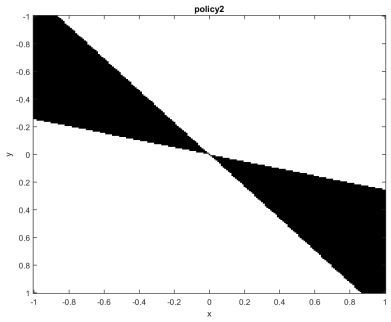
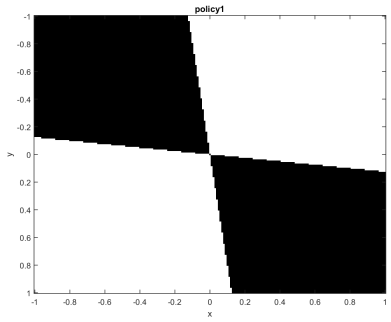
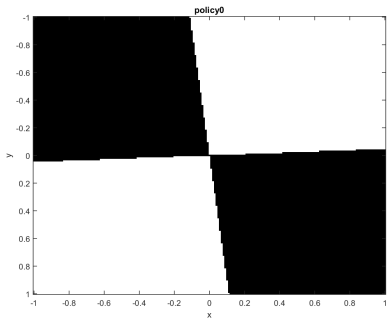
1 Problem3 plots



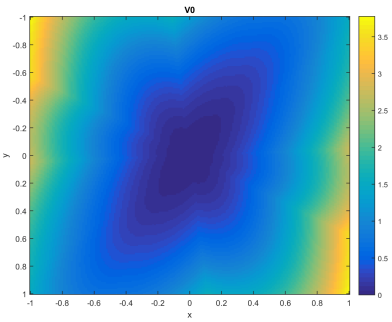
(a) policy2



(b) policy1



(c) policy0



(d) Optimal Value function

Figure 1: All the plots for problem 3b. Note that the dark region in all policies means optimal choice is select policy 1 and while region means select 2.

2 Problem 4

2.1 Problem Formulation

Due to the assumption of no cycles with negative cost, the optimal path need not have more than $|V|$ elements. Then we can formulate the DSP problem as a DFS with $T := |\mathcal{V}| - 1$ stages:

1.State space: $\mathcal{X}_0 := \{s\}, \mathcal{X}_T := \{\tau\}, \mathcal{X}_t := \mathcal{V} \backslash \{\tau\}$ for $t = 1, \ldots, T-1$

2.Control space: $u_{T-1} := \{\tau\}$ and $u_t := \mathcal{V} \backslash \{\tau\}$ for $t = 0, \ldots, T-2$

3.Dynamics(motion model): $x_{t+1} = u_t$ for $u_t \in \mathcal{U}_t, t = 0, \ldots, T-1$

4.Costs: $q(\tau) := 0$ and $\ell_t(x_t, u_t) = c_{x_t, u_t}$ for $t = 0, \ldots, T-1$

2.2 Technical Approach

By implementing Dynamic Programming algorithm on DSP problem, we can have the follow algoritm:

Because the DSP problem is symmetric: an optimal path from s to t is also a shortest path from t to s, where all arc directions are flipped.

For simplicity, we only need to implement label correcting (LC) algorithm. It is a general algorithm for SP problems that does not necessarily visit every node of the graph. It is shown as follow:

Algorithm 1 Deterministic Shortest Path via Dynamic Programming

Input: node set $\mathcal{V}$, start $s \in \mathcal{V}$, goal $\tau \in \mathcal{V}$, and costs c_{ij} for $i, j \in \mathcal{V}$
$T = |\mathcal{V}| - 1$
$V_T(\tau) = 0$
$V_{T-1}(i) = c_{i,\tau}, \quad \forall i \in \mathcal{V} \setminus \{\tau\}$
$\pi_{T-1}(i) = \tau, \quad \forall i \in \mathcal{V} \setminus \{\tau\}$
for $t = (T-2), \ldots, 0$ **do**
    $Q_t(i,j) = c_{i,j} + V_{t+1}(j), \quad \forall i, j \in \mathcal{V} \setminus \{\tau\}$
    $V_t(i) = \min_{j \in \mathcal{V} \setminus \{\tau\}} Q_t(i,j), \quad \forall i \in \mathcal{V} \setminus \{\tau\}$
    $\pi_t(i) = \arg\min_{j \in \mathcal{V} \setminus \{\tau\}} Q_t(i,j), \quad \forall i \in \mathcal{V} \setminus \{\tau\}$
    if $V_t(i) = V_{t+1}(i), \forall i \in \mathcal{V} \setminus \{\tau\}$ **then**
        break

Figure 2: Dynamic Programming algorithm for DSP problem

Algorithm 2 Deterministic Shortest Path via Forward Dynamic Programming

Input: node set $\mathcal{V}$, start $s \in \mathcal{V}$, goal $\tau \in \mathcal{V}$, and costs c_{ij} for $i, j \in \mathcal{V}$
$T = |\mathcal{V}| - 1$
$V_0^F(s) = 0$
$V_1^F(j) = c_{s,j}, \quad \forall j \in \mathcal{V} \setminus \{s\}$
for $t = 2, \ldots, T$ **do**
    $V_t^F(j) = \min_{i \in \mathcal{V} \setminus \{s\}} \left(c_{i,j} + V_{t-1}^F(i)\right), \quad \forall j \in \mathcal{V} \setminus \{s\}$
    if $V_t^F(\tau) < \infty$ **then**
        break

Figure 3: Forward Dynamic Programming algorithm for DSP problem

Algorithm 3 Label Correcting Algorithm

OPEN $\leftarrow \{s\}$, $g_s = 0$, $g_i = \infty$ for all $i \in \mathcal{V} \setminus \{s\}$
while OPEN is not empty **do**
    Remove i from OPEN
    for $j \in$ Children(i) **do**
        if $(g_i + c_{ij}) < g_j$ **and** $(g_i + c_{ij}) < g_\tau$ **then**
            $g_j \leftarrow (g_i + c_{ij})$
            Parent$(j) \leftarrow i$
            if $j \neq \tau$ **then**
                OPEN $\leftarrow$ OPEN $\cup \{j\}$

Figure 4: Label correcting algorithm for DSP problem

2.3 Results

2.3.1 p1

minimum cost path: 42 43 44 53 61 70 79 80 81 82 83 84 85 86 87 98 109

optimal cost-to-go values: 16.00 15.00 14.00 13.00 12.00 11.00 10.00 9.00 8.00 7.00 6.00 5.00 4.00 3.00 2.00 1.00 0.00

2.3.2 p2

minimum cost path: 20 31 42 53 64 75 86 97 108 109 120

optimal cost-to-go values: 67.66 67.41 66.94 66.11 64.53 61.38 55.53 46.00 32.62 16.53 0.00

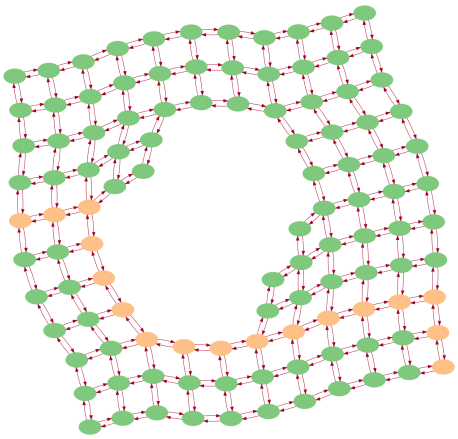


Figure 5: Plot for problem 1.

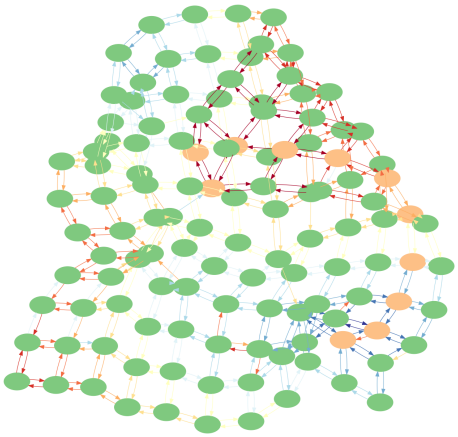


Figure 6: Plot for problem 2.

2.3.3 p3

minimum cost path: 2 5 9 13 18 23

optimal cost-to-go values: 6.24 4.83 3.41 2.41 1.41 0.00

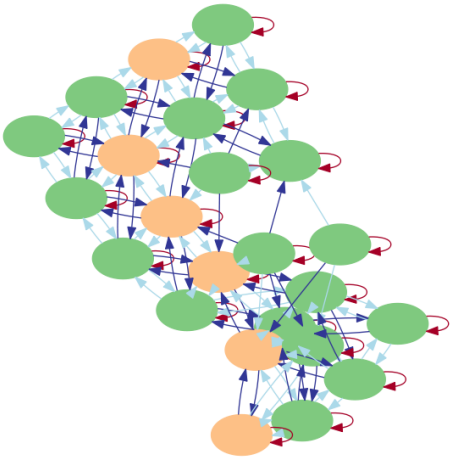


Figure 7: Plot for problem 3.

2.3.4 p4

minimum cost path: 11 22 33 44 55 66 67 68 69 70 71 72 83 84 85 96 97 108 109 120

optimal cost-to-go values: 82.52 82.27 81.79 80.97 79.61 77.52 74.48 70.33 64.93 58.18 50.12 41.16 32.30 24.61 18.83 14.55 10.69 6.89 3.21 0.00

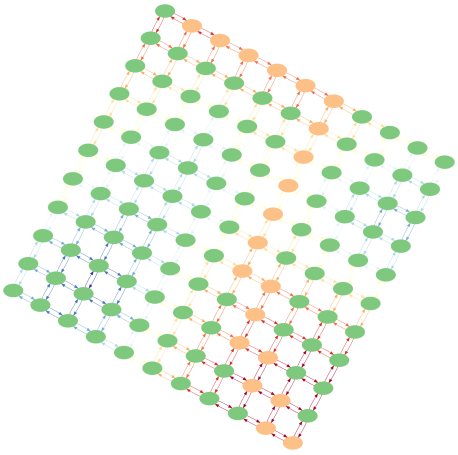


Figure 8: Plot for problem 4.

2.3.5 p5

minimum cost path: 0 11 22 23 24 25 26 27 28 29 30 31 32 41 50 58 67 76 87 98 109

optimal cost-to-go values: 41.66 39.66 37.66 35.66 33.66 31.66 29.66 27.66 25.66 23.66 21.66 19.66 17.66 15.66 13.66 11.66 9.66 7.66 5.66 2.45 0.00

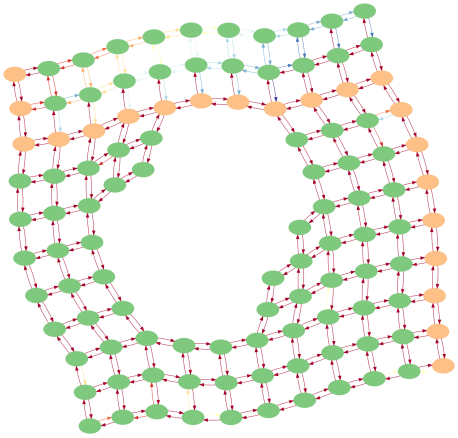


Figure 9: Plot for problem 5.

2.3.6 p6

minimum cost path: 27 38 37 45 46 109 120 119 118

optimal cost-to-go values: 43.18 42.49 42.21 41.96 29.97 22.60 16.67 4.04 0.00

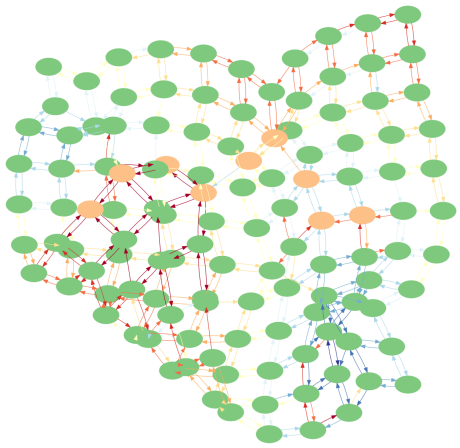


Figure 10: Plot for problem 6.

3 Problem 5

3.1 Problem Formulation

3.1.1 POMDP

We first define a POMDP $(\mathcal{X}, \mathcal{U}, \mathcal{Z}, p_f, p_h, \ell, \gamma)$ problem as follows:

1.State space $\mathcal{X}$: We keep track of the number of papers(np), scissors(ns) and rocks(nr) that opponent gives. Also, we keep track of our total score differential(Sd). Our state space are extremely huge. At time step t, we have (2*t+1) number of possible scores and $\begin{pmatrix} t+2 \\ 2 \end{pmatrix}$ number of possible number of combinations of total number of (paper ,scissors rocks). Thus,

$$(2*0+1)\begin{pmatrix} 2 \\ 2 \end{pmatrix} + (2*1+1)(2) + \ldots + (2*100+1)\begin{pmatrix} 102 \\ 2 \end{pmatrix} = 6\begin{pmatrix} 103 \\ 4 \end{pmatrix} + \begin{pmatrix} 103 \\ 3 \end{pmatrix} \tag{1}$$

number of states in total.

2.Control space $\mathcal{U}$: Control space is just what you move at each round.(paper, scissor or rock)

3.Observation space $\mathcal{Z}$: Observation space is win(1), draw(0) or lose(-1) at each round.

4.Observation model p_h: After each round, we have an observation of the game, which is win(+1), draw(0) or loss(-1), which is an absolute certain value.

5.Dynamics(motion model) p_f: Given history of opponent and my movement at time t, we need to formulate the distribution of next state $x_{t+1} \sim p_f(\cdot|x_t, u_t)$. Note that:

$$\begin{aligned}
&Pr(rock|X_r = x_r, X_p = x_p, X_s = x_s) = \\
&Pr(rock|F = rock, X_r = x_r, X_p = x_p, X_s = x_s)Pr(F = rock|X_r = x_r, X_p = x_p, X_s = x_s) \\
&+ Pr(rock|F = paper, X_r = x_r, X_p = x_p, X_s = x_s)Pr(F = paper|X_r = x_r, X_p = x_p, X_s = x_s) \\
&+ Pr(rock|F = scissors, X_r = x_r, X_p = x_p, X_s = x_s)Pr(F = scissors|X_r = x_r, X_p = x_p, X_s = x_s)
\end{aligned}$$

Since

$$\Pr\left(\text{rock}\,|F = \text{rock}, X_r = x_r, X_p = x_p, X_s = x_s\right)_s = 0.5 \tag{2}$$

$$\Pr\left(F = \text{rock}\,|X_r = x_r, X_p = x_p, X_s = x_s\right) = \frac{\Pr\left(X_r = x_r, X_p = x_p, X_s = x_s|F = \text{rock}\right) * \Pr\left(F = \text{rock}\right)}{\Pr\left(X_r = x_r, X_p = x_p, X_s = x_s\right)} \tag{3}$$

Assume that the prior $\Pr(F = \text{rock}) = \Pr(F = \text{scissor}) = \Pr(F = \text{paper}) = \frac{1}{3}$,

Also,

$$\begin{aligned}\Pr(F = \text{rock}\,|X_r = x_r, X_p = x_p, X_s = x_s) + \Pr(F = \text{scissor}\,|X_r = x_r, X_p = x_p, X_s = x_s) \\ + \Pr(F = \text{paper}\,|X_r = x_r, X_p = x_p, X_s = x_s) = 1\end{aligned} \tag{4}$$

Here we have

$$\Pr(F = \text{rock}\,|X_r = x_r, X_p = x_p, X_s = x_s) = \frac{\begin{pmatrix} x_r + X_p + X_s \\ x_r \end{pmatrix}}{\begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}} \tag{5}$$

In this case, we have

$$\begin{aligned}&\Pr(\text{rock}\,|X_r = x_r, X_p = x_p, X_s = x_s) = \\ &\frac{0.5 * \begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}{\begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}\end{aligned} \tag{6}$$

We have calculate the conditional probability on State X_t, then we can formulate the motion model by updating X_{t+1}. Without the loss of generality, assume we have the movement$u_t = paper$, then

$$\begin{aligned}&\Pr\left(X_r^{t+1} = x_r + 1, X_p^{t+1} = x_p, X_s^{t+1} = x_s, Sd^{t+1} = sd + 1 | U_t = paper\right) = \\ &\Pr\left(\text{rock}\,|X_r^t = x_r, X_p^t = x_p, X_s^t = x_s, Sd^t = sd\right) = \\ &\frac{0.5 * \begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}{\begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}\end{aligned} \tag{7}$$

$$\begin{aligned}&\Pr\left(X_r^{t+1} = x_r, X_p^{t+1} = x_p + 1, X_s^{t+1} = x_s, Sd^{t+1} = sd | U_t = paper\right) = \\ &\Pr\left(\text{paper}\,|X_r^t = x_r, X_p^t = x_p, X_s^t = x_s, Sd^t = sd\right) = \\ &\frac{0.5 * \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}{\begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}\end{aligned} \tag{8}$$

$$\begin{aligned}&\Pr\left(X_r^{t+1} = x_r, X_p^{t+1} = x_p, X_s^{t+1} = x_s + 1, Sd^{t+1} = sd - 1 | U_t = paper\right) = \\ &\Pr\left(\text{scissor}\,|X_r^t = x_r, X_p^t = x_p, X_s^t = x_s, Sd^t = sd\right) = \\ &\frac{0.5 * \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + 0.25 * \begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix}}{\begin{pmatrix} x_r + x_p + x_s \\ x_r \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_p \end{pmatrix} + \begin{pmatrix} x_r + x_p + x_s \\ x_s \end{pmatrix}}\end{aligned} \tag{9}$$

which is similar for $U_t = rock, scissor$.

6.Initial state x_0: $x_0 = \{x_s = x_r = x_p = 0, sd = 0\}$

7.Planning horizon: T = 100.

8.Costs: We only have the terminal cost here, we define $q(Sd) := -Sd$, which is the negtive of the score differential(Sd).

Here is the summation of POMDP formulation.

$$\begin{array}{l} \min_{\pi_{0:T-1}} E\left[\gamma^T q\left(x_T\right) + \sum_{t=0}^{T-1} \gamma^t \ell_t\left(x_t, u_t\right)\right] \\ \text{s.t.} \quad x_{t+1} \sim p_f\left(\cdot|x_t, u_t\right), \quad t = 0, \ldots, T-1 \\ \quad z_{t+1} \sim p_h\left(\cdot|x_t\right), \quad t = 0, \ldots, T-1 \\ u_t \sim \pi_t\left(\cdot|i_t\right), \quad t = 0, \ldots, T-1 \\ x_0 \sim b_0(\cdot) \equiv \text{ prior pdf over the hidden state } x_0 \end{array} \tag{10}$$

3.1.2 MDP

Then, we use the equavalence to formulate MDP($\mathcal{B}$,$\mathcal{U}$, p_ψ, ρ, γ). Here we define our MDP problem as follows:

1.State space $\mathcal{B} := \mathcal{P}(\mathcal{X})$ is the continuous space of pmfs over $\mathcal{X}$, which has been defined above.

2.Control space: Similar as POMDP, control space is just what you move at each round.(paper, scissor or rock)

3.Transition model p_ψ: In MDP, we need to eliminate our observation space. The transformed motion model is the Bayes filter $b_{t+1} = \psi\left(b_t, u_t, z_t\right)$, where z_t plays the role of noise or in probabilistic terms. Given the distribution of Z_{t+1}, b_t, u_t, we can easily formulate b_{t+1}. Because at each round, given my movement, the probability of win, draw and lose is the probability of +1,0,-1 to Sd, and according to u_t, we can easily formulate what our opponent's movement as follows:

Transition model	$u_t = paper$	$u_t = scissor$	$u_t = rock$
$z_t = win(+1)$	$x_r+ = 1, sd+ = 1$	$x_p+ = 1, sd+ = 1$	$x_s+ = 1, sd+ = 1$
$z_t = draw(0)$	$x_p+ = 1$	$x_s+ = 1$	$x_r+ = 1$
$z_t = loss(-1)$	$x_s+ = 1, sd- = 1$	$x_r+ = 1, sd- = 1$	$x_p+ = 1, sd- = 1$

4.Initial state b_0: $b_0 = \{x_s = x_r = x_p = 0, sd = 0\}$

5.Planning horizon: T = 100.

6.Costs: We only have the terminal cost here, we define $q(Sd) := -Sd$, which is the negative of the score differential(Sd).

Here is the summation of MDP formulation:

$$\min_{\pi_{0:T-1}} V_0^\pi\left(b_0\right) = E\left[\gamma^T \rho_T\left(b_T\right) + \sum_{t=0}^{T-1} \gamma^t \rho_t\left(b_t, u_t\right)\right] \tag{11}$$

$$\begin{array}{l} \text{s.t. } b_{t+1} = \psi\left(b_t, u_t, z_{t+1}\right), \quad t = 0, \ldots, T-1 \\ \quad z_{t+1} \sim \eta\left(\cdot|b_t, u_t\right), \quad t = 0, \ldots, T-1 \\ \quad u_t \sim \pi_t\left(\cdot|b_t\right), \quad t = 0, \ldots, T-1 \end{array} \tag{12}$$

3.2 Technical Approach

For simplicity, we first propose the optimal policy and then prove its optimality.

3.2.1 Optimal Policy

By applying dynamic programming algorithm on our model, we can formulate the optimal policy as follows: Each round, we choose the movement with largest count as opponent's favor. Then, we select our movement according to opponent's favor to beat him.(i.e. $F^t = argmax\{x_p^t, x_s^t, x_r^t\}$) We randomly choose movement when 3 action history are equal.

3.2.2 Prove of Optimality

Algorithm 1 Dynamic Programming

Input: MDP $(\mathcal{X}, \mathcal{U}, p_f, \ell_t, \mathfrak{q}, \gamma)$, initial state $x_0 \in \mathcal{X}$, and horizon T

$V_T(x) = \mathfrak{q}(x), \quad \forall x \in \mathcal{X}$

for $t = (T-1) \dots 0$ **do**

$\quad Q_t(x,u) \leftarrow \ell_t(x,u) + \gamma \mathbb{E}_{x' \sim p_f(\cdot|x,u)} \left[V_{t+1}(x') \right], \quad \forall x \in \mathcal{X}, u \in \mathcal{U}(x)$

$\quad V_t(x) = \min_{u \in \mathcal{U}(x)} Q_t(x,u), \qquad \forall x \in \mathcal{X}$

$\quad \pi_t(x) = \arg\min_{u \in \mathcal{U}(x)} Q_t(x,u), \qquad \forall x \in \mathcal{X}$

return policy $\pi_{0:T-1}$ and value function V_0

Figure 11: DP algorithm

Inferring DP algorithm, we only have terminal cost, so here $\ell_t(x,u) = 0$. So, our optimal policy should be choosen from

$$\pi_t(x) = \arg\min_{u \in \mathcal{U}(x)} Q_t(x,u), \quad \forall x \in \mathcal{X} \tag{13}$$

which is equal to

$$\pi_t(x) = \arg\min_{u \in \mathcal{U}(x)} E_{x' \sim p_f(\cdot|x,u)} \left[V_{t+1}(x') \right] = \arg\max_{u \in \mathcal{U}(x)} E_{x' \sim p_f(\cdot|x,u)} \left[Sd' \right], \quad \forall x \in \mathcal{X} \tag{14}$$

According to the motion model defined above,

$$\begin{aligned} Q_t(x,u) &= \Pr\left(Sd^{t+1} = sd+1 | U_t\right) * (sd+1) + \Pr\left(Sd^{t+1} = sd | U_t\right) * (sd) \\ &\quad + \Pr\left(Sd^{t+1} = sd-1 | U_t\right) * (sd-1) \\ &= sd + \Pr\left(Sd^{t+1} = sd+1 | U_t\right) - \Pr\left(Sd^{t+1} = sd-1 | U_t\right) \end{aligned} \tag{15}$$

Thus,

$$\begin{aligned} \pi_t(x) &= \arg\max_{u \in \mathcal{U}(x)} E_{x' \sim p_f(\cdot|x,u)} \left[Sd' \right], \quad \forall x \in \mathcal{X} \\ &= \arg\max_{u \in \mathcal{U}(x)} \left[\Pr\left(Sd^{t+1} = sd+1 | U_t\right) - \Pr\left(Sd^{t+1} = sd-1 | U_t\right) \right] \\ &= \arg\max_{u \in \mathcal{U}(x)} \left[\begin{pmatrix} x_r + x_p + x_s \\ x_{u_{win}} \end{pmatrix} - \begin{pmatrix} x_r + x_p + x_s \\ x_{u_{lose}} \end{pmatrix} \right] \end{aligned} \tag{16}$$

So, we can easily observed that we need to choose a movement(u_{win}) which can win opponent's top number of action($x_{u_{win}}$).

3.3 Result

We can observed in the plot that our optimal policy is indeed optimal. With increasing number of games, the mean score differential continuous increasing, while stochastic and deterministic policy are around 0. Also, optimal policy tend to be a straight line in mean plot when number of games are becoming large, since our evidence are accumulating, we are more absolute about our inference result. Also, the mean score differential of our optimal policy approximate 25 out of 100 games, which means 50% chance of winning, 25% drawing and 25% of losing.

From standard deviation plot, we can observe that all policy are showing increasing trend, since our state space is increasing as time increase. Moreover, the optimal policy has largest standard deviation, which might because our optimal policy is dependent of our opponent's behavior, if our opponent's movement are abnormal at very beginning, our policy might choose some bad result.

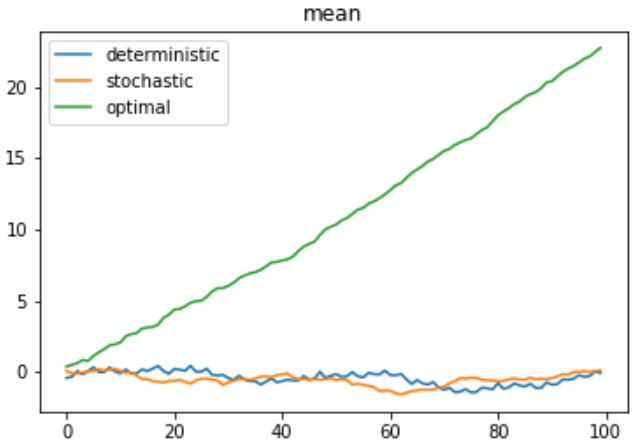


Figure 12: Mean over number of games.

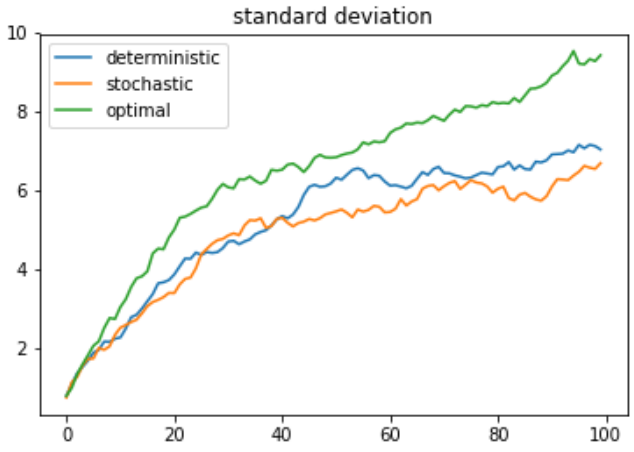


Figure 13: Standard Deviation over number of games.