Reinforcement Learning HW2

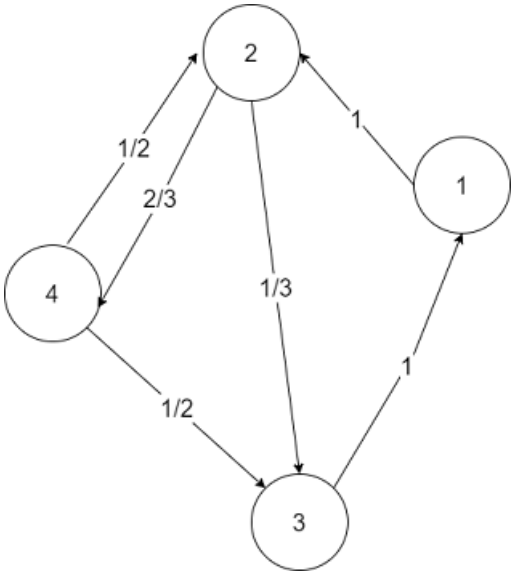
Yonatan Ariel Slutzky Eyal Grinberg

20 April 2023

1 Theoretical Questions

1.1 Question 1

1.1.1 Section 1



1.1.2 Section 2

The following is a cycle with a positive probability:

$$P(1 \to 2 \to 4 \to 3 \to) = 1 \cdot \frac{2}{3} \cdot \frac{1}{2} \cdot 1 > 0$$

Thus, all of the states belong to the same communicating class, and the chain is irreducible.

1.1.3 Section 3

We'll prove the period of state 2 is 1, and thus since the chain is irreducible, this is also the period for the rest of the states. This will also mean that the chain is a-periodic.
The following holds:

$$P(2 \to 4 \to 2) = \frac{2}{3} \cdot \frac{1}{2} > 0$$

$$P(2 \to 3 \to 1 \to 2) = \frac{1}{3} \cdot 1 \cdot 1 > 0$$

Thus, 2's period equals $GCD(2, 3) = 1$.

1.1.4 Section 4

We'll solve the following equations system to find the invariant distribution which exists since the chain is irreducible and a-periodic:

$$\sum_{i=1}^{4} \mu_i = 1 \wedge \mu^\top P = \mu^\top \to$$

$$\mu_3 = \mu_1$$

$$\mu_1 + \frac{1}{2}\mu_4 = \mu_2 \to \mu_1 = \mu_4$$

$$\frac{1}{3}\mu_2 + \frac{1}{2}\mu_4 = \mu_3$$

$$\frac{2}{3}\mu_2 = \mu_4$$

Plugging into the constraint gives

$$1 = 4.5\mu_4 \to \mu = (\frac{2}{9}, \frac{1}{3}, \frac{2}{9}, \frac{2}{9})^\top$$

1.1.5 Section 5

Using the invariant distribution, we get that

$$\forall i \in \{1, 3, 4\}.E(T_i) = \frac{1}{\mu_i} = 4.5, E(T_2) = \frac{1}{\mu_2} = 3$$

Therefore, by definition, all states are positive recurrent (this is also a class property which makes sense).

1.1.6 Section 6

We'll define the following transition matrix:

$$P' = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In this MC, the cycle $1 \to 2 \to 3 \to 1$ has probability 1, and thus $E(T_1) = 3$. Since 4 is in a different class, we also get that $d_1 = 3$.
Therefore, since $s_0 = 1$, we get that $\forall m \in \mathbb{N} \cup \{0\}.P^m_{1,1} = \chi_{m\%3=0}$.

1.2 Question 2

1.2.1 Section 1

We'll denote the following MDP:

$$S := [2k-1] \setminus \{0\}, s_0 = k$$

$$A(.) := \{(.+1)\%2k, (.-1)\%2k\}$$

$$\forall i,j \in S.P((.+1)\%2k)_{ij} = \chi_{j=(i+1)\%2k}$$

$$\forall i,j \in S.P((.-1)\%2k)_{ij} = \chi_{j=(i-1)\%2k}$$

$$\forall s \in S, a \in A.R(s,a) = \chi_{s=0}$$

1.2.2 Section 2

The optimal policy must first reach 0 from k. The least amount of steps possible for doing this is k steps, since $k - 0 = k = 2k - k$.
After reaching 0, the optimal policy will choose either 1 or $2k-1$, and then revert back to 0 indefinitely (since 0 is the only state that receives any positive reward).
Therefore, an optimal policy is:

$$\forall s \in S \setminus \{0\}.\pi(s)(s-1)\%2k, \pi(0) = 1$$

1.2.3 Section 3

By definition of the value iteration algorithm, we get the following:

$$\forall s \in S.V_1(s) = max_{a \in A}(r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)V_0(s')) =$$

$$= r(s,a) = \chi_{s=0}$$

Thus, the only state which changes its value is 0, with the new value being 1.

1.2.4 Section 4

By definition of the value iteration algorithm, we get the following:

$$\forall s \in S. V_2(s) = max_{a \in A}(r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)V_1(s')) =$$

$$= \chi_{s=0} + \gamma \chi_{s \in \{1,2k-1\}}$$

Thus, the only states which change their values are $1, 2k-1$, with the new values being γ.

1.2.5 Section 5

We'll solve the problem for a general $k \in \mathbb{N}$, and then plug in $k = 2$ to solve the problem in our case.
The optimal policy from every $s \in S$ first reaches 0 with as few steps as possible ($s = 0$ takes 0 steps).
Then, the optimal policy will choose either 1 or $2k-1$, and then revert back to 0 indefinitely.
Therefore, since $r(s,a) = \chi_{s=0}$ we get that

$$V^*(s) = \gamma^{min(s,2k-s)} \sum_{i=0}^{\infty} \gamma^{2i} = \frac{\gamma^{min(s,2k-s)}}{1-\gamma^2} =$$

Plugging in the states for $k = 2$ gives

$$V^*(0) = \frac{1}{1-\gamma^2}, V^*(1) = \frac{\gamma}{1-\gamma^2}, V^*(2) = \frac{\gamma^2}{1-\gamma^2}, V^*(3) = \frac{\gamma}{1-\gamma^2}$$

1.3 Question 3

1.3.1 Section 1

We'll denote the following MDP:

$$S := \{(b,d) | b \subseteq \{0,..,N\}, d \in \{0,..,9,x\}\}$$

$$\forall d \in \{0,..,9\}. P(s_0 = (\{0,..,N\}, d)) = \frac{1}{10}$$

Each state is a tuple comprised of the empty slots so far, and the last drawn digit. x represents the end of the game when it appears as the digit.

$$A(b,d) = \{i \leftarrow d | i \in b\}$$

The available actions from the current empty slots b and the current digit d is placing d in each of the empty slots of b.

$$\forall (b,d) \in S. i \in b, d' \in \{0,..,9\}. P((b \setminus \{i\}, d') | (b,d), i \leftarrow d) = \frac{1}{10}$$

Given the chosen slot and the current empty slots and digit, the probability to remove the chosen slot from the empty slots and to draw another digit is $\frac{1}{10}$ for every digit.

$$\forall (b,d) \in S.i \in b.R((b,d), i \leftarrow d) = d \cdot 10^i$$

The terminal reward for every terminal state is 0, since the immediate rewards sum to the desired total reward.

1.3.2 Section 2

We'll first quote Hardy's theorem - Let $n \in \mathbb{N}, x_1 \leq .. \leq x_n, y_1 \leq .. \leq y_n$. The following holds

$$max_{\{i_j\}_{j=1}^n \in Perm([n])} \sum_{j=1}^{n} x_{i_j} y_j = \sum_{j=1}^{n} x_j y_j$$

In other words, the permutation which maximizes the above sum of pairs, is the one that pairs the smallest x with the smallest y, the second smallest x with the second smallest y and so on. A simple proof by contradiction proves the theorem.
Let $n \in [N]$. We'll denote the available slots $\{i_j\}_{j=1}^n \subseteq [N] \cup \{0\}$.
We'll denote $f(i_1, .., i_n)$ to be the expected optimal value (over the first digit drawn), and $f(i_1, ., , i_n|d)$ to be the expected optimal value given that the first digit is d.
By convention, f of the empty set will be 0.
We'll prove by induction on n the amount of available slots that there exist numbers

$$-\infty = a_{0,n} \leq a_{1,n} \leq ... \leq a_{n,n} = \infty$$

such that the optimal choice in the first step is to pick the jth slot if the digit drawn is contained in $(a_{j-1,n}, a_{j,n}]$, where the numbers $\{a_{j,n}\}_{j=1}^{n-1}$ aren't dependant on the available slots.
In addition,

$$f(i_1, .., i_{n-1}) = \sum_{j=1}^{n-1} 10^{i_j} a_{j,n}$$

Proving the above will show that the optimal policy in each stage is only dependent on the drawn digit and on the amount of available slots left, and isn't dependent on the identity of the available slots.
$\underline{n = 1:}$ In this case,

$$-\infty = a_{0,1}, \infty = a_{1,1}$$

and indeed, we only have a single option of picking the slot.
$\underline{n = 2:}$ In this case,

$$-\infty = a_{0,2}, \infty = a_{1,2}$$

Denoting $a_{1,2} = 4.5$ will result in the desired behaviour (this can be proven easily, and shown empirically in the next section).

In addition, we get that

$$f(i_1) = \frac{1}{10}\sum_{d=0}^{9} d \cdot 10^{i_1} = \sum_{j=1}^{n-1} 10^{i_j} a_{j,n}$$

$\underline{n \to n+1}$: Let $n \in [N]$. Suppose the argument is true for n. Let d be the first drawn digit. By definition,

$$f(i_1, .., i_{n+1}|d) = max_{k\in[n+1]}(d \cdot 10^{i_k} + f(i_1, .., i_{k-1}, i_{k+1}, .., i_{n+1})) = (*)$$

By the inductive assumption, we get that the optimal policy of the n empty slots problem decides on the next slot independently of the identity of the available slots.
Thus, we'll denote $a_{j,n+1}$ to be the expected size of the digit the optimal policy assigns to the jth slot in the n empty slots problem. These values are independent of the available slots.
Therefore, we get that

$$f(\bar{i}_1, .., \bar{i}_n) = \sum_{j=1}^{n} 10^{\bar{i}_j} a_{j,n+1}$$

Since f is maximal, we get by Hardy's theorem and since $10^{\bar{i}_1} \leq .. \leq 10^{\bar{i}_n}$ that

$$a_{1,n+1} \leq .. \leq a_{n,n+1}$$

We'll also denote by convention

$$-\infty = a_{0,n+1}, \infty = a_{1,n+1}$$

Hence,

$$(*) = max_{k\in[n+1]}(d \cdot 10^{i_k} + \sum_{j=1}^{k-1} 10^{i_j} a_{j,n+1} + \sum_{j=k+1}^{n} 10^{i_j} a_{j,n+1})$$

Employing Hardy's theorem, we get that k^* is that for which

$$d \in (a_{k^*-1,n+1}, a_{k^*,n+1}]$$

since otherwise we could find an order for which the value is strictly greater. Therefore, the optimal policy will choose the jth slot if the first digit is within $(a_{j-1,n+1}, a_{j,n+1}]$ concluding the proof. QED
The above proof is adapted from the paper A Sequential Stochastic Assignment Problem.

1.3.3 Section 3

We'll compute the optimal policy using backward dynamic programming:
- When no slots are available, the game has ended and the optimal policy makes

no moves. Thus, $\pi_3^*(b,d) = \emptyset$.
- When there's a single slot available, the optimal policy simply plugs the digit into it. Formally,

$$\forall i \in \{0,1,2\}, d \in \{0,..,9\}.\pi_2^*(\{i\},d) = i \leftarrow d$$

- Let $j < i \in \{0,1,2\}$. Thus, we get that $\forall d \in \{0,..,9\}$,

$$\pi_1^*(\{0,1,2\}\backslash\{i,j\},d) = \begin{cases} i, & d \cdot 10^i + \frac{1}{10}\sum_{d'=0}^{9} d' \cdot 10^j > d \cdot 10^j + \frac{1}{10}\sum_{d'=0}^{9} d' \cdot 10^i \\ j, & else \end{cases} \leftarrow d =$$

$$\begin{cases} i, & d \cdot 10^i + 4.5 \cdot 10^j > d \cdot 10^j + 4.5 \cdot 10^i \\ j, & else \end{cases} \leftarrow d$$

$$(i \cdot \chi_{d>4.5} + j \cdot \chi_{d\leq 4.5}) \leftarrow d =$$

- For $\forall d \in \{0,..,9\}$ we get that $\pi_0^*(\emptyset,d)$ chooses to plug d into the slot which achieves the maximal option below:

$$0 : d + \frac{1}{10} max_{i\in\{1,2\},j\in\{1,2\}\backslash\{i\}}(\sum_{d'=0}^{9}(d' \cdot 10^i + 4.5 \cdot 10^j)) = d + 495$$

$$1 : 10d + \frac{1}{10} max_{i\in\{0,2\},j\in\{0,2\}\backslash\{i\}}(\sum_{d'=0}^{9}(d' \cdot 10^i + 4.5 \cdot 10^j)) = 10d + 454.5$$

$$2 : 100d + \frac{1}{10} max_{i\in\{0,1\},j\in\{0,1\}\backslash\{i\}}(\sum_{d'=0}^{9}(d' \cdot 10^i + 4.5 \cdot 10^j)) = 100d + 49.5$$

Therefore,

$$\forall d \in \{0,..,4\}.\pi_0^*(\emptyset,d) = 0$$
$$\forall d \in \{5,..,9\}.\pi_0^*(\emptyset,d) = 2$$

1.4 Question 4

Let $V_1, V_2 \in \mathbb{R}^{|S|}$. We get the following

$$\forall s \in [|S|].|(T(V_1))_s - (T(V_2))_s| =$$

$$= |\frac{1}{|A|}\sum_{a\in A}(r(s,a)+\gamma\sum_{s'\in S}p(s'|s,a)(V_1)_{s'}) - \frac{1}{|A|}\sum_{a\in A}(r(s,a)+\gamma\sum_{s'\in S}p(s'|s,a)(V_2)_{s'})| \leq$$

$$\frac{\gamma}{|A|}\sum_{a\in A}\sum_{s'\in S}p(s'|s,a)|(V_1)_{s'} - (V_2)_{s'}| = \frac{\gamma|A|}{|A|}\cdot 1 \cdot |(V_1)_{s'} - (V_2)_{s'}| \leq \gamma||V_1 - V_2||_\infty$$

Thus,

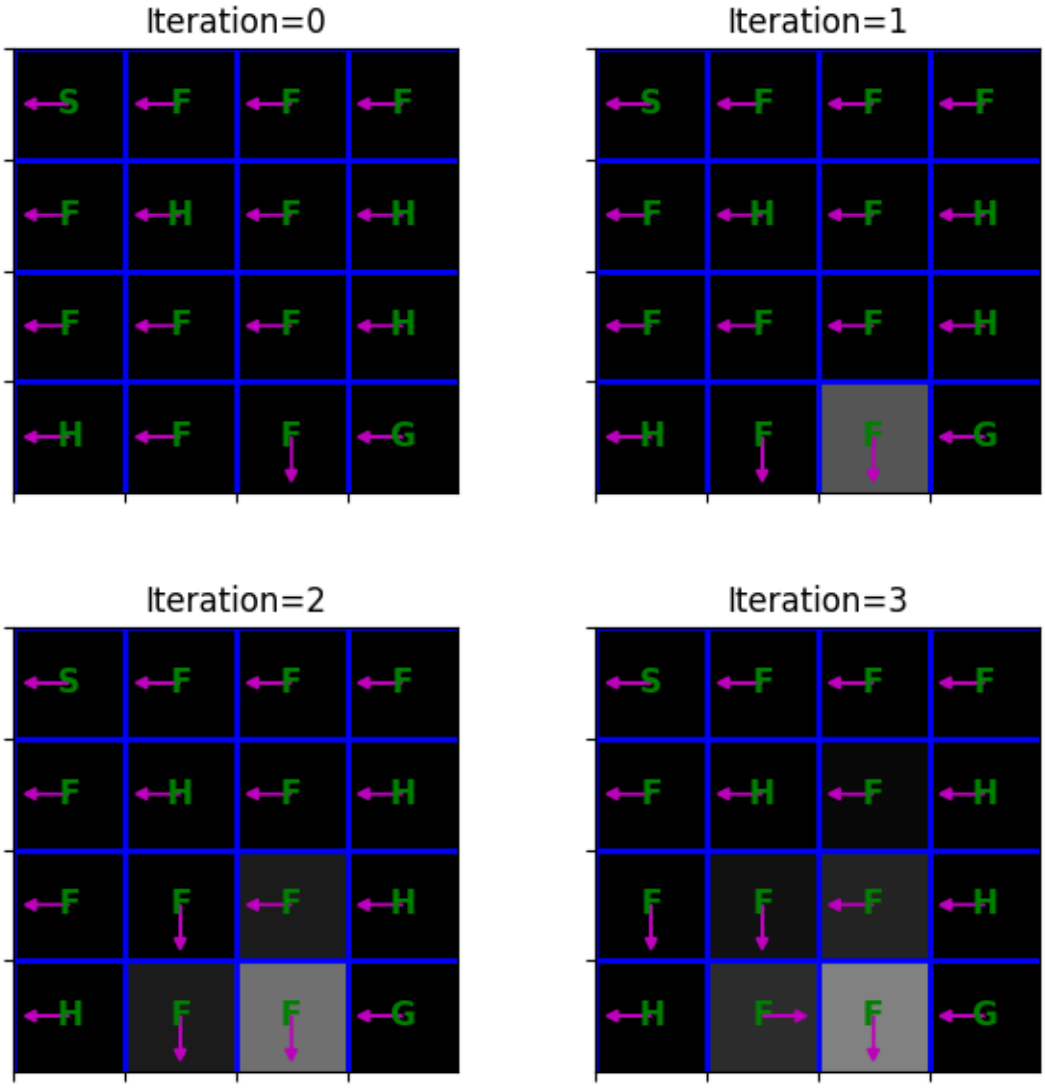
$$||T(V_1) - T(V_2)||_\infty \leq \gamma||V_1 - V_2||_\infty$$

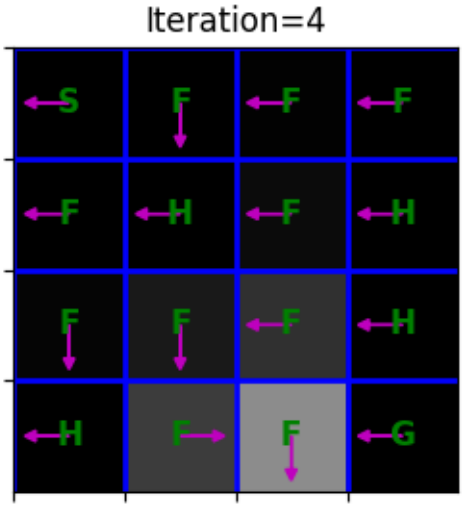
Hence, T is a γ-contracting operator w.r.t $||.||_\infty$. QED

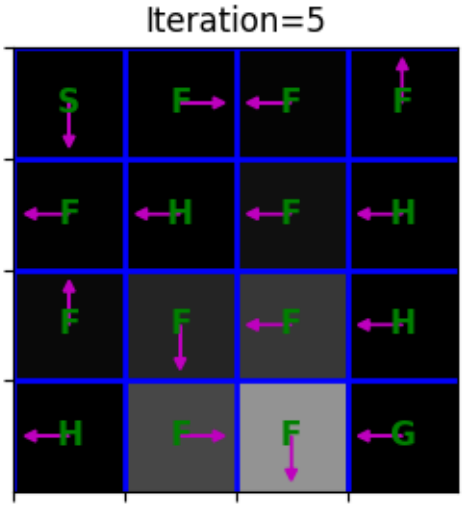
2 Practical Questions

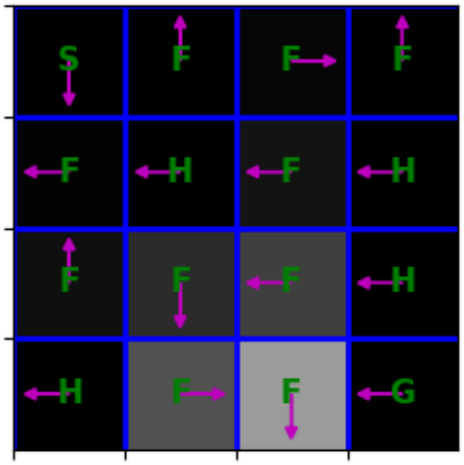
2.1 Question 1

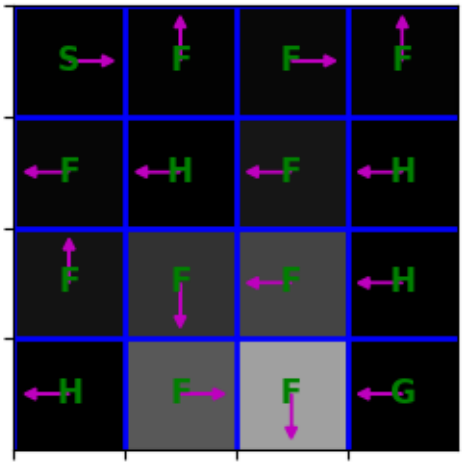
The following are the optimal policies in each iteration:

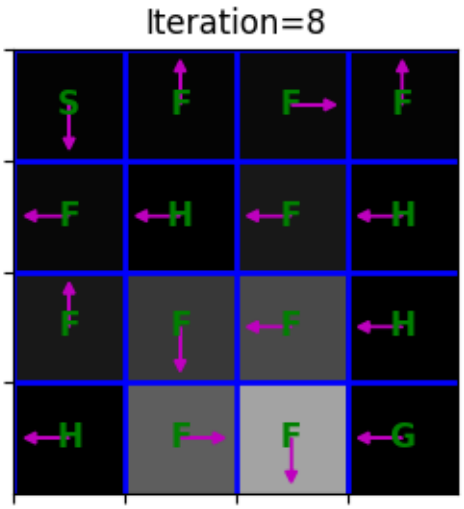


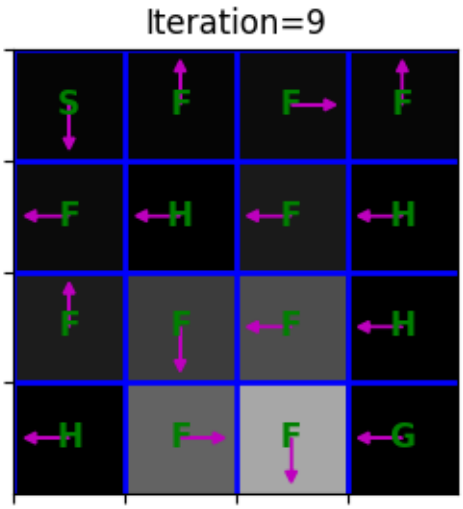

Iteration=4
S F F F
F H F H
F F F H
H F F G

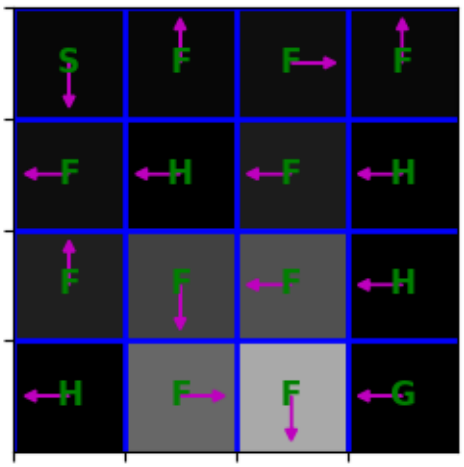

Iteration=5
S F F F
F H F H
F F F H
H F F G

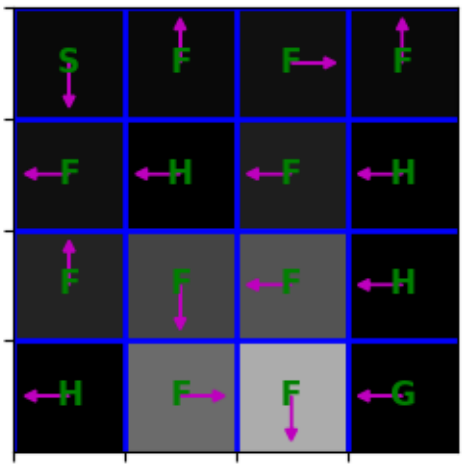

Iteration=6
S F F F
F H F H
F F F H
H F F G

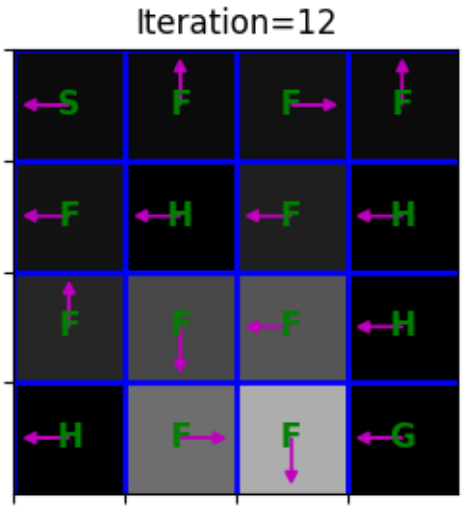

Iteration=7
S F F F
F H F H
F F F H
H F F G

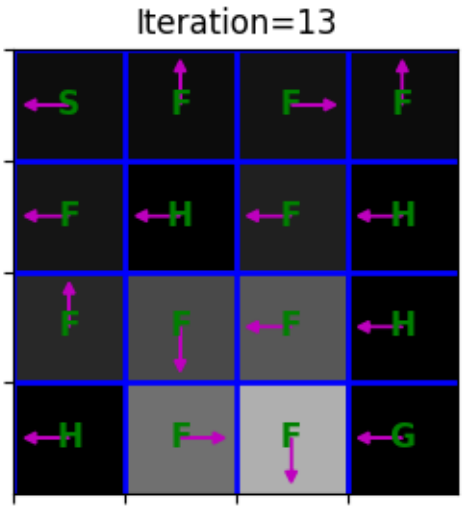

Iteration=8
S F F F
F H F H
F F F H
H F F G

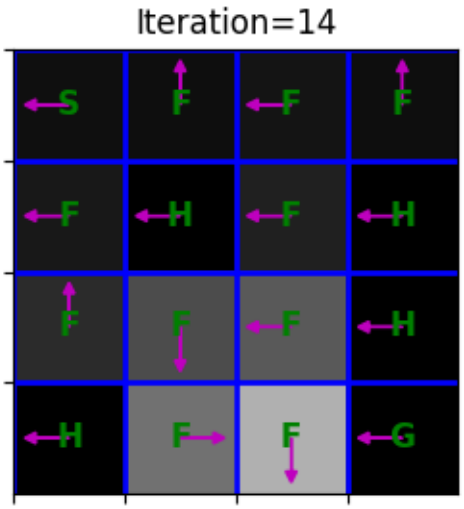

Iteration=9
S F F F
F H F H
F F F H
H F F G

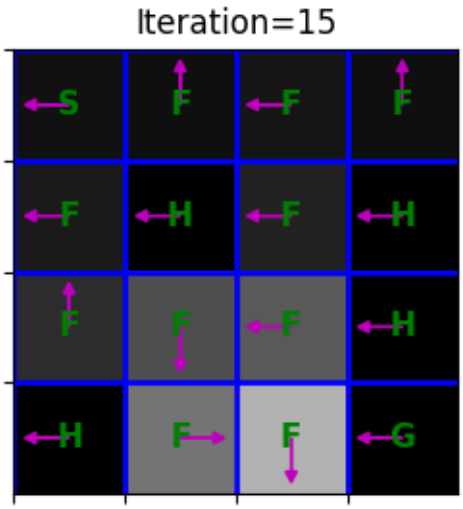

Iteration=10
S F F F
F H F H
F F F H
H F F G

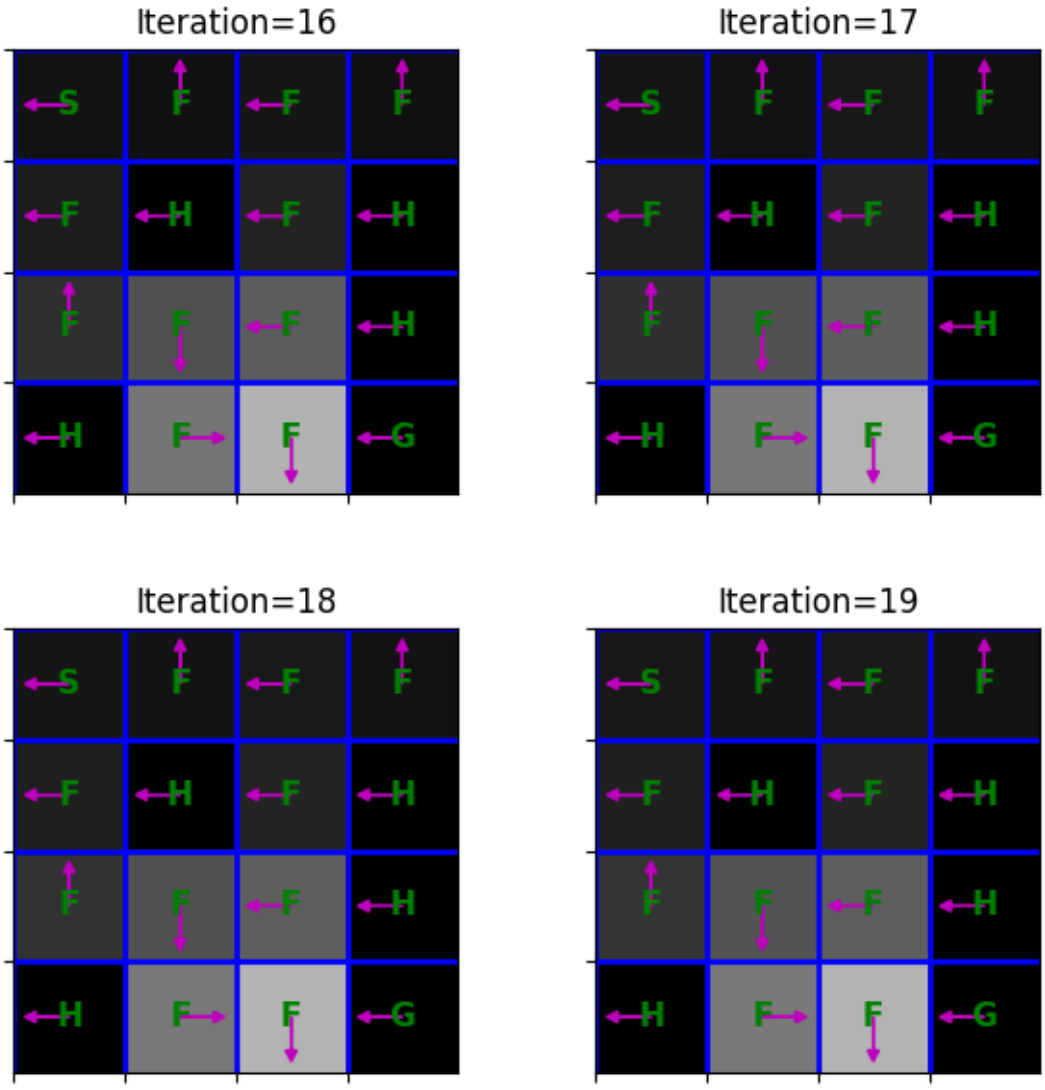

Iteration=11
S F F F
F H F H
F F F H
H F F G


Iteration=12
S F F F
F H F H
F F F H
H F F G


Iteration=13
S F F F
F H F H
F F F H
H F F G


Iteration=14
S F F F
F H F H
F F F H
H F F G

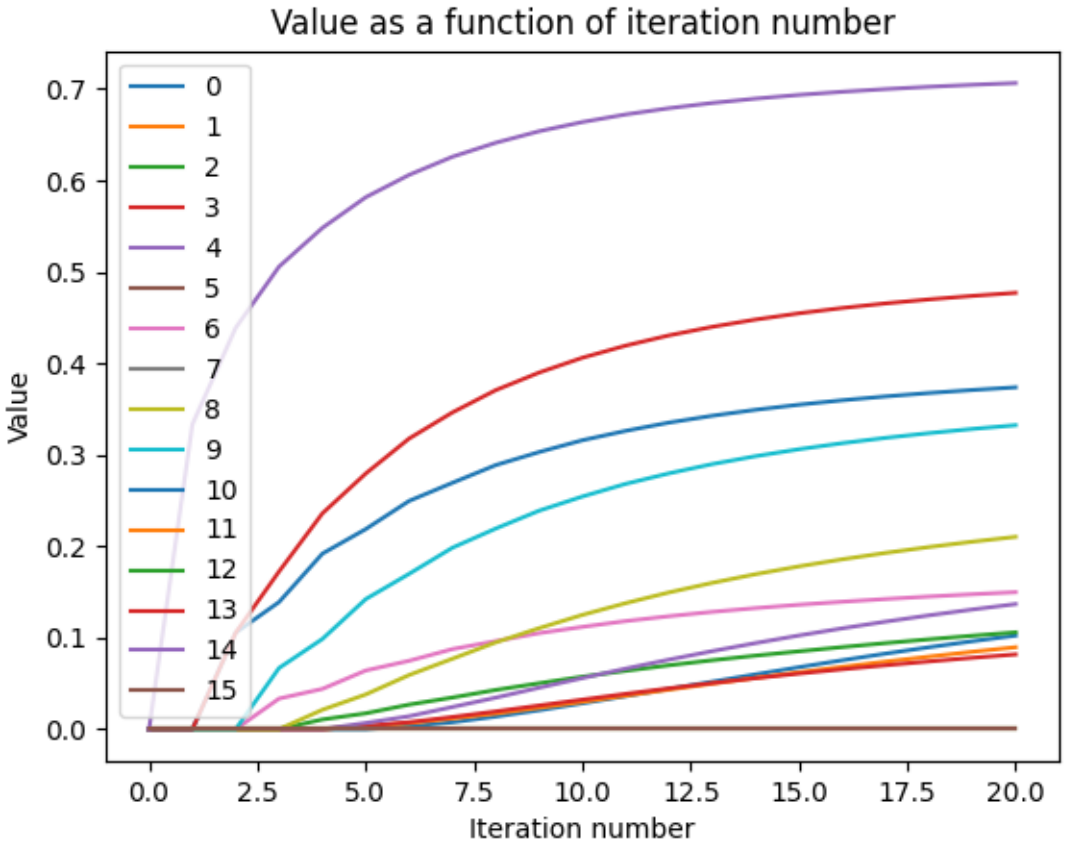

Iteration=15
S F F F
F H F H
F F F H
H F F G



The following is the progression of value as a function of iteration number:

Iteration	max|V-Vprev|	# chg actions	V[0]
0	0.33333	N/A	0.000
1	0.10556	1	0.000
2	0.06685	1	0.000
3	0.06351	2	0.000
4	0.04357	1	0.000
5	0.03821	4	0.003
6	0.02857	2	0.008
7	0.02437	1	0.014
8	0.01952	1	0.021
9	0.01624	0	0.028
10	0.01384	0	0.036
11	0.01173	0	0.044
12	0.01047	1	0.052
13	0.00948	0	0.060
14	0.00852	1	0.068
15	0.00782	0	0.075
16	0.00733	0	0.083
17	0.00694	0	0.090
18	0.00656	0	0.096
19	0.00618	0	0.102

The following is a plot of state values as a function of iteration number:

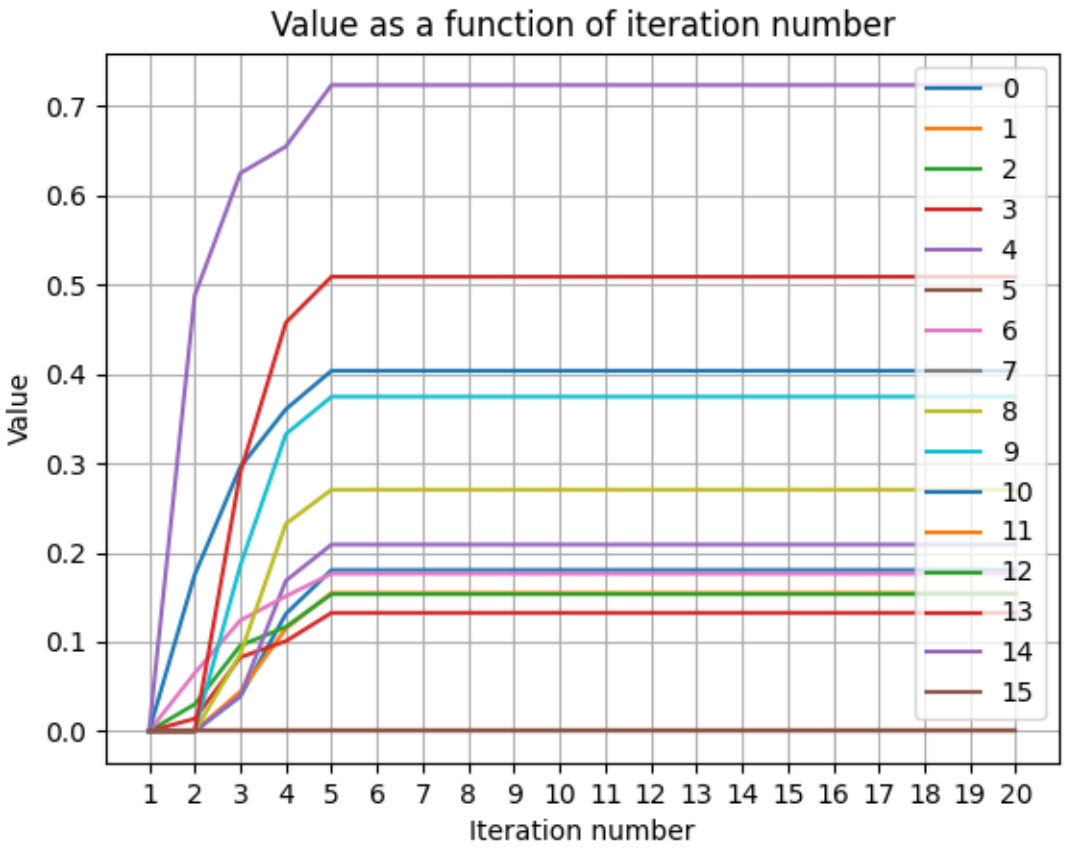

Value as a function of iteration number
Value
Iteration number
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

2.2 Question 2

The following is the progression of value as a function of iteration number:

Iteration	# chg actions	V[0]
0	1	0.00000
1	9	0.00000
2	5	0.03903
3	4	0.13140
4	1	0.18047
5	0	0.18047
6	0	0.18047
7	0	0.18047
8	0	0.18047
9	0	0.18047
10	0	0.18047
11	0	0.18047
12	0	0.18047
13	0	0.18047
14	0	0.18047
15	0	0.18047
16	0	0.18047
17	0	0.18047
18	0	0.18047
19	0	0.18047

The following is a plot of state values as a function of iteration number:


Value as a function of iteration number
Value
Iteration number
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15