A graph Patrol Problem with a random attacker and an observable patroller

Thomas Lowbridge
School of Mathematical Sciences
University of Nottingham

February 22, 2018

Contents

1 Introduction to a random attacker patroller game with observation **1**

2 Single node problem **3**

3 Deterministic Attack time **3**
3.1 Correction to approach for Deterministic 6
3.2 Index Heuristic 9
3.3 Lower bound 9

4 Bernoulli Attack time **10**
4.1 Correction to approach for bernoulli 12

5 Ideas for future **i**

Appendices **i**

A Observations are always zero **i**

B Bernoulli Attack time with equal floors **ii**

1 Introduction to a random attacker patroller game with observation

The model has a graph, $Q = (N, E)$, with a set of nodes labeled 1 to n, $N = \{1, ..., n\}$, and a set of edges linking these nodes. The adjacency matrix $a = (a_{i,j})_{i,j \in N}$, has $a_{i,j} = 1$ if i and j are adjacent and $a_{i,j} = 0$ if they are not adjacent. By definition we will use $a_{i,i} = 1 \quad \forall i \in N$.

An attacker has some attack time for node i, called X_i and chooses to attack node i with some probability, p_i. The attackers arrive according to some Poisson process with rate Λ, so by Poisson thinning they arrive at node i according to a Poisson process with rate $\lambda_i = \Lambda p_i$.

The patroller, uses some walk (with possible waiting) to patrol the graph.We assume that a patrollers walk is able to capture all attacks that have already begun, but not completed. But unlike the 'normal' setting the past unit time, the attackers do not start their attacks and instead will wait for the patroller to leave. Each missed attack at node i inccures a cost of c_i to the patroller.

We can formulate the state space, as the delineation of separate nodes. $\Omega = \{(\boldsymbol{s}, \boldsymbol{v}) = \quad | \quad s_i = 1, 2, ..., v_i = 0, 1, 2, ... \quad \forall i \in N\}$. Where $\boldsymbol{s} = (s_1, ..., s_n)$ has each s_i represent the number of time periods since the last visit for that node i and $\boldsymbol{v} = (v_1, ..., v_n)$ has each v_i represent the number of attackers present in the last time period when the node i was last visited (i.e The number of attackers known to be beginning their attack s_i time ago at node i).

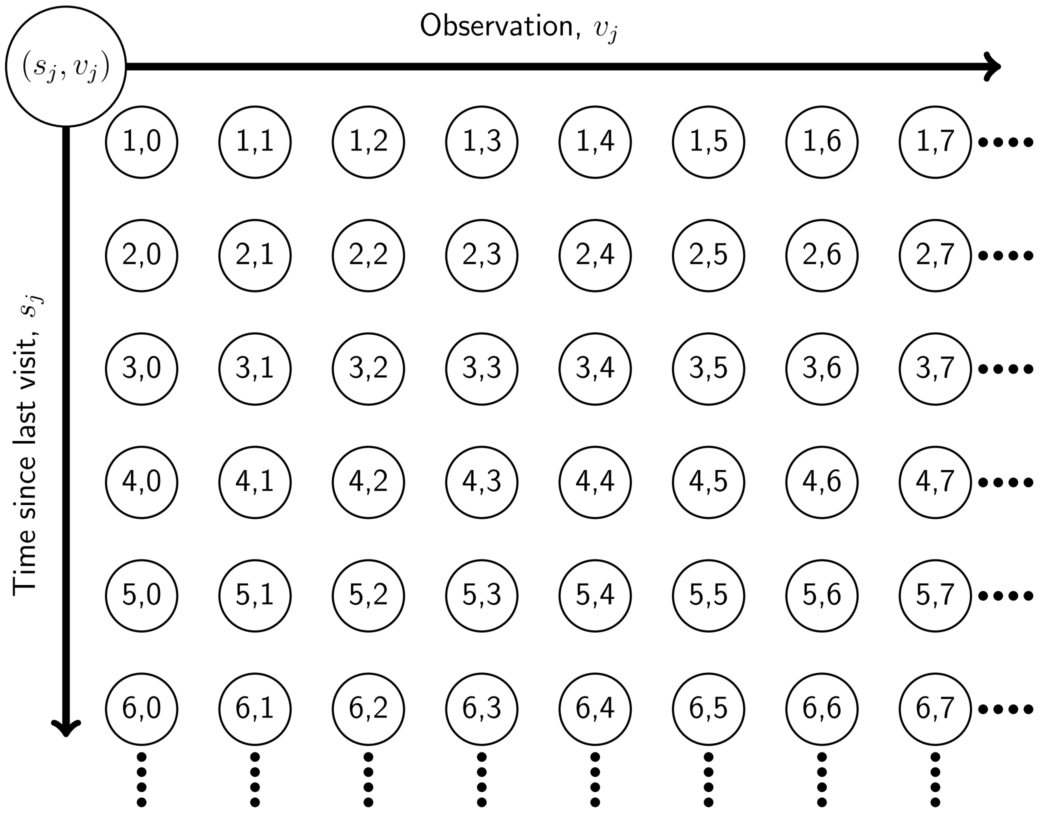

Figure 1.1: State space diagram

The s_i increment by 1 if the node is not visited upon each action, or if the node is visited reset to $s_i = 1$. The v_i do not change for nodes not visited, when a node is visited, the v_i 'reset' according to the Poisson distribution $Po(\lambda_i \times 1) = Po(\lambda)$. Due to $s_i = 1$ if and only if the patroller is currently at this node, we will use $l(\boldsymbol{s}) = \arg\min_{i \in N} s_i$ to represent the current node.

As the future of the process is independent of its past, the process can be formulated as a Markov decision process(MDP), where at the end of the period, the patroller chooses which adjacent node to visit. Thus the action space is $\mathcal{A} = \{j \mid a_{l(\boldsymbol{s}),j} = 1\}$, with a deterministic, stationary policy, $\pi : \Omega \to \mathcal{A}$.

The transitions of the MDP aren't entirely deterministic, $\boldsymbol{s}$ is purely deterministic, but $\boldsymbol{v}$ is partially probabilistic. In state $(\boldsymbol{s}, \boldsymbol{v})$ with the decision to visit node $i \in \mathcal{A}$, then the state will transition to $(\widetilde{\boldsymbol{s}}, \widetilde{\boldsymbol{v}})$ where $\widetilde{s}_j = s_j + 1$ if $j \neq i$ and $\widetilde{s}_j = 1$ if $j = i$ and $\widetilde{v}_j = o_j$ if $j \neq i$ and $v_j \sim Po(\lambda)$ if $j = i$.

To write down the cost function, which is dependent on the state $(\boldsymbol{s}, \boldsymbol{v})$ and the action to visit node i chosen, we will look at the expected cost of incurred at all nodes and sum these costs for the next time period.

$$C_j(\boldsymbol{s},\boldsymbol{v},i) = \begin{cases} c_j\lambda_j \int_0^{s_j} P(t-1 < X_j \leq t)dt + v_j P(0 < X_j \leq s_j) \text{ for } i \neq j \\ c_j\lambda_j \int_0^{s_j-1} P(t-1 < X_j \leq t)dt + v_j P(0 < X_j \leq s_j) \text{ for } i = j \end{cases}$$

$$= \begin{cases} c_j\lambda_j \int_{s_j-1}^{s_j} P(X_j \leq t)dt + v_j P(X_j \leq s_j) \text{ for } i \neq j \\ c_j\lambda_j \int_{s_j-2}^{s_j-1} P(X_j \leq t)dt + v_j P(X_j \leq s_j) \text{ for } i = j \end{cases} \quad (1)$$

With $C(\boldsymbol{s},\boldsymbol{v},i) = \sum_{j=1}^{n} C_j(\boldsymbol{s},\boldsymbol{v},i)$ being the cost function for the MDP.

We will now make the assumptions that X_j is bounded by B_j and that instead of using $Po(\lambda)$ for the observation transition and placing a bound on this Poisson distribution, named b_j, so we are now drawing from a truncated Poisson distribtion, henceforth called $TPo(\lambda, b_j)$. Then we can immediately say that the $o_j \leq b_j$ state is finite and the state s_j has the same cost function for $s_j \geq B_j + 2$ and hence we will restrict our space to this. So our modified transition is $\tilde{s_j} = \min(s_j + 1, B_j + 2)$ if $j \neq i$ and $\tilde{s_j} = 1$ if $i = j$. $\tilde{v_j} = v_j$ if $i \neq j$ and $v_j \sim TPo(\lambda, b_j)$ if $i = j$.

The truncated Poisson distribution, $TPo(\lambda, b)$ acts like normal Poisson for any value not equal to b, with the tails probability stored at b. That is $P(TPo(\lambda,b) = i) = \begin{cases} P(Po(\lambda) = i) \text{ If } i \neq b \\ P(Po(\lambda) \geq b) \text{ If } i = b \\ 0 \text{ Otherwise} \end{cases}$

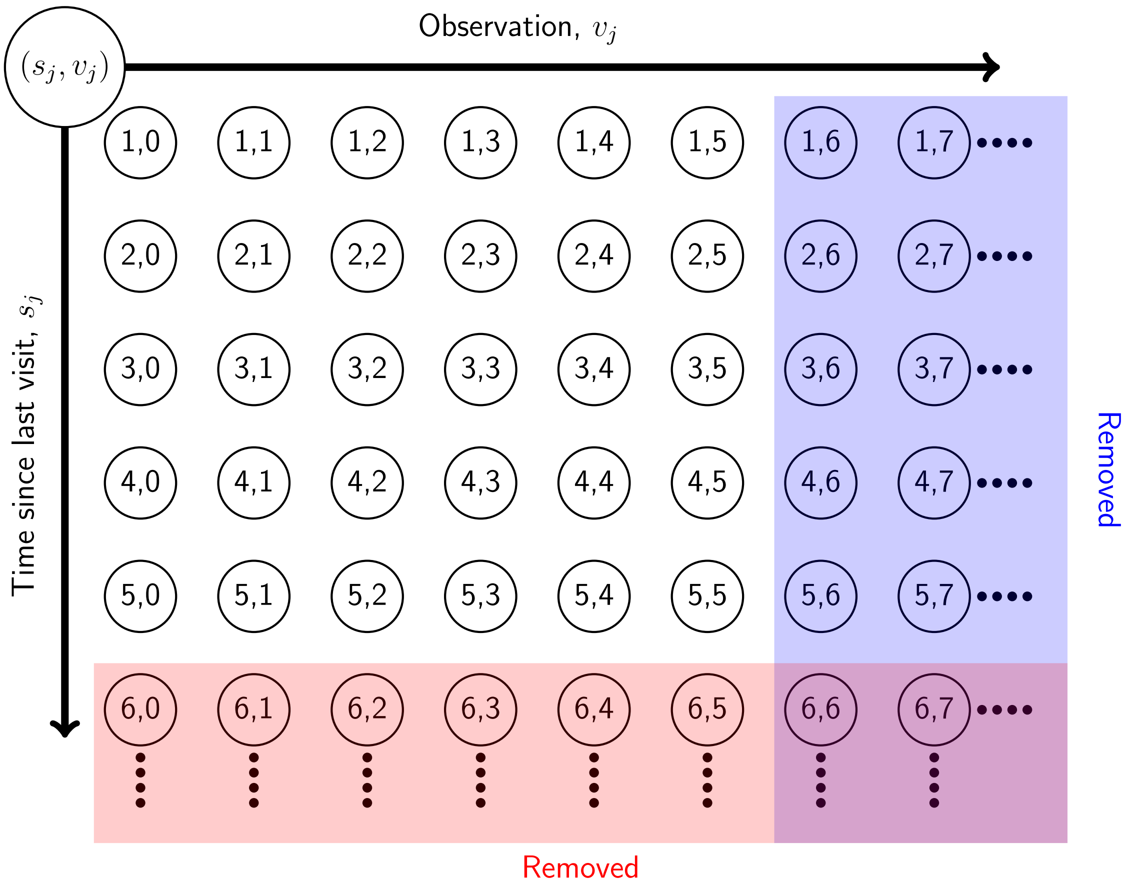

Figure 1.2: State space diagram, with $b_j = 5$ and $B_j = 4$

Further reduction is possible as if $X_j \leq B_j$ then any observations v_j which started s_j time units ago is bound to have finished if $s_j \geq B_j + 1$. So our new state space is further reduced to having only $(\lfloor B_j \rfloor + 1, 0)$ when $s_j = \lfloor B_j \rfloor + 1$.

So $\Omega = \{(\boldsymbol{s},\boldsymbol{v}) | s_i = 1, 2, .., \lfloor B_i \rfloor + 1, v_i = 1, ..., b_i \, \forall i \in N\} \cup \{(\lfloor B_j \rfloor + 2, 0)\}$.

With further modified transitions that if $s_j = \lfloor B_j \rfloor + 1$ then $\tilde{v_j} = 0$ for $i \neq j$.

Now our state space and action space are finite we need only consider deterministic, stationary policies. Applying such a policy generates a sequence of states under a given policy π, namely $\{\psi_\pi^k(\boldsymbol{s}_0, \boldsymbol{o}_0), k = 0, 1, 2, ...\}$. However we are not guaranteed to every have a regenerating process when the same node is visited due to the unpredictable nature of $o_i \sim TPo(\lambda, b_i)$. Unless

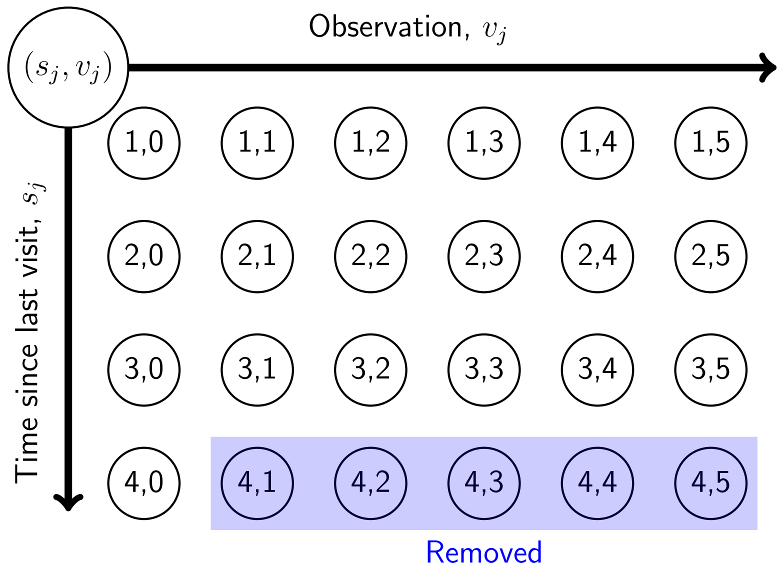

Figure 1.3: State space diagram, with $b_j = 5$ and $B_j = 4$ and further reduction

$b_i = 0 \quad \forall i \in N$ then we have removed the probabilistic nature of v_i's transition. We will not focus on the special case of $b_i = 0 \quad \forall i \in N$ but it is shown how to develop a index for the single node problem in Appendix A.

Note. There may be a problem with having a non-deterministic attack time, as this may cause the observable attacker behaviour to no longer take the form of $(\boldsymbol{s}, \boldsymbol{v})$ but the patroller may have other information such as visiting early may mean if they caught less than expected then some in the observed section exist, so a full history could be recorded. However we will not deal with this and assume once a node is visited its last observation is lost and the system is rest???

2 Single node problem

Focusing on the problem of a single nodes and stripping off the index, i, for the nodes. This problem has a visiting cost, $\omega > 0$ and we are looking to minimize the long run cost of the system.

3 Deterministic Attack time

Consider the case where $X = x$, where x is a constant (So $B = x$). Then we can further reduce the state space, as we choosing to visit later rather than earlier (as long as its not too late) allows us to possibly catch more (as we know when the attacks can start to finish). So we limit the state space with non-zero observed attackers to only have $s_j = \lfloor B \rfloor + 1$, as visiting at then gets any attacks caught when visiting at any $s_j < \lfloor B \rfloor + 1$.

So in the deterministic case $\Omega = \{(\lfloor B \rfloor + 1, v) \,|\, v = 0, 1, ..., b\} \cup \{(\lfloor B \rfloor + 2, 0)\}$.

Suppose now we are in the state $(\lfloor B \rfloor + 1, v)$ for some $v = 1, 2, ..., b$ then our decision is either to

- Visit now
- Visit at the next time step
- Visit k time steps later, $k \geq 2$

Note. The logic of these is simply that the patroller can decide when to visit, but if they reach $(\lfloor B \rfloor + 2, 0)$ and decide to wait for one time period, they will not transition and therefore the same decide will be made again.

Now we write down the long-run average costs of following such a strategy

- Visit now:

$$\frac{c\lambda \int_0^{\lfloor B \rfloor} P(X \leq t)dt + \omega}{\lfloor B \rfloor + 1} = \frac{\omega}{\lfloor B \rfloor + 1} \tag{2}$$

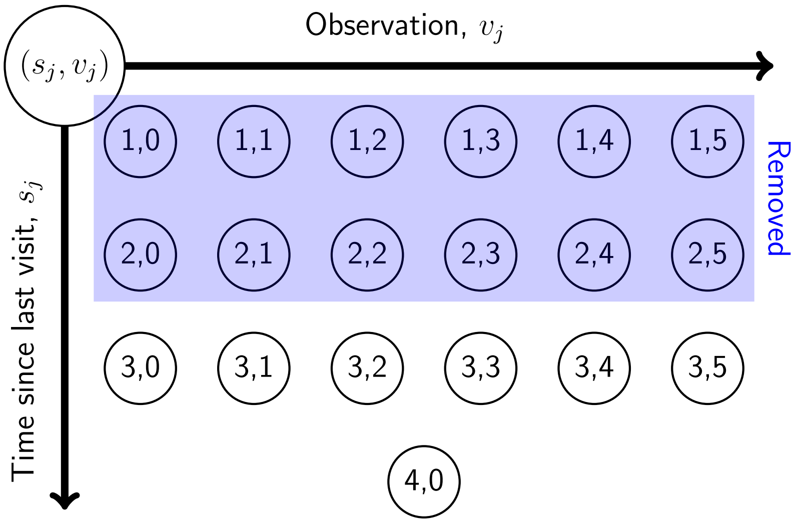

Figure 3.1: Deterministic state space diagram, with $b_j = 5$ and $B_j = 4$

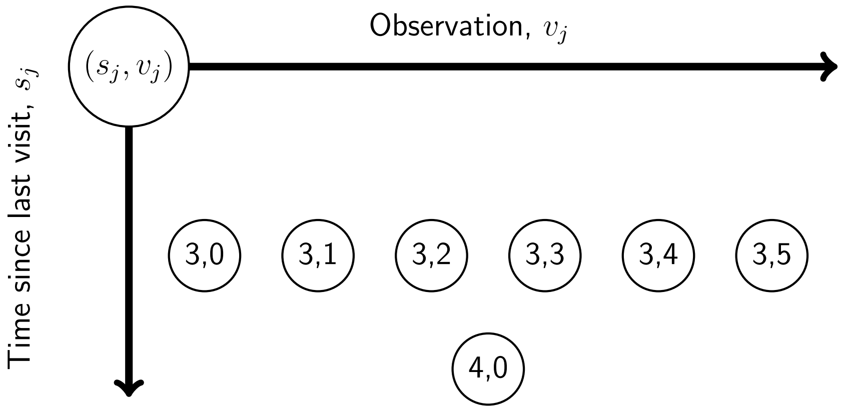

Figure 3.2: Final deterministic, with $b_j = 5$ and $B_j = 4$

- Visit at the next time step:

$$\frac{vc + c\lambda \int_0^{\lfloor B \rfloor + 1} P(X \leq t)dt + \omega}{\lfloor B \rfloor + 2} = \frac{vc + c\lambda(\lfloor B \rfloor + 1 - B) + \omega}{\lfloor B \rfloor + 2} \tag{3}$$

- Visit k time steps later, $k \geq 2$:

$$\frac{vc + c\lambda \int_0^{\lfloor B \rfloor + k} P(X \leq t)dt + \omega}{\lfloor B \rfloor + k + 1} = \frac{vc + c\lambda(\lfloor B \rfloor + k - B) + \omega}{\lfloor B \rfloor + k + 1} \tag{4}$$

Our first decision is if we should Visit now, this depends on if Equation 2 is less than equation 3 and 4.

Well for this to be true we get that $\omega < c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v)$ and $\omega < c(\lfloor B \rfloor + 1)(\lambda(\frac{\lfloor B \rfloor - B}{k} + 1) + \frac{v}{k})$. Hence as the second inequality ($\forall k = 2, 3, ...$) is guaranteed by the first inequality we get.

Visit now if:

$$\omega < c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v) \tag{5}$$

Similarly for visiting at the next time step, we get $\omega > c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v)$ and $\omega < c(\lambda(B + 1) - \frac{v}{k-1})$ for all $k = 2, ...$ so $\omega < c(\lambda(B + 1) - v)$.

Visit next time step if:

$$c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v) < \omega < c(\lambda(B + 1) - v) \tag{6}$$

Note. The question is whether it is possible to get an empty region here. Well if $c(\lfloor B \rfloor+1)(\lambda(\lfloor B \rfloor-B+1)+v) > c(\lambda(B+1)-v)$ is empty it will enforce the never visit decision immediately

We will never visit, if we at a point were we will choose $k + 1$ over k for all $k = 2, ...$

This happens when $\omega > c(\lambda(B + 1) - v)$.

Never visit if:

$$\omega > c(\lambda(B + 1) - v) \tag{7}$$

So to conclude, we either end up with 3 or 2 regions

- If $c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v) \leq c(\lambda(B + 1) - v)$
  - Visit immediately if $\omega \leq c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v)$
  - Visit next time step if $c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v) \leq \omega \leq c(\lambda(B + 1) - v)$
  - Never Visit if $\omega \geq c(\lambda(B + 1) - v)$
- If $c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v) > c(\lambda(B + 1) - v)$
  - Visit immediately if $\omega \leq c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1) + v)$
  - Never visit if $\omega \geq c(\lfloor B \rfloor + 1)(\lambda(\lfloor B \rfloor - B + 1 + v)$

Note. If $v = 0$ then we never fall into the second region as $(\lfloor B \rfloor + 1)(\lfloor B \rfloor - B + 1) = (\lfloor B \rfloor + 1)(1 - R) \leq \lfloor B \rfloor + 1 \leq B + 1$ where the remainder upon flooring is defined by $R = B - \lfloor B \rfloor$ and $0 \leq R < 1$.

Moreover the region condition is reduced to $v \leq \lambda R$ or $v > \lambda R$.

Also of interest is if $X = B$ has B very close to an interger, eg.$B = 3.01$ then the condition to fall into the second region is that $v < \frac{\lambda}{100}$, requiring a very high rate to fall into the category of never deciding to wait one time step.

Remark. We remark that $\omega > c(\lambda(B+1)-v) = \omega' + c(\lambda-v)$. Where ω' is the old never visiting cost (under the non-observable attackers), so as $o \leq R\lambda$ $(0 \leq R < 1)$ we know that $\lambda > v$, so the visiting cost must be higher than before to consider not visiting. This is as we would expect.

If $v > \lambda$ then in the second we still get a higher critical visiting cost to never visit (as $c\lambda(\lfloor B \rfloor + 1)(R + 1) > c\lambda B$).

Example 3.1. Suppose $X = 2.1$ and $b = 5$, then our state space is reduced to $\Omega = \{(3, v)\,|v = 0, 1, ..., 5\} \cup \{(4, 0)\}$ and we make the decision at state $(3, v)$ first based on the calculation of $c(2 + 1)(\lambda(2 - 2.1 + 1) + v) = c(2.7\lambda + 3v)$ versus $c(\lambda(2.1 + 1) - o) = c(3.1\lambda - o)$ so depends if $0.1\lambda \geq v$ or $0.1\lambda > v$.

Suppose now that we have $v = 1$ and $\lambda = 20$ then we fall into the first case and we will make the decision based on the cost, ω:

- Visit immediately if $\omega \leq c(2.7 \times 20 + 3 \times 1) = 57c$
- Wait then Visit if $57c < \omega < c(3.1 \times 20 - 1) = 61c$
- Never Visit if $\omega \geq 61c$

However if instead $v = 4$ while $\lambda = 20$ then we fall into the first case and will make the decision based on the cost, ω:

- Visit immediately if $\omega \leq c(2.7 \times 20 + 3 \times 4) = 66c$
- Never Visit if $\omega \geq 66c$

Note. $69c > c(3.1 \times 20 - 4) = 56c$ so we are never going to wait then visit (and in fact we are more incentivised to visit earlier)

3.1 Correction to approach for Deterministic

We will first have a concrete argument as to why the decision is always to wait when in a state (s, v) with $s < \lfloor B \rfloor + 1$. From this position consider the policy π_k which waits k time periods and then renews and follows the optimal policy, σ, with $k = 0, ..., \lfloor B \rfloor + 1 - s$.

Using such a policy will get us that

$$V_n^{\pi_k}(x, v) = \omega + E[V_{n-k-1}^{\sigma}(\theta)] \tag{8}$$

where θ is the state upon renewal (i.e it is the state $(1, V) \sim (1, TPo(\lambda))$.

Now we will pick policy π_{k+1} over π_k (or be indifferent) if

$$\begin{aligned}
&\lim_{n\to\infty} V_n^{\pi_k}(x, v) - V_n^{\pi_{k+1}}(x, v) \geq 0 \\
\iff &\lim_{n\to\infty} E[V_{n-k}^{\sigma}(\theta) - V_{n-k-1}^{\sigma}(\theta)] \geq 0 \\
\iff &g \geq 0
\end{aligned}$$

Where g is the average long-run costs of the system, coming from $V_n(x, v) = ng + \phi(x, v)$ for large n, and $\phi(x, v)$ is some initial bias for not starting in equilibrium from the initial state (x, v).

Now as the Dynamic Programming only has positive costs, it is impossible for $g < 0$. So we have that $g \geq 0$, so it is better to pick policy π_{k+1} over π_k, as this argument holds for all $k = 0, ..., \lfloor B \rfloor + 1 - s$ (and all v) it is best to wait till $(\lfloor B \rfloor + 1, v)$ before making a decision.

This formally shows the removal of the prior states.

Now we will skip to the state $(\lfloor B \rfloor + 2, 0)$ and suggest again a policy π_k which waits k time periods before renewing and then follows some optimal policy, σ.

Using such a policy will get us

$$V_n^{\pi_k}(x, v) = \omega + c\lambda k + E[V_{n-k-1}^{\sigma}(\theta)] \tag{9}$$

And again we will pick a policy π_{k+1} over π_k (or be indifferent) if

$$\begin{aligned}
&\lim_{n\to\infty} V_n^{\pi_k}(\lfloor B \rfloor + 2, 0) - V_n^{\pi_{k+1}}(\lfloor B \rfloor + 2, 0) \geq 0 \\
\iff &\lim_{n\to\infty} -c\lambda + E[V_{n-k}^{\sigma}(\theta) - V_{n-k-1}^{\sigma}(\theta)] \geq 0 \\
\iff &g \geq c\lambda
\end{aligned}$$

So hence if $g \geq c\lambda$ we will wait forever, as this has no dependence on k.

We will now argue that $g_{max} = c\lambda$, that is at worst the long-run average cost is $c\lambda$. This can be seen by the strategy π_{neg}, a strategy which never renews no matter what state we are in. I.e the patroller neglects the node. We will have that, for large n

$$V_n^{\pi_{neg}}(x, v) = nc\lambda + \phi(x, v)$$

We can observe this by looking at the actual decay of the process, under a policy that never renews

$$V_n^{\pi_{neg}}(x, v) = V_{n-(\lfloor B \rfloor + 2 - x)}^{\pi_{neg}} + \phi(x, v) \qquad = (n - (\lfloor B \rfloor + 2 - x))c\lambda + phi(x, v)$$

Where $\phi(x, v) = \underbrace{vc\lambda}_{\text{Observed finishing}} + \underbrace{(1 - R)c\lambda}_{\text{arrivals that finish}}$ $(R = B - \lfloor B \rfloor)$ is the 'transfer'/bias of moving from the initial state to $(\lfloor B \rfloor = 2, 1)$.

Hence for any service cost, ω we can achieve a $g = c\lambda$. So we will always be in the case of $g \leq c\lambda$ and hence we will renew if we hit the state $(\lfloor B \rfloor + 2, 0)$ and hence for a state $(\lfloor B \rfloor + 1, v)$ we have two types of policy, π_0 a policy which renews now and then follows some optimal policy σ or policy, π_1 which waits one period until $(\lfloor B \rfloor + 2, 0)$ and then renews and follows some optimal policy σ.

So we will choice to renew now over waiting if

$$\begin{aligned}
&\lim_{n\to\infty} V_n^{\pi_1}(\lfloor B \rfloor + 1, v) - V_n^{\pi_0}(\lfloor B \rfloor + 1, v) \geq 0 \\
\iff &\lim_{n\to\infty} (\lfloor B \rfloor - B + 1)c\lambda + vc + E[V_{n-1}^{\sigma}(\lfloor B + 2 \rfloor, 0)] - (\omega + E[V_{n-1}^{\sigma}(\theta)]) \geq 0 \\
\iff &\lim_{n\to\infty} (\lfloor B \rfloor - B + 1)c\lambda + vc + E[\omega + V_{n-2}^{\sigma}(\theta)] - \omega - E[V_{n-1}^{\sigma}(\theta)] \geq 0 \\
\iff &\lim_{n\to\infty} (\lambda(\lfloor B \rfloor - B + 1) + v)c + E[V_{n-2}^{\sigma}(\theta) - V_{n-1}^{\sigma}(\theta)] \geq 0 \\
\iff &(1 - R + v)c\lambda - g \geq 0 \\
\iff &g \leq c(\lambda(1 - R) + v)
\end{aligned}$$

So we renew now if $g \leq (\lambda(1-R)+v)c$, as we are guaranteed that $g \leq c\lambda$ we are clearly in this region unless $\lambda R - v \geq 0 \iff v \leq \lambda R$, so if we will renew in v we definitely renew in $v + 1, v + 2, ..., b$. Their is some critical value, $v_{\text{crit}} = \left\lceil \frac{g}{c} + \lambda(R - 1) \right\rceil$ such that for $v \geq v_{\text{crit}}$ we renew immediately and for $v < v_{\text{crit}}$ we wait one time period and renew.

We have two special cases for g, that is when we always renew immediately or always wait then renew, these are $g \leq c\lambda(1-R)$ (i.e we renew for all v if we renew for $v = 0$) and $g \geq c(\lambda(1 - R) + b)$ (i.e we always wait then renew for all v if we wait for $v = b$)

However between these regions, we have a divide of the nodes into a set we renew immediately and a set we wait then renew.

However we are assuming we are in the case of $g \leq c\lambda$ for so the example 3.3 we cannot have a situation as $g_{\max} = 1$. Really in this case we would just follow the neglecting strategy, π_{neg}.

We really need to split the region for g (which is bounded above by $c\lambda$) into different $v = v_{\text{crit}}$.

Say now we have that $b > \lambda R$ then we have some $v_{max} = \max\{v \in \{0, 1, ..., b + 1\} \mid v \leq \lambda R\}$, some critical maximum (as at some point we will turn on the neglecting strategy)

Note. $v_{\max} = 0$ means we always renew and $v_{\max} = b + 1$ means we never renew.

We do by having

- $v_{\text{crit}} = 0$ if $g \leq c\lambda(1 - R)$
- $v_{\text{crit}} = k$ if $c\lambda(1 - R) + c(k - 1) < g \leq c\lambda(1 - R) + kc$ for $k = 1, ..., v_{\max} - 1$

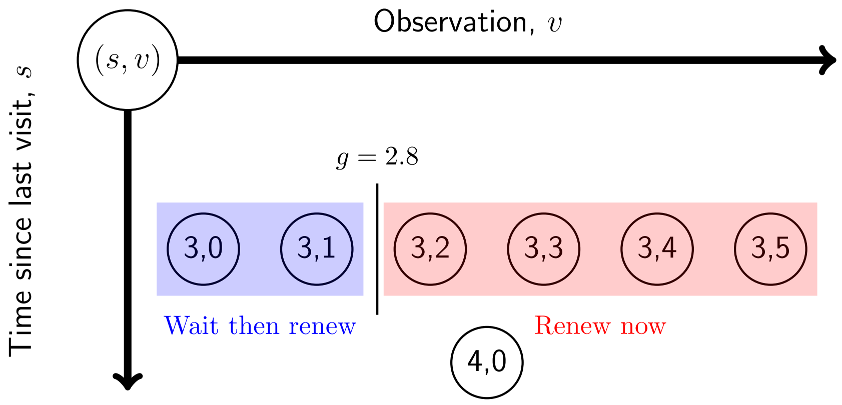

Figure 3.3: With $c = \lambda = 1$, $B = 4.4 \implies R = 0.4$ so $v_{\text{crit}} = 2$ when $2.6 \leq g < 3.6$ (e.g $g = 2.8$)

- $v_{\text{crit}} = v_{\max}$ if $c\lambda(1-R) + c(v_{\max} - 1) < g \leq c\lambda$

Let us say that our $v_{\text{crit}} = k$ then we have that

$$\begin{aligned}
g_k(\omega) &= \frac{\text{Expected cost per renewal}}{\text{Expected renewal length}} \\
&= \frac{\omega P(TPo(\lambda,b) \geq k) + (\omega + c\lambda(1-R))P(TPo(\lambda,b) = 0) + ... + (\omega + c\lambda(1-R))P(TPo(\lambda,b) = k-1)}{(\lfloor B \rfloor + 1)P(TPo(\lambda,b) \geq k) + (\lfloor B \rfloor + 2)P(TPo(\lambda,b) \leq k-1)} \\
&= \frac{\omega P(TPo(\lambda,b) \geq k) + (\omega + c\lambda(1-R))P(TPo(\lambda) \leq k-1) + c\sum_{i=0}^{k-1} iP(TPo(\lambda,b) = i)}{(\lfloor B \rfloor + 1)P(TPo(\lambda,b) \geq k) + (\lfloor B \rfloor + 2)(1 - P(TPo(\lambda,b) \geq k))} \\
&= \frac{\omega - c\lambda(1-R)P(TPo(\lambda,b) \geq k) + \sum_{i=0}^{k-1} iP(TPo(\lambda,b) = i)}{\lfloor B \rfloor + 2 - P(TPo(\lambda,b) \geq k)}
\end{aligned}$$

Now we can translate our bounds on g to become bounds on ω

For $v_{\text{crit}} = 0$ we have $0 \leq \omega \leq c\lambda(1-R)(\lfloor B \rfloor + 2) \equiv \Delta(0)$.

For $v_{\text{crit}} = k$ we have $\delta(k) < \omega \leq \Delta(k)$

For $v_{\text{crit}} = v_{\max}$ we have $\delta(v_{\max}) < \omega \leq \widetilde{\Delta}$

Where

- $\delta(k) = c(\lambda(1-R)(\lfloor B \rfloor + 2) + (k-1)(\lfloor B \rfloor + 2 - P(TPo(\lambda,b) \geq k)) - \sum_{i=0}^{k-1} iP(TPo(\lambda,b) = i))$
- $\Delta(k) = c(\lambda(1-R)(\lfloor B \rfloor + 2) + k(\lfloor B \rfloor + 2 - P(TPo(\lambda,b) \geq k)) - \sum_{i=0}^{k-1} iP(TPo(\lambda,b) = i))$
- $\widetilde{\Delta} = c(\lambda(\lfloor B \rfloor + 2 - RP(TPo(\lambda,b) \geq v_{\max})) - \sum_{i=0}^{v_{\max}-1} iP(TPo(\lambda,b) = i))$

Now we really want consistent bounds for ω (i.e to have no over or under lap), so we would like $\delta(k) = \Delta(k-1)$, luckily by considering the formula's this is true.

$$\begin{aligned}\delta(k) &= c(\lambda(1-R)(\lfloor B\rfloor+2)+(k-1)(\lfloor B\rfloor+2-P(TPo(\lambda,b)\geq k))-\sum_{i=0}^{k-1} iP(TPo(\lambda,b)=i))\\ &= c(\lambda(1-R)(\lfloor B\rfloor+2)+(k-1)(\lfloor B\rfloor+2-P(TPo(\lambda,b)\geq k-1))+(k-1)P(TPo(\lambda,b)=k-1)-\sum_{i=0}^{k-1} iP(TPo(\lambda,b)=i))\\ &= c(\lambda(1-R)(\lfloor B\rfloor+2)+(k-1)(\lfloor B\rfloor+2-P(TPo(\lambda,b)\geq k-1))-\sum_{i=0}^{k-2} iP(TPo(\lambda,b)=i))=\Delta(k-1)\end{aligned}$$

We can now create an index based on our bounds for ω Then if $\omega \in [\Delta(k), \Delta(k+1)]$ ($\Delta(-1)=0$ by declaration) we should pick $V_{\text{crit}}=k$ and when $\omega \geq \widetilde{\Delta}$ we should pick to never visit the node.

3.2 Index Heuristic

To develop a heuristic based on indices we attach a suffix to the prior index which helps us to decide if it is worth visiting now or later when we are in state $(\lfloor B\rfloor+1, v)$ dependent on the observed attackers v. We must now account for time, as the patroller doesn't really want to visit until it has been $s=\lfloor B\rfloor+1$ we will divide the index by this value to account for visiting early.

$$\begin{aligned}W_i(s_i,v_i) &= \begin{cases}\frac{\Delta_i(k)}{\lfloor B_i\rfloor+2-s_i} \text{ If } s_i\leq\lfloor B_i\rfloor+1, v_i<v_{i,\max}\\ \frac{\widetilde{\Delta}_i}{\lfloor B_i\rfloor+2-s_i} \text{ If } s_i\leq\lfloor B_i\rfloor+1, v_i=v_{i,\max}\end{cases}\\ &= \begin{cases}\frac{c_i(\lambda_i(1-R_i)(\lfloor B_i\rfloor+2)+v_i(\lfloor B_i\rfloor+2-P(TPo(\lambda_i,b_i)\geq v_i))-\sum_{j=0}^{v_i-1} jP(TPo(\lambda_i,b_i)=j))}{\lfloor B_i\rfloor+2-s_i} \text{ If } s_i\leq\lfloor B_i\rfloor+1, v_i<v_{i,\max}\\ \frac{c_i(\lambda_i(\lfloor B_i\rfloor+2-R_iP(TPo(\lambda_i,b_i)\geq v_{i,\max}))-\sum_{j=0}^{v_{i,\max}-1} jP(TPo(\lambda_i,b_i)=j))}{\lfloor B_i\rfloor+2-s_i} \text{ If } s_i\leq\lfloor B_i\rfloor+1, v_i=v_{i,\max}\end{cases}\end{aligned} \tag{10}$$

Where $R_i=B_i-\lfloor B_i\rfloor$, $v_{i,\max}=\max\{v\in\{0,1,...,b_i+1\}\,|\,v\leq\lambda_iR_i\}$

and

$$\begin{aligned}W_i(s_i=\lfloor B_i\rfloor+2,v_i) &= \widetilde{\Delta}_i\\ &= c_i(\lambda_i(\lfloor B_i\rfloor+2-R_iP(TPo(\lambda_i,b_i)\geq v_{i,\max}))-\sum_{j=0}^{v_{i,\max}-1} jP(TPo(\lambda_i,b_i)=j))\end{aligned} \tag{11}$$

Note. Because the set of Δ's are monotone, this index is monotone in both s_i and v_i. We also note that $W_i(\lfloor B_i\rfloor+1, v_{i,\max}) = W_i(\lfloor B_i\rfloor+2, v_i)$ for any v_i.

To implement the heuristic we initially assume that the patroller has neglected the region for a long period of time, that will be to set $s_i=\lfloor B_i\rfloor+2$, calculate the index for all the nodes and pick the node with the largest index (that is we will start at the node with the largest $\widetilde{\Delta}_i$). We then precede from here by calculating the indices of all adjacent nodes and moving to the node with the highest index, we repeat this process.

As the index is monotone and increasing in time, we will have that a nodes index grows (up to a cap of $\widetilde{\Delta}_i$) with time. However upon visiting it is not necessarily true that its it returns to the lowest value (as it may go to a state with a high v_i causing it to be higher).

As we only have control over when we visit, we note that the value $W_i(s_i,v_i)$ is the value at which the patroller is indifferent from visiting at s_i or s_i+1 time units.

3.3 Lower bound

Recall $C^{\text{OPT}}\geq C^{\text{TR}}\geq C(\omega)$. We will aim to compute the tighest bound by getting $C_{max}(\omega)=\max_{\omega\geq 0} C(\omega)$.

For a given ω we will define

$$K_i(\omega, v_i) = \begin{cases} \infty \text{ If } \omega > W_i(\lfloor B_i \rfloor + 2, 0) \\ \min\{k \,|\, W_i(k, v_i) > \omega\} \text{ Otherwise} \end{cases} \tag{12}$$

so that $K_i(\omega)$ represents the optimal interval between visits to node i when the last visit observed v_i attackers. That is how long to wait after we've just visited and observed v_i attackers.

So $C(\omega) = \sum_{i=1}^{n} C_i(\omega) - \omega$ where $C_i(\omega)$ is the optimal long run cost rate for node i when charged ω to visit.

That is

$$C_i(\omega) = \begin{cases} c_i\lambda_i \text{ If } \omega > W_i(\lfloor B_i \rfloor + 2, 0) \\ E[g_{Y_i}(\omega)] \text{ Otherwise} \end{cases} \tag{13}$$

$$= \begin{cases} c_i\lambda_i \text{ If } \omega > W_i(\lfloor B_i \rfloor + 2, 0) \\ \sum_{j=0}^{b_i} g_j(\omega) P(TPo(\lambda_i, b_i) = j) \text{ Otherwise} \end{cases} \tag{14}$$

Where $Y_i \sim TPo(\lambda_i, b_i)$.

We could also suggest that $C_i(\omega) = \sum_{j=0}^{b_i} C_i(\omega, v_j) P(TPo(\lambda_i, b_j))$ where $C_i(\omega, v_j)$ is the optimal long run cost rate for node i when charged ω and when we just observed v_j attackers.

That is $C_i(\omega, v_j) =$

4 Bernoulli Attack time

Consider the case where $X = \begin{cases} x_1 \text{ with probability } p \\ x_2 \text{ with probability } 1-p \end{cases}$ then we apply the same logic to attempt to get a decision dependent on the visiting cost. We will assume without loss of generality that $x_2 > x_1$, then $B = x_2$. We will get some reduction of the state space as before, but it will not be as drastic, by applying the same logic there may be a gap between some states we will never choose to visit. We will formally show this reduction in the appendix **??**

We will limit the state space with non-zero observed attackers to have either $s_j = \lfloor x_1 \rfloor + 1$ or $s_j = \lfloor x_2 \rfloor + 1$, due to the first one catching all the attacks caught for any $s_j < \lfloor x_1 \rfloor + 1$ and the second one catching all attacks caught for any $\lfloor x_1 \rfloor < s_j < \lfloor x_2 \rfloor + 1$.

So in the Bernoulli case $\Omega\{(\lfloor x_1 \rfloor + 1, o) \,|\, o = 0, 1, ..., b\} \cup \{(\lfloor x_1 \rfloor + 1, o) \,|\, o = 0, 1, ..., b\} \cup \{(\lfloor x_1 \rfloor + 2, o)\}$.

We will be assuming that $\lfloor x_1 \rfloor \neq \lfloor x_2 \rfloor$ as otherwise it follows that only one s_j survives and we fall into the deterministic category (bar some changes to the values) see Apendix B.

Suppose we are in the state $(\lfloor x_1 \rfloor + 1, o)$ for some $o = 1, 2, ..., b$ then our decision is either to

- Visit now
- Wait till we are in state $(\lfloor x_2 \rfloor + 1, o)$ and then visit
- Wait till we are in state $(\lfloor x_2 \rfloor + 2, o)$ and then visit
- Wait till we are in state $(\lfloor x_2 \rfloor + 2, o)$ and wait

Now we write down the long-run average costs of following such a strategy

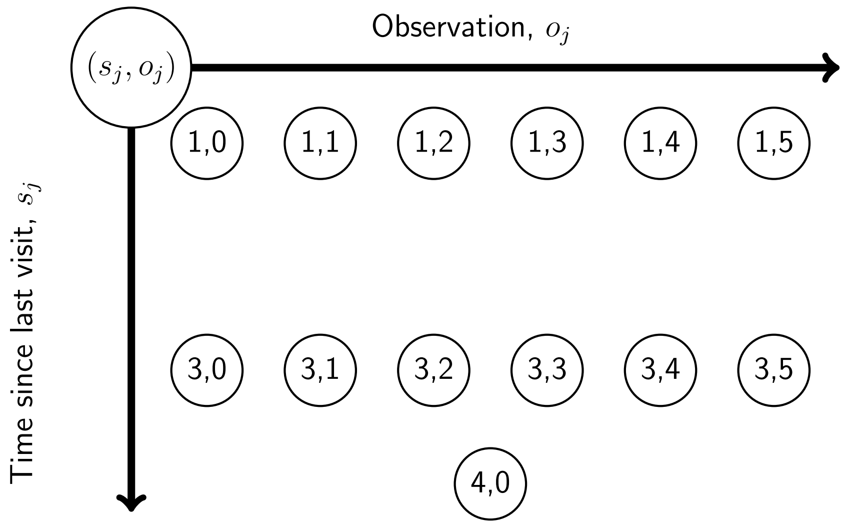

Figure 4.1: Bernoulli with $b_j = 5$, $x_{1,j} = 3.1$ and $x_{2,j} = 1.1$

- Visit now:

$$\frac{c\lambda \int_0^{\lfloor x_1 \rfloor} P(X \leq t)dt + \omega}{\lfloor x_1 \rfloor + 1} = \frac{\omega}{\lfloor x_1 \rfloor + 1} \tag{15}$$

- Wait till state $(\lfloor x_2 \rfloor + 1, o)$ and then visit:

$$\begin{aligned} &\frac{c\lambda \int_0^{\lfloor x_2 \rfloor} P(X \leq t)dt + coP(X \leq \lfloor x_2)\rfloor) + \omega}{\lfloor x_2 \rfloor + 1} \\ &= \frac{c\lambda(\lfloor x_2 \rfloor - x_1)p + cop + \omega}{\lfloor x_2 \rfloor + 1} \end{aligned} \tag{16}$$

- Wait till state $(\lfloor x_2 \rfloor + 2, 0)$ and then visit:

$$\begin{aligned} &\frac{c\lambda \int_0^{\lfloor x_2 \rfloor + 1} P(X \leq t)dt + coP(X \leq \lfloor x_2 \rfloor + 1) + \omega}{\lfloor x_2 + 2 \rfloor} \\ &= \frac{c\lambda((x_2 - x_1)p + (\lfloor x_2 \rfloor + 1 - x_2)) + co + \omega}{\lfloor x_2 \rfloor + 2} \end{aligned} \tag{17}$$

- In state $(\lfloor x_2 \rfloor + 2, 0)$ waiting $k \geq 1$ then visiting:

$$\begin{aligned} &\frac{c\lambda \int_0^{\lfloor x_2 \rfloor + 1 + k} P(X \leq t)dt + coP(X \leq \lfloor x_2 \rfloor + 1 + k) + \omega}{\lfloor x_2 + 2 + k \rfloor} \\ &= \frac{c\lambda((x_2 - x_1)p + (\lfloor x_2 \rfloor + 1 + k - x_2)) + co + \omega}{\lfloor x_2 \rfloor + 2 + k} \end{aligned} \tag{18}$$

Very similar to before we will look at when certain costs are better. Starting with Visit now beating all others if $\omega < \frac{cp(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_2 \rfloor - x_1) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor}$ and $\omega < \frac{c(\lfloor x_1 \rfloor + 1)(\lambda(x_2 - x_1)p + (\lfloor x_2 \rfloor - x_2 + 1) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor + 1}$ and $\omega < \frac{c(\lfloor x_1 \rfloor + 1)(\lambda(x_2 - x_1)p + (\lfloor x_2 \rfloor - x_2 + 1 + k) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor + 1 + k}$ for all $k \geq 1$. The first inequality guarantees the other two so we get

Visit now if:

$$\omega < \frac{cp(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_2 \rfloor - x_1) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor} \tag{19}$$

We can similarly find the visit in state $(\lfloor x_2 \rfloor + 1, o)$ by requiring that $\omega > \frac{cp(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_2 \rfloor - x_1) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor}$ and $\omega < c(\lambda(p((x_2 - x_1)(\lfloor x_2 \rfloor + 1) - (\lfloor x_2 \rfloor - x_1)(\lfloor x_2 \rfloor + 2)) + (\lfloor x_2 \rfloor - x_2 + 1)(\lfloor x_2 \rfloor + 1) + o((1 - p)(\lfloor x_2 \rfloor + 1) - p))$ and $\omega < \frac{c}{k+1}(\lambda(p((x_2 - x_1)(\lfloor x_2 \rfloor + 1) - (\lfloor x_2 \rfloor - x_1)(\lfloor x_2 \rfloor + 2 + k)) + (\lfloor x_2 \rfloor - x_2 + k + 1)(\lfloor x_2 \rfloor + 1) + o((1 - p)(\lfloor x_2 \rfloor + 1) - p(k + 1)))$. Note the second inequality implies the third one, so

Visit in state $(\lfloor x_2 \rfloor + 1, o)$ if:

$$\begin{aligned}
&\frac{cp(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_2 \rfloor - x_1) + o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor} < \omega < \\
&c(\lambda(p((x_2 - x_1)(\lfloor x_2 \rfloor + 1) - (\lfloor x_2 \rfloor - x_1)(\lfloor x_2 \rfloor + 2)) + (\lfloor x_2 \rfloor - x_2 + 1)(\lfloor x_2 \rfloor + 1) + o((1-p)(\lfloor x_2 \rfloor + 1) - p)) \\
&= c(\lambda(1-p)((\lfloor x_2 \rfloor - x_2)(\lfloor x_2 \rfloor + 1) + \lfloor x_2 \rfloor) + 1 + px_1 + o((1-p)(\lfloor x_2 \rfloor + 1) - p))
\end{aligned} \tag{20}$$

Again we could possible not have this region if the left hand side overlaps the right (but for now we will ignore it).

Again we can similarly find the visit in state $(\lfloor x_2 \rfloor + 2, 0)$ by requiring that $\omega > \frac{cp(\lfloor x_1 \rfloor+1)(\lambda(\lfloor x_2 \rfloor - x_1)+o)}{\lfloor x_2 \rfloor - \lfloor x_1 \rfloor}$ and $\omega > c(\lambda(p((x_2 - x_1)(\lfloor x_2 \rfloor + 1) - (\lfloor x_2 \rfloor - x_1)(\lfloor x_2 \rfloor + 2)) + (\lfloor x_2 \rfloor - x_2 + 1)(\lfloor x_2 \rfloor + 1) + o((1-p)(\lfloor x_2 \rfloor + 1) - p))$ and $\omega < c(\lambda(1 + E[X]) - o)$. Now the second inequality implies the first one (???? double check this ????), so

Visit in state $(\lfloor x_2 \rfloor + 2, 0)$ immediately if:

$$\begin{aligned}
&c(\lambda(1-p)((\lfloor x_2 \rfloor - x_2)(\lfloor x_2 \rfloor + 1) + \lfloor x_2 \rfloor) + 1 + px_1 + o((1-p)(\lfloor x_2 \rfloor + 1) - p)) \\
&< \omega < c(\lambda(1 + E[X]) - o)
\end{aligned} \tag{21}$$

Again there is a possibility of overlap (but for now we will ignore it)

Finally we will never visit if:

$$\omega > c(\lambda(1 + E[X]) - o) \tag{22}$$

Example 4.1. Suppose $x_1 = 0.1$,$x_2 = 2.1$ with $p = 0.2$ and $b = 5$, then our state space is reduced to $\Omega = \{(1, o)\,|o = 0, 1, ..., 5\} \cup \{(1, o)\,|o = 0, 1, ..., 5\} \cup \{(4, 0)\}$ and we make the decision at state $(1, o)$. We start by calculating the various critical regions, for which we will use $\lambda = 20$ and $o = 1$.

First critical value(Equation 12): $\frac{c(0+1)(20\times(2-0.1)+1)}{2-0} = \frac{39c}{2} = 19.5c$

Second critical value(Equation 13): $c(20 \times (1-0.2)((2-2.1)(2+1)+1)+1+0.2\times 0.1+1\times((1-0.2)(2+1)-0.2)) = 14.42c$

Third critical value(Equation 14): $c(20 \times (1 + 0.2 \times 0.1 + 0.8 \times 2.1) - 1) = 53c$.

So in this case the decision will be based on the cost, ω:

- Visit immediately if $\omega \leq 19.5c$
- Wait till $(4, 0)$ (three time periods) and visit immediately if $19.5c < \omega < 53c$
- Never visit if $\omega \geq 53c$

4.1 Correction to approach for bernoulli

We will first have a concrete argument as to why we can renew all states, (s, v) with $s < \lfloor x_1 \rfloor + 1$. From this position consider the policy which waits k time periods and then renews and follows the optimally policy, σ, with $k = 0, ..., \lfloor x_1 \rfloor + 1 - s$.

Using such a policy will get us that

$$V_n^{\pi_k}(s, v) = \omega + E[V_{n-k-1}^{\sigma}(\theta)] \tag{23}$$

Where θ is the state upon renewal (i.e the state $(1, V) \sim (1, TPo(\lambda))$

Now we will pick policy π_{k+1} over π_k (or be indifferent) if

$$\begin{aligned}
&\lim_{n\to\infty} V_n^{\pi_k}(s, v) - V_n^{\pi_{k+1}}(s, v) \geq 0 \\
\iff &\lim_{n\to\infty} E[V_{n-k-1}^{\sigma}(\theta) - V_{n-k-2}^{\sigma}(\theta)] \geq 0 \\
\iff &g \geq 0
\end{aligned}$$

Where g is the average long-run costs of the system, coming from $V_n = ng + \phi(s, v)$ for large n, where $\phi(s, v)$ is some initial bias for not starting in the equilibrium but from the initial state (s, v).

Now as the Dynamic Programming only have positive costs, it is impossible for $g < 0$. So we have $g \geq 0$, so it is better to pick policy π_{k+1} over π_k , as the argument holds for all $k = 0, ..., \lfloor x_1 \rfloor + 1 - s$ (and all v) it is best to wait till $(\lfloor x_1 \rfloor + 1, v)$ before making a decision, i.e if we are too renew we might as well renew at $(\lfloor x_1 \rfloor + 1, v)$ (though we might not renew here).

This formally shows the removal of the initial section of states (i.e those at the top).

Now we will skip to the state $(\lfloor x_2 \rfloor + 2, 0)$ and again suggest a policy, π_k which waits k time periods before renewing and then follows some optimal policy, σ.

Using such as policy will get us

$$\begin{aligned} V_n^{\pi_k}(s, v) &= \omega + c\lambda pk + c\lambda(1-p)k + E[V_{n-k-1}^{\sigma}(\theta)] \\ &= \omega + c\lambda k + E[V_{n-k-1}^{\sigma}(\theta)] \end{aligned}$$

so we will pick policy π_{k+1} over π_k (or be indifferent) if

$$\begin{aligned} &\lim_{n\to\infty} V_n^{\pi_k}(s, v) - V_n^{\pi_{k+1}}(s, v) \geq 0 \\ \iff &\lim_{n\to\infty} -c\lambda + E[V_{n-k-1}^{\sigma}(\theta) - V_{n-k-2}^{\sigma}(\theta)] \\ \iff &-c\lambda + g \geq 0 \\ \iff &f \geq c\lambda \end{aligned}$$

So hence if $g \geq c\lambda$ we will wait forever, as this has no dependence on k.

We will now argue that $g_{\max} = c\lambda$, that is the worst long-run average cost is $c\lambda$. This can be be seen by the strategy, π_{neg}, a strategy which never renews no matter what state we are in. I.e the patroller neglects the node. We will have that, for large n

$$V_n^{\pi_{\text{neg}}}(s, v) = nc\lambda + \phi(s, v)$$

We can observe this by looking at the actual decay of the process under a policy that never renews

$$\begin{aligned} V_n^{\pi_{\text{neg}}}(s, v) &= \begin{cases} nc\lambda \text{ If } s \geq \lfloor x_2 \rfloor + 2 \\ V_{n-(\lfloor x_2 \rfloor + 2 - s)} + \phi(s, v) \text{ If } s \leq \lfloor x_2 \rfloor + 1 \end{cases} \\ &= \begin{cases} nc\lambda \text{ If } s \geq \lfloor x_2 \rfloor + 2 \\ (n - (\lfloor x_2 \rfloor + 2 - s))c\lambda + \phi(s, v) \text{ If } s \leq \lfloor x_2 \rfloor + 1 \end{cases} \end{aligned}$$

where

$$\begin{aligned} \phi(s, v) &= \begin{cases} \underbrace{vc}_{\text{Observed finishing}} + \underbrace{c\lambda p(\lfloor x_2 \rfloor - x_1 + 1)}_{x_1 \text{ type attackers finishing}} + \underbrace{c\lambda(1-p)(\lfloor x_1 \rfloor - x_1 + 1)}_{x_2 \text{ type attackers finishing}} \text{ If } s \leq \lfloor x_1 \rfloor + 1 \\ \underbrace{vc(1-p)}_{\text{Observed } x_2 \text{ types finishing}} + \underbrace{c\lambda p(\lfloor x_2 \rfloor - x_1 + 1 - s)}_{x_1 \text{ type attackers finishing}} + \underbrace{c\lambda(1-p)(\lfloor x_1 \rfloor - x_1 + 1)}_{x_2 \text{ type attackers finishing}} \text{ If } \lfloor x_1 \rfloor + 2 \leq s \leq \lfloor x_2 \rfloor + 1 \\ 0 \text{ If } s \geq \lfloor x_2 \rfloor + 2 \end{cases} \\ &= \begin{cases} vc + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) + c\lambda p(x_2 - x_1) \text{ If } s \leq \lfloor x_1 \rfloor + 1 \\ vcp + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) + c\lambda p(x_2 - x_1 - s) \text{ If } \lfloor x_1 \rfloor + 2 \leq s \leq \lfloor x_2 \rfloor + 1 \\ 0 \text{ If } s \geq \lfloor x_2 \rfloor + 2 \end{cases} \end{aligned}$$

is the ‘transfer’/bias from the initial state to $(\lfloor x_2 \rfloor + 2, 0)$.

Hence for any service cost, ω we can achieve a $g = c\lambda$. So we will always be in the case of $g \leq c\lambda$ and hence will renew if we hit the state $(\lfloor x_2 \rfloor + 2, 0)$.

Now in this cases of $g \leq c\lambda$ (where we will not just follow neglection) we need to make a decision about what to do at states (s, v) for $\lfloor x_1 \rfloor + 1 \leq s \leq \lfloor x_2 \rfloor + 1$, $0 \leq v \leq b$.

The real question again is how long we wait, before renewal. Let us suppose we are in one of these states (s, v) then we notice that we will have slightly different costs depending which region we are in and how long we wait. We again use π_k be the policy to wait k $(k = 0, 1, ..., \lfloor x_2 \rfloor - \lfloor x_2 \rfloor + 1)$ time units before renewing and then following some optimal policy, σ.

If we are in $(\lfloor x_1 \rfloor + 1, v)$ then we will get

$$\begin{aligned}
V_n^{\pi_k}(\lfloor x_1 \rfloor + 1, v) &= \begin{cases} \omega + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = 0 \\ \omega + vcp + c\lambda p(\lfloor x_1 \rfloor - x_1 + 1 + (k-1)) + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } 1 \leq k \leq \lfloor x_2 \rfloor - \lfloor x_1 \rfloor \\ \omega + vc + c\lambda p(\lfloor x_1 \rfloor - x_1 + 1 + \lfloor x_2 \rfloor - \lfloor x_1 \rfloor) + c\lambda(1-p)(\lfloor x_2 \rfloor - x_2 + 1) + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = \lfloor x_2 \rfloor - \lfloor x_1 \rfloor + 1 \end{cases} \\
&= \begin{cases} \omega + E[V_{n-1}^{\sigma}(\theta)] \text{ If } k = 0 \\ \omega + vcp + c\lambda p(\lfloor x_1 \rfloor - x_1 + 1 + (k-1))E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } 1 \leq k \leq \lfloor x_2 \rfloor - \lfloor x_1 \rfloor \\ \omega + vc + c\lambda p(x_2 - x_1) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) \text{ If } k = \lfloor x_2 \rfloor - \lfloor x_1 \rfloor + 1 \end{cases}
\end{aligned}$$

If we are in (s, v) for $1 \leq s \leq \lfloor x_2 \rfloor$

$$\begin{aligned}
V_n^{\pi_k}(\lfloor x_1 \rfloor + 1, v) &= \begin{cases} \omega + c\lambda pk + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } 0 \leq k \leq \lfloor x_2 \rfloor - \lfloor x_1 \rfloor - s \\ \omega + vc(1-p) + c\lambda p(\lfloor x_2 \rfloor - \lfloor x_1 \rfloor - s + 1) + c\lambda(1-p)(\lfloor x_2 \rfloor - x_2 + 1) + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = \lfloor x_2 \rfloor - \lfloor x_1 \rfloor + 1 - \end{cases} \\
&= \begin{cases} \omega + c\lambda pkE[V_{n-k-1}^{\sigma}(\theta)] \text{ If } 0 \leq k \leq \lfloor x_2 \rfloor - \lfloor x_1 \rfloor - s \\ \omega + vc(1-p) + c\lambda p(x_2 - \lfloor x_1 \rfloor - s) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) \text{ If } k = \lfloor x_2 \rfloor - \lfloor x_1 \rfloor - s + 1 \end{cases}
\end{aligned}$$

If we are in $(\lfloor x_2 \rfloor + 1, v)$

$$\begin{aligned}
V_n^{\pi_k}(\lfloor x_2 \rfloor + 1, v) &= \begin{cases} \omega + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = 0 \\ \omega + vc(1-p) + c\lambda p + c\lambda(1-p)(\lfloor x_2 \rfloor - x_2 + 1) + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = 1 \end{cases} \\
&= \begin{cases} \omega + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = 0 \\ \omega + vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) + E[V_{n-k-1}^{\sigma}(\theta)] \text{ If } k = 1 \end{cases}
\end{aligned}$$

We now seek to compare policies for states, we we shall start with the state, $(\lfloor x_2 \rfloor + 1, v)$ then we will pick policy π_0 over π_1 if

$$\begin{aligned}
&\lim_{n\to\infty} V_n^{\pi_1}(\lfloor x_2 \rfloor + 1, v) - v_n^{\pi_0}(\lfloor x_2 \rfloor + 1, v) \geq 0 \\
\iff &\lim_{n\to\infty} \omega + vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) + E[V_{n-2}^{\sigma}(\theta)] - (\omega + E[V_{n-1}^{\sigma}(\theta)]) \\
\iff &\lim_{n\to\infty} E[V_{n-2}^{\sigma}(\theta) - V_{n-1}^{\sigma}(\theta)] + vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1) \\
\iff &g \leq vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor)) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1)
\end{aligned}$$

So we will renew immediately in $(\lfloor x_2 \rfloor + 2, 0)$ from here if $g \leq vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor)) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1)$. We will use $R_2 = x_2 - \lfloor x_2 \rfloor$ $(0 \leq R_2 < 1)$.

So to have the possibility to wait and renew we will need that $vc(1-p) + c\lambda pR_2 + c\lambda(1 - R_2) \leq c\lambda \iff c(1-p)(v - R_2\lambda) \leq 0 \iff v - R_2\lambda \leq 0 \iff v \leq R_2\lambda$.

Note. We note that this sort of result is analogous to the deterministic case.

now we note that we can again create a $v_{\text{crit}}(\lfloor x_2 \rfloor + 1)$, from which we will renew immediately for all $v \geq v_{\text{crit}}(\lfloor x_2 \rfloor + 1)$ and for $v < v_{\text{crit}}(\lfloor x_2 \rfloor + 1)$ we wait.

We rearrange $g \geq vc(1-p) + c\lambda p(x_2 - \lfloor x_2 \rfloor)) + c\lambda(\lfloor x_2 \rfloor - x_2 + 1)$ to get $v_{\text{crit}}(\lfloor x_2 \rfloor + 1) = \left\lceil \frac{g}{c(1-p)} - \frac{\lambda}{(1-p)} - \lambda R_2 \right\rceil$.

We again know that, $g_{\max} = c\lambda$ so we get the maximum $v_{\text{crit}} = \left\lceil \frac{c\lambda}{c(1-p)} - \frac{\lambda}{(1-p)} - \lambda R_2 \right\rceil = \lceil \lambda R_2 \rceil$

So when $R_2 = 0$ then we immediately renew and will never wait in the state $(\lfloor x_2 \rfloor + 1, v)$ for any v.

5 Ideas for future

- Extend X to be a binomial distribution
- Extend X to be any discrete distribution
- Extend X to be a normal distribution by approximation to binomial
- Test Deterministic case against strategic attacker (non-random) on extended star graph

Appendices

A Observations are always zero

On a single node we are limited to the state space of $\Omega = \{(1,0), ..., (\lfloor B \rfloor + 2, 0)\}$, then on this state space we can implement a policy which returns every k time units, this will gives an average long run cost of

$$f(k) = \frac{c\lambda \int_0^{k-1} P(X \leq t)dt + \omega}{k} \tag{24}$$

So to find out when the patroller would be indifferent from choosing to return every k or every $k+1$, solve $f(k+1) - f(k) = 0$ giving

$$\frac{1}{k(k+1)}(c\lambda(k \int_{k-1}^{k} P(X \leq t)dt - \int_0^{k-1} P(X \leq t)dt) - \omega) = 0$$

Prompting an index of

$$W(k) = c\lambda(k \int_{k-1}^{k} P(X \leq t)dt - \int_0^{k-1} P(X \leq t)dt)$$

We note that $W(0) = 0$ and for $k \geq B+1$ $W(k) = c\lambda(k - int_0^{k+1} P(X \leq t)dt = c\lambda(1 + \int_0^{k-1} P(X > t)dt) = c\lambda(1 + E[X])$. We will now show that:

- $W(k)$ is non-decreasing
- The optimal policy when $\omega \in [W(k-1), W(k)]$ is to visit every k time units
- If $w \geq c\lambda(1 + E[X])$ then it is optimal to never visit

Proof. First

$$\begin{aligned} W(k+1) - W(k) &= c\lambda((k+1) \int_k^{k+1} P(X \leq t)dt - \int_0^k P(X \leq t)dt \\ &\quad - (k \int_{k-1}^{k} P(X \leq t)dt - \int_0^{k-1} P(X \leq t)dt)) \\ &= c\lambda((k+1) \int_k^{k+1} P(X \leq t)dt - k \int_{k-1}^{k} P(X \leq t)dt - \int_k^{k-1} P(X \leq t)dt) \\ &= c\lambda(k+1)(\int_k^{k+1} P(X \leq t)dt - \int_{k-1}^{k} P(X \leq t)dt) \geq 0 \end{aligned}$$

As $P(X \le t)$ is non-decreasing.

Second if $\omega \in [W(k-1), W(k)]$ then we will show that $f(m)$ is non-increasing for $m \le k$ and non-decreasing for $m \ge k$.

For $m \le k$

$$\begin{aligned} f(m) - f(m-1) &= \frac{1}{m(m-1)}\left(c\lambda(m-1)\int_0^{m-1} P(X \le t)dt - c\lambda m \int_0^{m-2} P(X \le t)dt - \omega\right) \\ &= \frac{1}{m(m-1)}(W(m-1) - \omega) \le \frac{1}{m(m-1)}(W(m-1) - W(k-1)) \le 0 \end{aligned}$$

Similarly for $m \ge k$

$$f(m+1) - f(m) = \frac{1}{m(m+1)}(W(m) - \omega) \ge \frac{1}{m(m+1)}(W(m) - W(k)) \ge 0$$

Hence choosing to visit every k time units is optimal.

Third and finally our upper limit of $c\lambda(1 + E[X])$ (Which is $W(k)$ for $k \ge B+1$) means we are indifferent from picking k and $k+1$ for $k \ge B+1$ so we will never visit. I.e $f(k+1) \le f(k) \iff w \ge W(k)$ so not optimal if $f(k+1) \ge f(k) \, \forall k \iff w \ge \sup_{k=1,2,\ldots} W(k) = \lim_{k \to \infty} W(k) = c\lambda(1 + E[X])$ □

B Bernoulli Attack time with equal floors

If we have $\lfloor x_1 \rfloor = \lfloor x_2 \rfloor$ then $\Omega = \{(\lfloor x_1 \rfloor + 1, o) \,|\, o = 0, 1, ..., b\} \cup \{(\lfloor x_1 \rfloor + 2, o)\}$

Suppose we are in the state $(\lfloor x_1 \rfloor + 1, o)$ for some $o = 1, 2, ..., b$ then our decision process is the same as the deterministic case

- Visit now
- Wait till we are in state $(\lfloor x_1 \rfloor + 2, 0)$ and wait for k time units before visiting, $k \ge 0$

We again write down long-run average costs of following such a strategy

- Visit now:

$$\frac{c\lambda \int_0^{\lfloor x_1 \rfloor} P(X \le t)dt + \omega}{\lfloor x_1 \rfloor + 1} = \frac{\omega}{\lfloor x_1 \rfloor + 1} \tag{25}$$

- Wait till state $(\lfloor x_1 \rfloor + 2, 0)$ then wait $k \ge 0$ time units before visiting:

$$\begin{aligned} &\frac{c\lambda \int_0^{\lfloor x_1 \rfloor + 1 + k} P(X \le t)dt + coP(X \le \lfloor x_1 \rfloor + 1 + k) + \omega}{\lfloor x_1 \rfloor + 2 + k} \\ &= \frac{c\lambda(p(x_2 - x_1) + (\lfloor x_1 \rfloor + 1 + k - x_2)) + co + \omega}{\lfloor x_1 \rfloor + 2 + k} \end{aligned} \tag{26}$$

Now to visit now we will only do so if $\omega < c(\lfloor x_1 \rfloor + 1)(\lambda(p\frac{x_2 - x_1}{k+1} + (\frac{\lfloor x_1 \rfloor - x_2}{k+1} + 1)) + \frac{o}{k+1})$ for all $k \ge 0$, We need to know if $\lambda(\lfloor x_1 \rfloor - px_1 + px_2 - x_2) + o \begin{cases} > 0 \\ < 0 \end{cases}$. In the first case we get to visit if:

$$\omega < c(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_1 \rfloor - px_1 + px_2 - x_2 + 1) + o) \tag{27}$$

In the second case we get to visit if:

$$\omega < c\lambda(\lfloor x_1 \rfloor + 1) \tag{28}$$

If we do no visit initially then we will visit once transitioned and waited $k+1$ over k if $\omega > c(\lambda(1 + E[X]) - o)$.

For in between values (dependent on which case) we will get that we will transition and visit immediately. I.e if First case

$$c(\lfloor x_1 \rfloor + 1)(\lambda(\lfloor x_1 \rfloor - px_1 + px_2 - x_2 + 1) + o) < \omega < c(\lambda(1 + E[X]) - o) \tag{29}$$

Second case

$$c\lambda(\lfloor x_1 \rfloor + 1) < \omega < c(\lambda(1 + E[X]) - o) \qquad (30)$$

Again these regions may or may not exist dependent on their conditions.