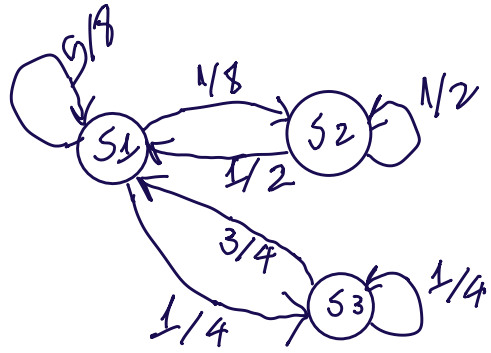
Large and Social Networks 1st set
Ioannis Christofilogiannis 2019030140

P1) a.



$$P_{3\times 3} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 5/8 & 1/8 & 1/4 \\ 1/2 & 1/2 & 0 \\ 3/4 & 0 & 1/4 \end{bmatrix} \end{array}$$

b. It is irreducible because we can visit every state starting from any state and aperiodic because we have self loops ⇒ Therefore it is ergodic.

c. $\cdot \frac{1}{8}\pi_{S1} = \frac{1}{2}\pi_{S2} \Rightarrow \pi_{S1} = 4\pi_{S2}$ (1)

$\cdot \frac{1}{4}\pi_{S1} = \frac{3}{4}\pi_{S3} \Rightarrow \pi_{S1} = 3\pi_{S3}$ (2)

$\cdot \pi_{S1} + \pi_{S2} + \pi_{S3} = 1 \xrightarrow{(1),(2)} \pi_{S1} + \frac{\pi_{S1}}{4} + \frac{\pi_{S1}}{3} = 1$

$\Leftrightarrow \frac{5\pi_{S1}}{4} + \frac{\pi_{S1}}{3} = 1$

$\Leftrightarrow \frac{15\pi_{S1} + 4\pi_{S1}}{12} = 1$

$\Leftrightarrow \boxed{\pi_{S1} = \frac{12}{19}}$

(1) $\Rightarrow \boxed{\pi_{S2} = \frac{3}{19}}$, (2) $\Rightarrow \boxed{\pi_{S3} = \frac{4}{19}}$

So stationary probability is: $\begin{array}{ccc} \pi_{S1} & \pi_{S2} & \pi_{S3} \\ \begin{bmatrix} 12/19 & 3/19 & 4/19 \end{bmatrix} \end{array}$

Because we can calculate stationary probability

the chain is time reversible and local balance holds.

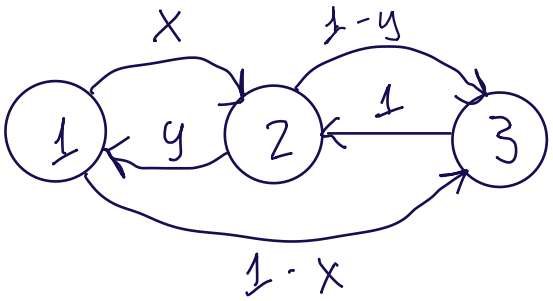
d. From (b) we have the stationary probability for S1 (working state) is 12/19 which translates to: ~63,15% fraction of time.

e. For each state the expected days between successive visits is: $m_{ii} = \frac{1}{\pi_i} \Rightarrow m_{s_2 s_2} = \frac{1}{\pi_{s_2}} = \frac{19}{3} \simeq 6.33$ days.

P3) a.

$$P(1 \to 1) = 0 \quad P(1 \to 2) = x \quad P(1 \to 3) = 1 - x$$
$$P(2 \to 1) = y \quad P(2 \to 2) = 0 \quad P(2 \to 3) = 1 - y$$
$$P(3 \to 1) = 0 \quad P(3 \to 2) = 1 \quad P(3 \to 3) = 0$$

Web page visit probability:



Because $0 < x < y < \frac{1}{2}$

↳ $1 - x > x$, (A)

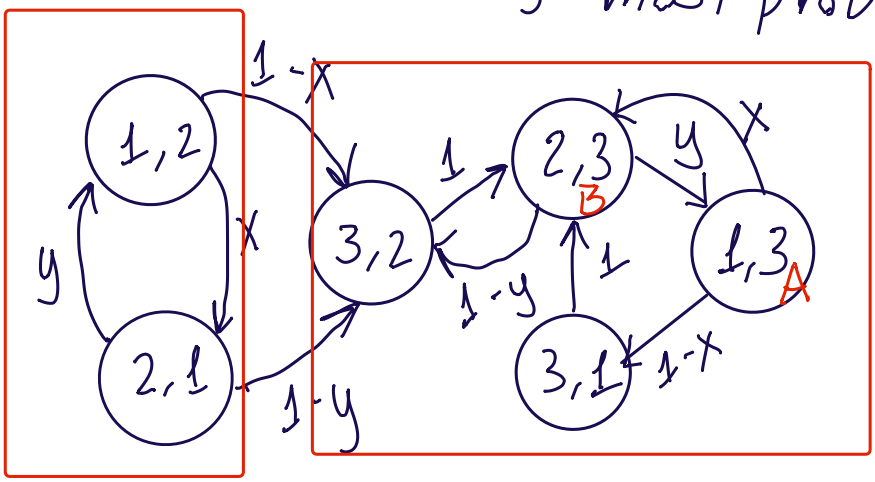
$1 - y > y$, (B)

$1 - x > 1 - y$, (C)

$y > x$ (D)

The model will be: i - current p

j - most probable next p

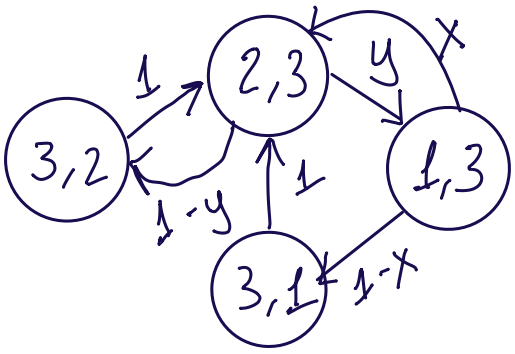


We ignore cases 1,1 2,2 3,3 because $i \neq j$ by definition

This is a complex Markov chain that can be analyzed in two parts, a transient (left) and a recurrent one.

The left part is transient as we expected, because if we start from pages 1 or 2 and visit page 3 the cache will never contain {1,2} again as page 3 is always the most likely (because of (A), (B)).

That means that after n repeats we will stay on the right part of the chain forever.



This part is irreducible and aperiodic.

Global Balance:

$\Pi_{3,1} = (1-x)\Pi_{1,3}$ (1)

$\Pi_{3,2} = (1-y)\Pi_{2,3}$ (2)

$\Pi_{2,3} = \Pi_{3,2} + \Pi_{3,1} + x\Pi_{1,3}$ (3)

$\Pi_{1,3} = y \cdot \Pi_{2,3}$ (4)

And also:

$\Pi_{3,1} + \Pi_{3,2} + \Pi_{2,3} + \Pi_{1,3} = 1$ (5)

So: (1),(4) (2) (4)

$$(1-x)\cdot y\,\Pi_{2,3} + (1-y)\,\Pi_{2,3} + \Pi_{2,3} + y\,\Pi_{2,3} = 1$$

$$\Leftrightarrow \Pi_{2,3}\left((1-x)\cdot y + 1 - \cancel{y} + 1 + \cancel{y}\right) = 1$$

$$\Leftrightarrow \Pi_{2,3} = \frac{1}{y - xy + 2}$$

(4) $$\Rightarrow \Pi_{1,3} = \frac{y}{y-xy+2}$$

(2) $$\Rightarrow \Pi_{3,2} = \frac{1-y}{y-xy+2}$$

(1) $$\Rightarrow \Pi_{3,1} = \frac{(1-x)\cdot y}{y-xy+2} = \frac{y-xy}{y-xy+2}$$

Stationary Probability:

$$\left[\Pi_{1,2}, \Pi_{1,3}, \Pi_{2,1}, \Pi_{2,3}, \Pi_{3,1}, \Pi_{3,2}\right]$$

$$= \left[0, \frac{y}{y-xy+2}, 0, \frac{1}{y-xy+2}, \frac{y-xy}{y-xy+2}, \frac{1-y}{y-xy+2}\right]$$

Using the stationary probability we can calculate the time portions in which the cache will contain the specified pages.

(i) $\{1,2\} \rightarrow \Pi_{1,2} + \Pi_{2,1} = 0 + 0 = 0$

(ii) $\{2,3\} \rightarrow \Pi_{2,3} + \Pi_{3,2} = \frac{1 + 1 - y}{y - xy + 2} = \frac{2 - y}{y - xy + 2}$

(iii) $\{1,3\} \rightarrow \Pi_{1,3} + \Pi_{3,1} = \frac{y + y - xy}{y - xy + 2} = \frac{y(2 - x)}{y - xy + 2}$

(b) Using the known stationary probabilities, we know pages will be requested at a rate

$P(i \rightarrow j)\Pi_s$ for page j

s: current state

$P(i \rightarrow j)$ transition from page i to page j

We only check the recurring part of the chain.

- Page 1:

$r = P(2 \rightarrow 1) \cdot \Pi_{2,3}$ But it is never cached in state 2,3 because 3 is the most likely page. So it does not add to the requested total.

- Page 2:

$r = P(3\to 2)\cdot n_{3,2} + P(3\to 2)\, n_{3,1} + P(1,3)\cdot n_{1,3}$

Only cached on the request from state (3,2)

So: $P(3\to 2)\, n_{3,2} = 1 \cdot \dfrac{1-y}{y-xy+2} = \boxed{\dfrac{1-y}{y-xy+2}}$

- Page 3:

$r = P(1,3)\, n_{1,3} + P(2,3)\cdot n_{2,3}$

As the most likely, it is always cached.

So: $(1-x)\cdot \dfrac{y}{y-xy+2} + (1-y)\cdot \dfrac{1}{y-xy+2} = \dfrac{y-xy+1-y}{y-xy+2}$

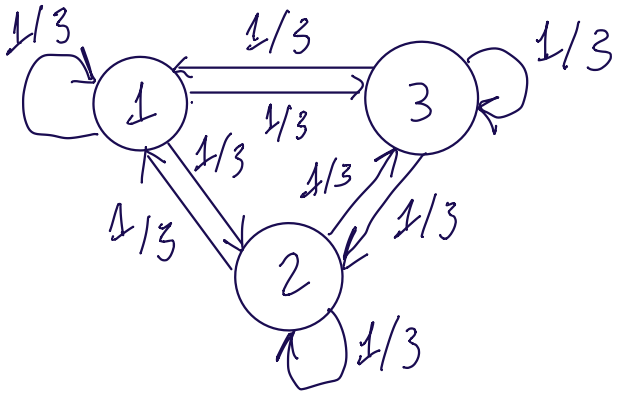
$= \boxed{\dfrac{1-xy}{y-xy+2}}$

So in total:

$$\text{Proportion} = \frac{1-y+1-xy}{y-xy+2} = \frac{-y-xy+2}{y-xy+2}$$

So this * number of requests gives us the number of requests on cached sites.

P4) In the first case, both chains look like this:



This chain is irreducible and aperiodic (self loops) So the nodes will have stationary probability:

$$\left.\begin{array}{l} \pi_1 = \pi_2 \\ \pi_1 = \pi_3 \\ \pi_2 = \pi_3 \end{array}\right\} \quad \pi_1+\pi_2+\pi_3 = 1 \quad 3\pi_1 = 1 \Rightarrow \pi_1 = 1/3$$

$$\begin{bmatrix} \pi_1 & \pi_2 & \pi_3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

We can model each state as:

i = Node A , j = Node B (i,j)

	1,2	1,3	2,1	2,3	3,1	3,2	1,1	2,2	3,3
1,2									
1,3									
2,1									
2,3									
3,1									
3,2									
1,1									
2,2									
3,3									

Q: 1/9

R: 1/9

0: 0

I:

1	0	0
0	1	0
0	0	1

$$I - Q = \begin{bmatrix} 8/9 & -1/9 & & & & \\ -1/9 & 8/9 & -1/9 & & & \\ & -1/9 & 8/9 & -1/9 & & \\ & & -1/9 & 8/9 & -1/9 & \\ & & & -1/9 & 8/9 & -1/9 \\ & & & & -1/9 & 8/9 \end{bmatrix}$$

We need to invert this table, so i will use Python:

```python
if __name__ == "__main__":
    table = [
        [Fraction(8, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9)],
        [Fraction(-1, 9), Fraction(8, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9)],
        [Fraction(-1, 9), Fraction(-1, 9), Fraction(8, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9)],
        [Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(8, 9), Fraction(-1, 9), Fraction(-1, 9)],
        [Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(8, 9), Fraction(-1, 9)],
        [Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(-1, 9), Fraction(8, 9)],
    ]
    inverted_matrix = invert_matrix(table)

    if inverted_matrix is not None:
        print("Inverted Matrix:")
        for row in inverted_matrix:
            print('  '.join(str(cell) for cell in row))
```

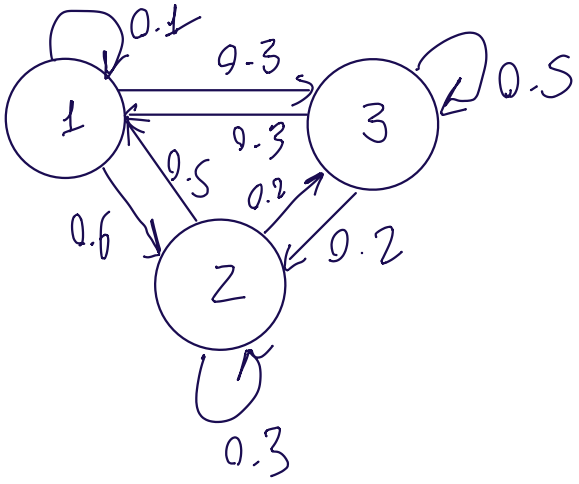
$$N = (I - Q)^{-1} = \begin{bmatrix} 4/3 & 1/3 & 1/3 & 1/3 & 1/3 & 1/3 \\ 1/3 & 4/3 & 1/3 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 4/3 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 4/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 1/3 & 4/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 1/3 & 1/3 & 4/3 \end{bmatrix}$$

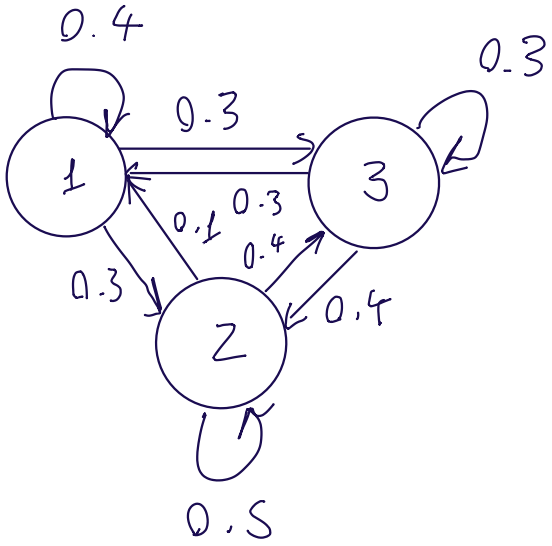
$$t = N \cdot C = \begin{bmatrix} 4/3 & 1/3 & 1/3 & 1/3 & 1/3 \\ 1/3 & 4/3 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 4/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 4/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 1/3 & 4/3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

For every starting state the expected time until absorbtion is 3, so for (1, 3) it is also equal to 3.

(1 ↑ A, 3 ↑ B)

In the second case:





We repeat the process, now doing the calculations and verifying them with Python.

Transient | Recurrent

	1,2	1,3	2,1	2,3	3,1	3,2	1,1	2,2	3,3
1,2	0.05	0.04	0.06	0.24	0.03	0.15	0.01	0.3	0.12
1,3	0.04	0.03	0.18	0.18	0.09	0.12	0.03	0.24	0.09
2,1	0.15	0.15	0.12	0.09	0.08	0.06	0.2	0.09	0.06
2,3	0.2	0.15	0.09	0.09	0.06	0.08	0.15	0.12	0.06
3,1	0.09	0.09	0.08	0.06	0.2	0.15	0.12	0.06	0.15
3,2	0.15	0.12	0.02	0.08	0.05	0.25	0.03	0.1	0.2
1,1	0	...					1	0	0
2,2	⋮						0	1	0
3,3							0	0	1

Q

$$N = (I - Q)^{-1} =$$

```
The N matrix is:
[[0.05 0.04 0.06 0.24 0.03 0.15]
 [0.04 0.03 0.18 0.18 0.09 0.12]
 [0.15 0.15 0.12 0.09 0.08 0.06]
 [0.2  0.15 0.09 0.09 0.06 0.08]
 [0.09 0.09 0.08 0.06 0.2  0.15]
 [0.15 0.12 0.02 0.08 0.05 0.25]]
```

$t = N \cdot c =$

```
t (expected steps) is:
[2.61854837 2.81779399 2.81561045 2.86173717 2.89804762 2.88142813]
```

Node A

Node B

So if we start from (1,3), the expected absorbtion time is $\simeq$ 2.8178 time slots.

Q verification:

```
The Q matrix is:
[[0.05 0.04 0.06 0.24 0.03 0.15]
 [0.04 0.03 0.18 0.18 0.09 0.12]
 [0.15 0.15 0.12 0.09 0.08 0.06]
 [0.2  0.15 0.09 0.09 0.06 0.08]
 [0.09 0.09 0.08 0.06 0.2  0.15]
 [0.15 0.12 0.02 0.08 0.05 0.25]]
```

Python Code:

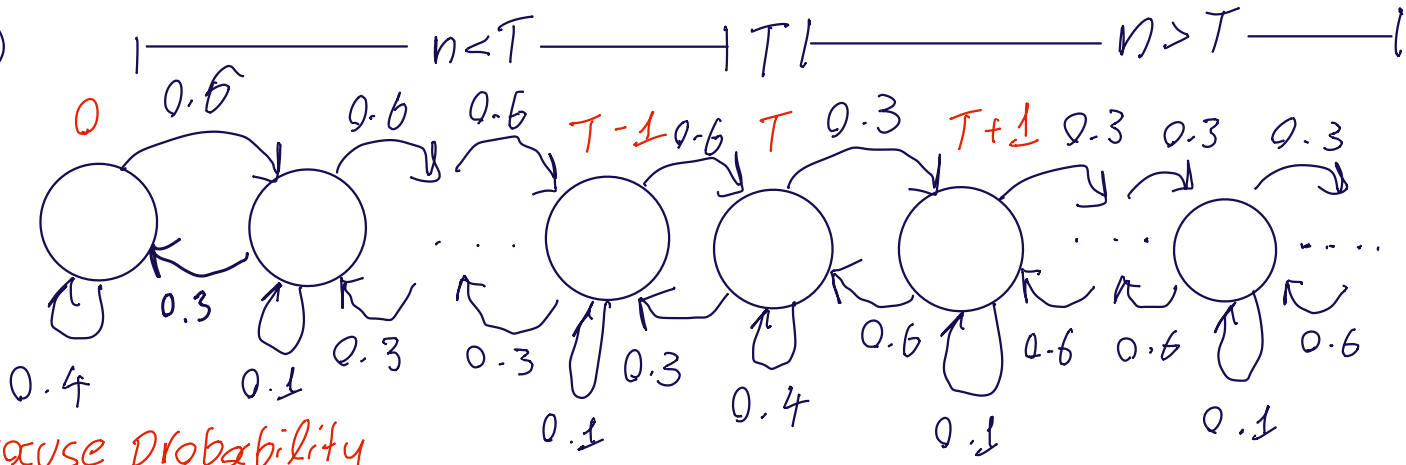
setup experiment, calculate P:

```python
import numpy as np

# Transition matrix for node A
P_A = np.array([
    [0.1, 0.6, 0.3],
    [0.5, 0.3, 0.2],
    [0.3, 0.2, 0.5]
])

# Transition matrix for node B
P_B = np.array([
    [0.4, 0.3, 0.3],
    [0.1, 0.5, 0.4],
    [0.3, 0.4, 0.3]
])

# Combined state space for both nodes, with transient first and recurrent after
states = [(1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2), (1, 1), (2, 2), (3, 3)]

# Initialize the combined transition matrix
P_combined = np.zeros((len(states), len(states)))

# Populate the combined transition matrix
for (i, state_i) in enumerate(states):
    for (j, state_j) in enumerate(states):
        P_combined[i, j] = P_A[state_i[0] - 1, state_j[0] - 1] * P_B[state_i[1] - 1, state_j[1] - 1]
print("The P matrix is:")
print(P_combined)
print("\n")

# To calculate the expected time to absorption, we need to work with the transient states.
transient_indices = [i for i, state in enumerate(states) if state[0] != state[1]]
```

Calculate Q, $N = (I - Q)^{-1}$, t :

```
# Extract the Q matrix, which is the transition probabilities between transient states
Q = P_combined[transient_indices, :][:, transient_indices]
print("The Q matrix is:")
print(Q)
print("\n")

I = np.eye(len(transient_indices))
N = np.linalg.inv(I - Q)

print("The N matrix is:")
print(Q)
print("\n")

# The expected number of steps until absorption for each transient state can be found as the sum of the rows of N
expected_steps = np.transpose(N.sum(axis=1))

print("t (expected steps) is:")
print(expected_steps)
print("\n")

# Get the expected steps from the initial state to absorption
expected_steps_collision = expected_steps[transient_indices.index(states.index((1, 3)))]
print(expected_steps_collision)
```

P.2)



Because probability must sum to 1, no negative num of jobs

I will use the stationary equations, assuming that limiting probabilities exist:

$\Pi_0 = 0.4\Pi_0 + 0.3\Pi_1 \Rightarrow 0.6\Pi_0 = 0.3\Pi_1 \Rightarrow \Pi_1 = 2\Pi_0$

$\Pi_1 = 0.6\Pi_0 + 0.3\Pi_2 + 0.1\Pi_1 \Rightarrow 1.2\Pi_0 = 0.3\Pi_2 \Rightarrow \Pi_2 = 4\Pi_0$

$\Pi_2 = 0.6\Pi_1 + 0.3\Pi_3 + 0.1\Pi_2 \Rightarrow 4\Pi_0 - 1.2\Pi_0 - 0.4\Pi_0 = 0.3\Pi_3$

$\Leftrightarrow \Pi_3 = 8\Pi_0$

(A)

$\Pi_{T-1} = 0.6\Pi_{T-2} + 0.3\Pi_T + 0.1\Pi_{T-1}$ (1)

$\Pi_T = 0.6\Pi_{T-1} + 0.4\Pi_T + 0.6\Pi_{T+1}$ (2)

$\Pi_{T+1} = 0.3\Pi_T + 0.6\Pi_{T+2} + 0.1\Pi_{T+1}$ (3)

$\sum_{i=0}^{\infty} \Pi_i = 1$

(*) Until T, $\Pi_n = 2^n \Pi_0$ as shown $(n < T)$

(1) $\Rightarrow \Pi_{T-1} = 2^{T-1}\Pi_0 = 0.6 \cdot 2^{T-2}\Pi_0 + 0.3\Pi_T + 0.1 \cdot 2^{T-1}\Pi_0$

$\Leftrightarrow 0.3\Pi_T = 2^{T-1}\Pi_0 - 0.6 \cdot 2^{T-2}\Pi_0 - 0.1 \cdot 2^{T-1}\Pi_0$

$\Leftrightarrow 0.3 n_T = 2^{T-1} n_0 (1 - 0.3 - 0.1)$

$\Leftrightarrow n_T = 2^{T-1} \cdot n_0 \cdot \frac{0.6}{0.3}$

$\Leftrightarrow n_T = 2^T n_0$, so the formula still holds for $n = T$

now for $n > T$:

(2) $2^T \cdot n_0 = 0.6 \cdot 2^{T-1} \cdot n_0 + 0.4 \cdot 2^T \cdot n_0 + 0.6 n^{T+1}$

$\Leftrightarrow n^{T+1} = \frac{2^T - 0.6 \cdot 2^{T-1} - 0.4 \cdot 2^T}{0.6} n_0$

$\Leftrightarrow n^{T+1} = \frac{2^T (1 - 0.6 \cdot \frac{1}{2} - 0.4)}{0.6} \cdot n_0$

$\Leftrightarrow n^{T+1} = 2^T \frac{0.3}{0.6} \cdot n_0 \Leftrightarrow \boxed{n_{T+1} = 2^{T-1} n_0}$

(3)

$0.9 \cdot 2^{T-1} n_0 = 0.3 \cdot 2^T n_0 + 0.6 n_{T+2}$

$\Leftrightarrow 3 \cdot 2^{T-1} n_0 = 2^T n_0 + 2 n_{T+2}$

$\Leftrightarrow n_{T+2} = \frac{3 \cdot 2^{T-1} n_0 - 2^T n_0}{2}$

$$\Leftrightarrow \pi_{T+2} = 3 \cdot 2^{T-2}\pi_0 - 2^{T-1}\pi_0$$

$$\Leftrightarrow \pi_{T+2} = 2^{T-1}\cdot\pi_0\left(3\cdot\frac{1}{2} - 1\right)$$

$$\Leftrightarrow \boxed{\pi_{T+2} = 2^{T-2}\cdot\pi_0}$$

pattern proved for $n > T \Rightarrow \pi_{T+s} = 2^{T-s}\cdot\pi_0$ where $s = T-n$

So the sum of probabilities:

$$\sum_{i=0}^{T} 2^i \pi_0 + \sum_{i=T+1}^{\infty} 2^{T+(T-i)}\pi_0 = 1$$

$$\Leftrightarrow \sum_{i=0}^{T} 2^i \pi_0 + \sum_{i=T+1}^{\infty} 2^{2T-i}\pi_0 = 1$$

series decreases as time increases

$$\Leftrightarrow \pi_0\left(\frac{1-2^{T+1}}{1-2}\right) + \sum_{i=T+1}^{\infty} 2^{2T-i}\pi_0 = 1$$

$$\Leftrightarrow \pi_0\left(2^{T+1}-1\right) + \sum_{j=0}^{\infty} \pi_0 2^{T-1-j} = 1 \quad \left(j = i-(T+1)\right)$$

Infinite geometric series with $|r| = \frac{1}{2} < 1$

$$\Leftrightarrow n_0 (2^{T+1} - 1) + n_0 2^{T-1} \cdot \frac{1}{1 - 1/2} = 1$$

$$\Leftrightarrow n_0 (2^{T+1} - 1) + n_0 \cdot 2^T = 1$$

$$\Leftrightarrow n_0 (2 \cdot 2^T - 1 + 2^T) = 1$$

$$\Leftrightarrow n_0 (3 \cdot 2^T - 1) = 1$$

$$\Leftrightarrow \boxed{n_0 = \frac{1}{3 \cdot 2^T - 1}}$$

And so:

- For $n = 0$: $n_{n=0} = \frac{1}{3 \cdot 2^T - 1}$
- for $n < T+1$: $n_n = 2^n \cdot n_0 = 2^n \cdot \frac{1}{3 \cdot 2^T - 1}$
- for $n \geq T+1$: $n_n = 2^{T-(n-T)} \cdot n_0 = 2^{2T-n} \cdot n_0$

$(s = n - T)$

2) $E[N] = \sum_{i=0}^{+\infty} i \cdot \pi_i = \sum_{i=0}^{T} i 2^i \cdot n_0 + \sum_{i=T+1}^{+\infty} i \cdot 2^{2T-i} \cdot n_0$

$= (2^{T+2}(T+2) - 2) n_0$

$\sum_{i=T+1}^{\infty} i n_0 2^{2T-i} = \sum_{j=0}^{\infty} (j+T+1) n_0 2^{2T-(j+T+1)}$

exponent $= 2^{T-1-j}$

$\Rightarrow n_0 \sum_{j=0}^{\infty} (j+T+1) 2^{T-1-j}$

$= n_0 2^{T-1} \left(\sum_{j=0}^{\infty} j 2^{-j} + (T+1) \underbrace{\sum_{j=0}^{\infty} 2^{-j}}_{=2(T+1)} \right)$

$S = \sum_{j=0}^{\infty} x^j = \frac{1}{1-x} \rightarrow \frac{d}{dx}\left(\frac{x}{1-x}\right) = \frac{1}{(1-x)^2}$

So $\sum_{j=0}^{\infty} j \left(\frac{1}{2}\right)^j = 2^2 = 4$

Total

$n_0 2^{T-1} (4 + 2T + 2)$

$= n_0 2^{T-1} (2T+6)$

Meaning that:

$$E[N] = \left(2^{T+2}(T+2) - 2\right) \cdot n_0 + 2^{T-1}(2T+6) \cdot n_0$$

$$= n_0 \left[2^{T+2} \cdot T + 2^{T+3} - 2 + 2^{T-1}(2T+6)\right]$$

$$= \frac{1}{3 \cdot 2^T - 1} \left[2^{T+3} + T \cdot 2^{T+2} + T \cdot 2^T + 6 \cdot 2^{T-1} - 2\right]$$

$$= \frac{2}{3 \cdot 2^T - 1} \left[2^{T+2} + T \cdot 2^{T+1} + T \cdot 2^{T-1} + 6 \cdot 2^{T-2} - 1\right]$$

3. We expect that as T increases the mean number of jobs will increase, because more 'time' will be spent on $n < T$ case which has a generally increasing num of jobs because of probabilities.

If we had a small T, $E[T]$ would slow down substantially after T converging to a value close to T because the chain is 'promoting' states close to T with the probabilities.

If $T \to \infty$, I expect $E[N] \to \infty$.