ...i_{N-1}\}\right\}$

$\mathbb{E}\{\text{reward of } L^1\} = \mathbb{E}\left\{\text{reward of}\{i_0, ...i_{k-1}\}\right\}$
$\quad + p_{i0}.pi_1....p_{i_k-1}(p_j\beta_j.Z_j + p_j.p_i.\beta_j.Z_j.\beta_i.Z_i)$
$\quad + p_{i_0}.p_{i_1}.....p_ik - 1.p_i.p_j.\mathbb{E}\left\{\text{reward of}\{i_{k+2}, ...i_{N-1}\}\right\}$
For $\mathbb{E}\{\text{reward of } L\} > \mathbb{E}\{\text{reward of } L^1\}$
We need:
$p_i\beta_i.Z_i + p_i.p_j.\beta_i.Z_i.\beta_j.Z_j > p_j\beta_j.Z_j + p_j.p_i.\beta_j.Z_j.\beta_i.Z_i$
Rearranging the terms gives us:
$\dfrac{p_i.\beta_i.Z_i}{1-p_i} > \dfrac{p_j.\beta_j.Z_j}{1-p_j}$
Thus the jobs need to be arranged in a decreasing order of the value $D_i = \dfrac{p_i.\beta_i.Z_i}{1-p_i}$
Thus at each stage we will look at the value of D_i for all jobs and schedule the next job accordingly. Thus the schedule may keep on changing after each stage.

5 Question 5:

5.1 Part a:

We have $J_{N-1}(x) \leq J_N(x) \forall x \in X$
$J_k(i) = \min_{a_k \in A(x_k)} \mathbb{E}\left[g(i,a,j) + J_{k+1}(j)\right]$
Let the transition probabilities be P_{ij}
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \mathbb{E}\left[g(i,a,j) + J_{N-1}(j)\right]$
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j) + \sum_j P_{ij} J_{N-1}(j)\right]$
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j)\right] + \sum_j P_{ij} J_{N-1}(j)$
Similarly we have,
$J_{N-1}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j)\right] + \sum_j P_{ij} J_N(j)$
$\therefore J_{N-1}(i) - J_{N-2}(i) = \sum_j P_{ij}(J_N(j) - J_{N-1}(j)) \geq 0$ as $J_{N-1}(x) \leq J_N(x)$
Doing this for k times, by induction we get,
$J_{k+1}(x) \geq J_k(x)$

5.2 Part b:

We have $J_{N-1}(x) \geq J_N(x) \forall x \in X$
$J_k(i) = \min_{a_k \in A(x_k)} \mathbb{E}\left[g(i,a,j) + J_{k+1}(j)\right]$
Let the transition probabilities be P_{ij}
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \mathbb{E}\left[g(i,a,j) + J_{N-1}(j)\right]$
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j) + \sum_j P_{ij} J_{N-1}(j)\right]$
$J_{N-2}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j)\right] + \sum_j P_{ij} J_{N-1}(j)$
Similarly we have,
$J_{N-1}(i) = \min_{a_k \in A(x_k)} \left[\sum_j P_{ij} g(i,a,j)\right] + \sum_j P_{ij} J_N(j)$
$\therefore J_{N-1}(i) - J_{N-2}(i) = \sum_j P_{ij}(J_N(j) - J_{N-1}(j)) \leq 0$ as $J_{N-1}(x) \geq J_N(x)$
Doing this for k times, by induction we get,
$J_{k+1}(x) \leq J_k(x)$

6 Question 6:

6.1 Part a:

This problem is similar to the asset selling problem where:
$\text{Actions} \rightarrow \begin{cases} a_1 : \text{Keep on proofreading} \\ a_2 : \text{Stop the proofreading} \end{cases}$
After N students the course instructor has to publish if he/she chooses to do the proof reading till the end. After N stages the terminal cost incurred will be equal to $c_2.(d)$ where d is the number of errors left to be detected. So the horizon is 'N' in this problem.And the cost at each stage is c1 which is the amount charged by the student.
And the States:$x_{k+1} \rightarrow \begin{cases} d_i : \text{Number of unique errors left to be detected} \\ T : \text{ if } x_k = T \text{ or } a_k = a_2 \end{cases}$
$\therefore$ there are a total of X+1 states and number of stages is N.

6.2 Part b:

So $J_k(d_k) = \min_{a_k \in \{a_1,a_2\}} \mathbb{E}_{d_{k+1}} \left((g_k(d_k, a_k, d_{k+1}) + J_{k+1}(d_{k+1}))\right)$
$J_k(d_k) = \min_{a_k \in \{a_1,a_2\}} \mathbb{E}_{d_{k+1}} \left((g_k(d_k, a_k, d_{k+1}) + J_{k+1}(d_{k+1}))\right)$
d_{k+1} is from a binomial distibution.
$\therefore J_k(d_k) = \min_{a_k \in \{a_1,a_2\}} \left[c_1 + \sum_{i=1}^{d_k} J_{k+1}(d_k)^{d_k} C_i . p_k^{d_k - i}(1-p_k)^{d_k}, c_2.d_k\right]$

6.3 Part c:

$\therefore$ continue proofreading if:
$c_1 + \sum_{i=1}^{d_k} J_{k+1}(d_k)^{d_k} C_i . p_k^{d_k - i}(1-p_k)^{d_k}, c_2.d_k < c_2.d_k$
Let $\alpha_k = c_2.d_k - \sum_{i=1}^{d_k} J_{k+1}(d_k)^{d_k} C_i . p_k^{d_k - i}(1-p_k)^{d_k}$
$\therefore \alpha_{k+1} = c_2.d_{k+1} - \sum_{i=1}^{d_{k+1}} J_{k+2}(d_{k+1})^{d_{k+1}} C_i . p_{k+1}^{d_{k+1} - i}(1-p_{k+1})^{d_{k+1}}$
Note that $d_k \geq d_{k+1}$ as number of errors left in one stage will always be less than equal to the number of total errors detected till the next stage.
$\therefore \alpha_k - \alpha_{k+1} \geq 0$
Hence it is monotonic and a solution exists as proved in the asset selling problem in the class.

6.4 Part d:

When X is a random variable, we change the states to number of errors detected.
States: $x_{k+1} \rightarrow \begin{cases} x_k + d_i : \text{ where } d_i \text{is the number of unique errors detected in that stage} \\ T : \text{ if } x_k = T \text{ or } a_k = a_2 \text{ or } x_k = X \end{cases}$
Cost: $g_k(x_k, a_k, x_{k+1}, X) \rightarrow \begin{cases} c1 : \text{ if } a_k = a1 \\ c2.(X - x_k) : \text{ if } a_k = a_2 \end{cases}$
$\mathrm{g}_N(x_N, X) = c2.(X - x_N)$
$So\mathrm{J}_k(x_k) = \min_{a_k \in \{a_1,a_2\}} \mathbb{E}_{x_{k+1}} \left((g_k(x_k, a_k, x_{k+1}, X) + J_{k+1}(x_{k+1}, X))\right)$
This is because our next state as well as the cost also depends on X which is a random variable. So if we have actually detected all the errors before N stages then we will go in the termination state.Thus there will be a distribution of X which will have be taken into account when taking the expectation over x_{k+1}

7 References:

Question 1: https://cs.stackexchange.com/questions/23599/how-is-dynamic-programming-different-from-brute-force, http://www.control.lth.se/media/Education/EngineeringProgram/FRTN05/2015/lec12_2015.pdf
Question 2: Section 1.2 of the course notes.
Question 3:Problem 4.19 DPOC book, http://web.stanford.edu/class/msande351/homework/Homework2.pdf
Question 4: Example 4.5.1 - The Quiz Problem. DPOC Book
Question 5: Class notes.
Question 6: Asset Selling problem discussed in the class.
Discussed with Varun Sundar(EE16B068)(Problem 3 and 6) and Rishhanth Maanav(EE16B036)(Problem 4 and 6)