**R.B. Kappetein**

# Optimal control of a server farm

**Master thesis, 24 January, 2013**

**Thesis supervisor: dr. F.M. Spieksma**

**Mathematical institute, University Leiden**

## Contents

## 1 Introduction

A server farm consists of ample servers that serve a stream of arriving customers. Upon a service completion, a server can be turned off. this might be beneficial to save power, and hence costs. However , for shutting down and starting up a server, extra power (e.e. costs) is incurred. Thus, there is a trade-off between the savings by turning servers off, and the extra costs made for the start-ups and shut-downs. We consider a model where arriving customers are taken into service directly. For this, we study the optimal control of such a server farm, that is, we derive the optimal dynamic control policy deciding when a server should be turned off after a demand completion, minimizing the expected costs.[1]
A server farm consists of an unlimited amount of servers, that serve an arriving stream of customers. Each server in the system can be in one of the following states: busy, off or standby. Busy servers consume power, standby servers consume less power, and off servers consume no power at all. Hence, by turning an idle server off, power and hence costs, can be saved. However extra power (i.e. costs) is incurred for starting up and shutting down a server . In this way, there is a trade-off between the potential cost savings, and the extra costs incurred. Thus the question is, when servers should be turned off, and when they should be idled. In this article you can find the optimal control scheme.[1]

Server farm power consumption accounts for more than 1.5% of the total electricity usage in the U.S., at a cost of nearly 4.5 billion. The rising cost of energy and the tremendous so naturally you want to minimize that power consumption.[3]

This model has been studied and has been formulated as a Markov Decision Process. There are two complicating factors: unbounded jumps, there are policies for which the associated Markov process is transient, e.g. the policy that always idles. Van Wijk et al cannot solve this problem and therefore they consider a variant with bounded service rates. In this thesis we consider the original problem: we show that we can reduce the problem to a finite state problem. This allows to show that an average expected optimal policy exists. We give an explicit algorithm to compute the optimal policy.

## 2 Model and notation

In this model we consider a server farm consisting of an unlimited amount of identical servers. Each server can be in one of three states: busy, idle or off. Customers arrive according to a Poisson process with parameter $\lambda > 0$. All customers require a service time that is exponentially distributed with parameter $\mu > 0$. An arriving customer will be served by an idle server. That idle server will then immediately be switched to busy. We assume that this switch is instantaneously. If there are no idle servers an off server will be switched on to serve the customer. There is no start up time for the server. So in both cases an arriving customer will be served immediately, since there is an unlimited amount of servers. When a customer leaves the system you can either idle that server or switch it off. There are costs for keeping the servers idle, switching them on and switching them off. The cost to put a server on are equal to $K^{on}$ and the cost of putting a server off is equal to $K^{off}$. The cost of the idle servers is equal to $c * i$ per time unit with $i$ the number of idle servers. The goal is to design an idling strategy that minimises the average expected cost per unit time

We can model this optimisation problem as a Markov Decision Process. The system state $X_t$ at time is a two-dimensional process, where $X_t = (i, j)$ denotes that $i$ servers are idle and $j$ busy at time $t$. The state space is therefore given by $\mathcal{S} = \mathbf{Z}^2$.

The possible actions in state $(i, j)$ with $i > 0$, are either to switch a server off immediately (0), to switch an server off (1) or the idle the server (2) when the next event is a service completion (1). Then the action space in state $(i, j)$ is equal to $A(i, j) = \{1, 2, 3\}$. The model description in (vW) does not allow for immediate switching off of servers. $A_t$, $t \geq 0$ describes the prescribed action at time $t$. In state $(0, j)$, $j > 0$, the action space $A(0, j) = \{1, 2\}$, since immediate server switch-off is impossible due to lack of idle servers. In state $(0, 0)$ no decision needs to be made, hence $A(0, 0) = \{0\}$.

In this thesis, we restrict to so-called stationary, deterministic strategies, $\mathcal{F}$. $f$ is a stationary, deterministic policy, if it assign an action to each state, i.o.w. $f(i, j) \in A(i, j)$ for all $(i, j) \in \mathcal{S}$.

The transition mechanism is now described by the following rates

$$q_{(i,j)(k,l)}(a) = \begin{cases} \lambda, & k = i-1, l = j+1, a \in A(i,j) & \text{if } i > 0 \\ j\mu & k = i, l = j-1, a = 1, & \text{if } (i,j) \neq (0,0) \\ j\mu & k = i+1, l = j-1, a = 2 & \text{if } (i,j) \neq (0,0) \\ \infty & k = i-1, l = j, a = 0 & \text{if } i \neq 0 \\ -\sum_{(k',l') \neq (i,j)} q_{(i,j)(k,l)}(a) & k = i, l = j \end{cases}$$

The costs associated with state $(i, j)$ consists of an action independent cost rate given by

$$c(i, j, a) = c \cdot i, a \in A(i, j)$$

and a lump cost associated with transitions: $d((0, j)(0, j+1), a) = K^{on}$, $a \in A(i, j)$, $d((i, j)(i - 1, j), 0) = K^{\mathit{off}} = d(i, j)(i, j - 1), 1)$.

Given that strategy $f$ is used, the state process is a Markov process, denoted by $X^f_t$ with possibly instantaneous states. Denote by $P^f_t$ the associated transition kernel and by $P^f$ the transition matrix of the associated jump chain. Then the average expected cost $\mathcal{C}^f(i, j)$ associated with strategy $f$, given initial state $(i, j)$ is given by

$$\begin{aligned} \mathcal{C}^f(i,j) &= \limsup_{T \to \infty} \frac{1}{T} \Big[ \int_0^T P^f_{t,(i,j)(k,l)} (c \cdot l + K^{on} \cdot \lambda \cdot \mathbf{1}_{(k=0)} + l\mu K^{\mathit{off}} \cdot \mathbf{1}_{(k>0, f(k,l)=1)}) dt \\ &\quad + \mathbf{E}\{N^f(T) \mid X^f(0) = (i,j)\} \cdot K^{\mathit{off}} \Big], \end{aligned}$$

where $N^f(T)$ is the number of instantaneous transitions of $X^f$ in $[0, T]$. A priori it is not even clear that $C^f(i, j)$ is independent of the initial state $(i, j)$.

The mathematical question is whether there exists a minimum cost strategy $f$, i.o.w. does there exist a strategy $f$ with

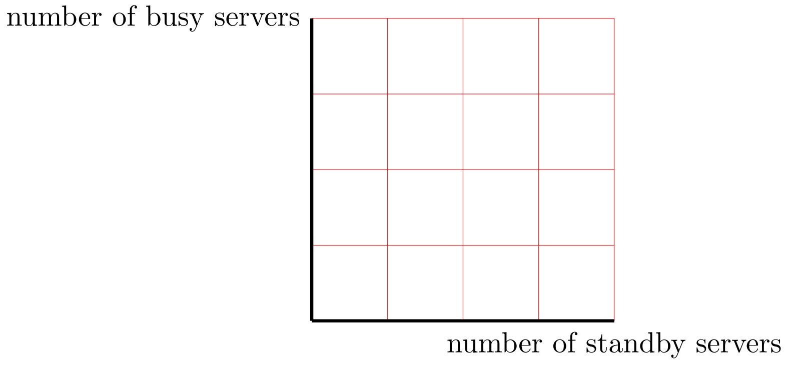
$$\mathcal{C}^f(i, j) = \inf_{g \in \mathcal{F}} \mathcal{C}^g(i, j), \qquad \text{for all } (i, j) \in \mathcal{S}.$$

If so, the objective is to give a fast (finite) procedure to determine the optimal strategy.

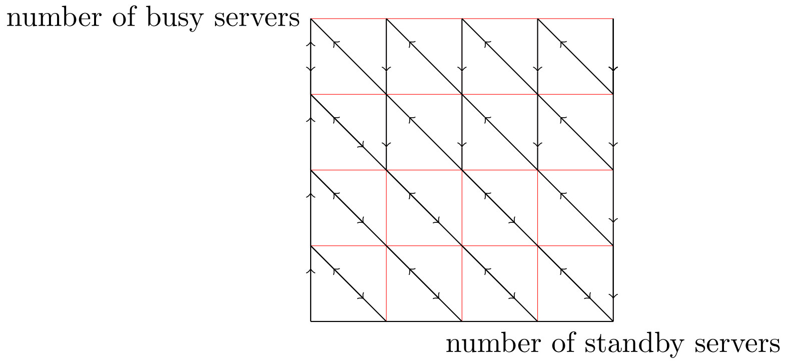
## 3 Basic properties and definitions

The number of customers in the system is completely independent of the number of idle servers, since every customers will be served immediately regardless of the number of idle servers. Therefore the number of customers in the system can be modelled as an $M|M|\infty$ queue.

In order to better understand the problem it's useful to visualize this in the following type of pictures.



If you have picture you can draw two arrows from each state to represent what happens when a customer leaves or enters the system. One arrow when a customer leaves and one for when a customer arrives. When a customer arrives you don't have any choice how to put the arrow but if a customer leaves you have two choices. You can either put the server off, so the arrow goes down or you can choose to idle the server and use that arrow. For an example you can look at the picture.



As you can see from the picture the arrows and the grid points form a directed graph.

The customers arrive according to a Poisson process so the interarrival times are exponentially distributed. The service time is also exponentially distributed. Since both the interarrival times and service times are exponentially distributed the system is memoryless. Since we know the interarrival times of the customers and the distribution of the service time of a customer we know the probability of going from one state to another given a strategy. The probability of going up from $(i, j)$ (i.e. j increases) is equal to $\frac{\lambda}{\lambda+j\mu}$ and the probability of going down (i.e. j decreases) is equal to $\frac{j\mu}{\lambda+j\mu}$.

**Definition 1.** *Diagonal $D(k)$ is the following set of points $\{(i, j)|i + j = k\}$. We call $D(k')$ a higher diagonal than $D(k)$ if $k' > k$.*

**Definition 2.** *Triangle $T(k)$ is the set points $\{(i, j)|0 \leq i + j \leq k\}$*

**Definition 3.** *A state is reachable if there is directed path from every point on the y-axis to that state.*

The motivation for this definition is that the $y$-axis is reached from any point in finite expected time.

**Definition 4.** *A system is stable if you reach every reachable state with probability 1 and the set of reachable states is non-empty.*

We do not require that every reachable state is reached in finite expected time, because one can construct strategies for which this does not hold. Moreover this is not relevant for our derivations.

## 4 Key properties

The following two lemma's are almost trivial, but they will prove to be vital for finding an optimal solution.

**Lemma 1.** *You can only get to a higher diagonal at* $(0, j)$.

*Proof.* Transitions to a higher diagonal only occur on the $y$-axis. □

**Lemma 2.** *You can only leave a triangle* $T(k)$ *at the point* $(0, k)$ *and the total time to leave the triangle is independent of the chosen strategy.*

*Proof.* Suppose you are in a triangle $T(k)$. From lemma 1 we know that you can only get above $D(k)$ at $(0, k)$. This immediately implies that you can only leave at $(0, k)$. You will leave triangle $T(k)$ when you are in state $(0, k)$ and a customer enters the system.
If you are in triangle $T(k)$ and there are $k$ customers in the system you can only be in state $(0, k)$. The total time to reach state $(0, k)$ is precisely the total time to reach state $k$ in the M/M/$\infty$-queue, which is independent of the chosen strategy. □

This lemma has an important consequence. The system is memoryless so you don't need to know what happened in the past. So the strategy you choose at your current point is independent on how you got there. The moment you leave a triangle is independent of the chosen strategy inside the triangle. It takes a finite time to leave the triangle and the place where you leave the triangle is independent of your chosen strategy. So it is sufficient to consider cost minimisation problem on triangles. In the next chapters we will derive a bound on the maximum triangle that we need to consider for the optimisation problem.

## 5 Stability

In this chapter we will look at the necessary conditions for a stable system. We first look at the number of customers in the system. As previously stated this can be modelled as an $M|M|\infty$ queue.

**Definition 5.** *$a(j)$ is the probability that there are $j+1$ customers in the system before the system gets empty given that you start with $j$ customers in the system.*

**Lemma 3.**

- $\dfrac{1}{\frac{(j+1)\mu}{\lambda}(1-a(j))+1} = a(j+1).$
- $\lim_{j\to\infty} a(j) = 0.$

*Proof.* We already know that $a(1)$ is equal to $\frac{\lambda}{\lambda+\mu}$. So we are going to look at $a(2)$. $a(2)$ is the probability that there are 3 customers in the system before the system gets empty if you start with 2 customers in the system. In order to calculate this we look at the following picture. The number in the box is the number of customers in the system and the probabilities to go from one state to another have been given.

$$\begin{tikzcd}
3 & 2 \arrow[l, bend right, "\frac{\lambda}{\lambda+2\mu}"'] \arrow[r, bend right, "\frac{2\mu}{\lambda+2\mu}"'] & 1 \arrow[l, bend right, "\frac{\lambda}{\lambda+\mu}"'] \arrow[r, bend right, "\frac{\mu}{\lambda+\mu}"'] & 0
\end{tikzcd}$$

As you can see from this picture you can directly get 3 customers in your system with probability $\frac{\lambda}{\lambda+2\mu}$. You can also go via 1, then to 2 and then go to 3. This has probability $\frac{2\mu}{\lambda+2\mu}\frac{\lambda}{\lambda+\mu}\frac{\lambda}{\lambda+2\mu}$. In this instance you make 1 loop. But you can also make $2, 3, \ldots$ loops. Each additional loop you have to multiply with another factor $\frac{2\mu}{\lambda+2\mu}\frac{\lambda}{\lambda+\mu}$. You can see you that this all adds up to:

$$\begin{aligned}
\frac{\lambda}{\lambda+2\mu}\sum_{n=0}^{\infty}\left(\frac{2\mu}{\lambda+2\mu}\frac{\lambda}{\lambda+\mu}\right)^n &= \frac{\lambda}{\lambda+2\mu}\frac{1}{1-\frac{2\mu}{\lambda+2\mu}\frac{\lambda}{\lambda+\mu}} \\
&= \frac{\lambda}{\lambda+2\mu(1-\frac{\lambda}{\lambda+\mu})} \\
&= a(2).
\end{aligned}$$

Now we know the probability $a(2)$. Next we are going to use $a(2)$ to calculate $a(3)$. Now suppose we have 3 customers in the system and we want to calculate $a(3)$. Then we get the following picture:

$$\begin{tikzcd}
4 & 3 \arrow[l, bend right, "\frac{\lambda}{\lambda+3\mu}"'] \arrow[r, bend right, "\frac{3\mu}{\lambda+3\mu}"'] & 2 \arrow[l, bend right, "\frac{\lambda}{\lambda+2\mu}"'] \arrow[r, bend right, "\frac{2\mu}{\lambda+2\mu}"'] & 1 \arrow[l, bend right, "\frac{\lambda}{\lambda+\mu}"'] \arrow[r, bend right, "\frac{\mu}{\lambda+\mu}"'] & 0
\end{tikzcd}$$

You can see that the probabilities are the same but there is an extra state. In this system you can directly go to 4 with probability $\frac{\lambda}{\lambda+3\mu}$ or you can go to 2 with probability $\frac{3\mu}{\lambda+3\mu}$. Since the tail is the same we already know the probability of going to 3 before visiting 0. This probability is equal to $a(2)$. And we know the probability of going to 0 before going before visiting 3. This is equal to $1-a(2)$. So you can replace picture 4 with this picture:

$$\begin{tikzcd}
4 & 3 \arrow[l, bend right, "\frac{\lambda}{\lambda+3\mu}"'] \arrow[r, bend right, "\frac{3\mu}{\lambda+3\mu}"'] & 2 \arrow[l, bend right, "a(2)"'] \arrow[r, bend right, "1-a(2)"'] & 0
\end{tikzcd}$$

You can now see that is the same picture as we've had with 2 customers in the system but with different probabilities. You can now compute the probability $a(3)$ in the same way.

$$\begin{aligned}
\frac{\lambda}{\lambda+3\mu}\sum_{n=0}^{\infty}(\frac{3\mu}{\lambda+3\mu}a_2)^n &= \frac{1}{\frac{3\mu}{\lambda}(1-a_2)+1} \\
&= a(3).
\end{aligned}$$

You can easily see that the tail of each picture is the same as the picture before it. So you can use your knowledge of the previous picture. The validity of the recursive formula in the first statement of the lemma can be proved inductively.

Now we are going to prove that $a(j) \to 0$ as $j \to \infty$. Suppose $a(n+1) \leq a(n)$ then:

$$\begin{aligned}
a(n+2) &= \frac{1}{\frac{(n+2)\mu}{\lambda}(1-a(n+1))+1} \\
&\leq \frac{1}{\frac{(n+2)\mu}{\lambda}(1-a(n))+1} \\
&< \frac{1}{\frac{(n+1)\mu}{\lambda}(1-a(n))+1} \\
&= a(n+1).
\end{aligned}$$

You can now easily see that $a(n+r) < a(n)$. Then we get:

$$\begin{aligned}
a(n+r+1) &= \frac{1}{\frac{(n+r)\mu}{\lambda}(1-a(n+r))+1} \\
&< \frac{1}{\frac{(n+r)\mu}{\lambda}(1-a(n))+1}.
\end{aligned}$$

We can now take the limit to infinity of the right-hand side.

$$\lim_{r\to\infty}\frac{1}{\frac{(r+n)\mu}{\lambda}(1-a(n))+1} = 0.$$

So if there exists $n$ such that $a(n+1) \leq a(n)$, then $a(j)$ goes to zero as j goes to infinity. We will prove this by contradiction, and so we're going to assume such $n$ does not exist. This means that $a(j)$ is a strictly increasing function in j.
Suppose that $a(j)$ is strictly increasing. That means that $1-a(j)$ must be strictly decreasing in j.

$$\begin{aligned}
1-a(j) > 1-a(j+1) &\Leftrightarrow 1-a(j) > 1-\frac{1}{\frac{j\mu}{\lambda}(1-a(j))+1}\\
&\Leftrightarrow 1-a(j) > \frac{\frac{j\mu}{\lambda}(1-a(j))}{\frac{j\mu}{\lambda}(1-a(j))+1}\\
&\Leftrightarrow (1-a(j))(\frac{j\mu}{\lambda}(1-a(j))+1) > \frac{j\mu}{\lambda}(1-a(j))\\
&\Leftrightarrow (1-a(j))(\frac{j\mu}{\lambda}(1-a(j))+1) - \frac{j\mu}{\lambda}(1-a(j)) > 0\\
&\Leftrightarrow \frac{j\mu}{\lambda}(1-a(j))^2 + (1-\frac{j\mu}{\lambda})(1-a(j)) > 0.
\end{aligned}$$

$$\begin{aligned}
\frac{j\mu}{\lambda}(1-a(j))^2 + (1-\frac{j\mu}{\lambda})(1-a(j)) &> 0\\
\frac{j\mu}{\lambda}(1-a(j)) + (1-\frac{j\mu}{\lambda}) &> 0\\
\frac{j\mu}{\lambda}(1-a(j)) &> \frac{j\mu}{\lambda}-1\\
(1-a(j)) &> \frac{(\frac{j\mu}{\lambda}-1)\lambda}{j\mu}\\
(1-a(j)) &> 1-\frac{\lambda}{j\mu}\\
\frac{\lambda}{j\mu} &> a(j).
\end{aligned}$$

If $a(j)$ is strictly increasing in j then $a(j) > a(1)$.

$$\begin{aligned}
\frac{\lambda}{j\mu} > a(j) > a(1) &\Rightarrow \frac{\lambda}{j\mu} > \frac{\lambda}{\lambda+\mu}\\
&\Rightarrow \lambda+\mu > j\mu\\
&\Rightarrow \frac{\lambda}{\mu}+1 > j.
\end{aligned}$$

This means that $a(j)$ can only be increasing if $j < \frac{\lambda}{\mu}+1$. So if j is larger then $\frac{\lambda}{\mu}+1$ then a(j) will be decreasing in j. And we've already seen that if it's decreasing then it goes to zero as j goes to infinity. This concludes the proof. □

**Lemma 4.** *A strategy has a stable system if and only if there exists $k \in \mathbb{N}$, such that each diagonal higher than $D(k)$ contains one state where the server is switched off upon a customer departure.*

*Proof.* Suppose a strategy has a stable system. Then there is at least one state that is reached with probability 1. If the probability that you go to $D(\infty)$ is positive then there cannot be any reachable state. Suppose that there is not a $k \in \mathbb{N}$ such that every diagonal higher than $D(k)$ contains one state where the server is switched off upon a customer departure. Then for any $m$ there is a $m' > m$ such that $D(m')$ only contains states where you idle the server upon a customer departure. Suppose you are on a lower diagonal than $D(m')$. The probability that at one point in time there are $m'$ customers in the system is 1. So there is a point in time that you are on $D(m')$. But once you are there you cannot go below $D(m')$ since there are no states where you switch the server off. So there is no path from any point on diagonal $D(m')$ to a point below that diagonal. This means that no point below $D(m')$ is reachable. But this property holds for any $m \in \mathbb{N}$. So there is no point that is reachable. So the system is not stable because there are no reachable states. So a stable system implies that each diagonal higher than $D(k)$ has at least one state where you switch a server off upon a departure, for a a certain $k \in \mathbb{N}$

Suppose there exists $k \in \mathbb{N}$, such that each diagonal higher than $D(k)$ contains one state where the server is switched off upon a customer departure. In order to prove that this strategy has a stable system we are going to compare the probability of going to a higher diagonal with the probability of going to a lower diagonal. Suppose that the probability of going to a lower diagonal goes to 1 then you cannot go to diagonal $\infty$. We are now going to look at the probability of going from $(0, k)$ to $(0, k+1)$ instead of $(k-1, 0)$ if the only state where you switch the server off is at $(k-1, 1)$ (the lowest possible point). Once we have proven that this probability goes to zero as $k$ goes to infinity we will see that the probability of going to the diagonal infinity is zero for all the strategies that satisfy our assumption. You can see that this probability is equal to $a(k)$ and we've already seen in lemma 3 that this goes to zero as k goes to infinity.

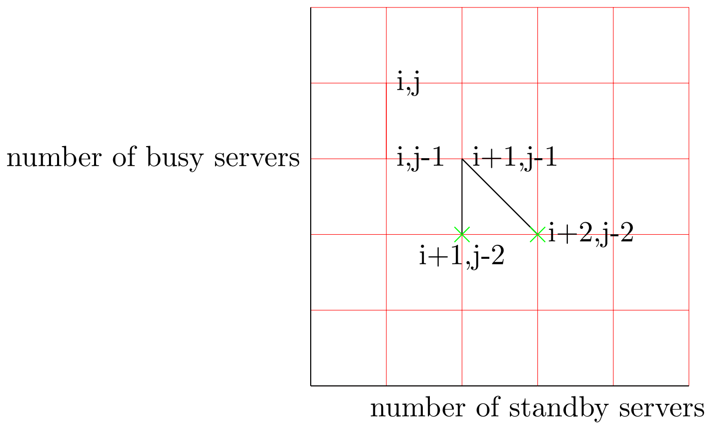
But we have only proven this if you start at the end of the diagonal and if you switch the server off on the other end that the probability of going to a lower diagonal goes to one. However it can be easily seen that if you start on any point on the diagonal that in order to go up to higher diagonal you first need to pass the state on the top, but on that state the probability of going down to a lower diagonal goes to 1. So in that case you also go down to a lower diagonal with probability 1. And in order to go all the way down you first need to pass all other states so if the state where you switch the server off was somewhere in between then you would also go down. So we have now proven that the probability of going down goes to 1 if there is at least one state where you switch the server off. This means that the probability of shooting away to infinity is 0. For any state there is a positive probability to go to any other reachable node in a finite number of steps. And since the probability of going to infinity is 0 you must go to any reachable node with probability 1.

$\square$

## 6 Strategies on a diagonal

Suppose you are at point $(i, j)$. At this moment you are in the triangle $T(i+j)$. Like previously stated the moment you leave the triangle is independent of the strategy that you choose. The place where you leave the triangle is also independent of the strategy and you will always leave the triangle at point $(0, i+j)$.
The strategy that you choose only has an influence on the cost that you make in that triangle. So clearly you want to choose the strategy that minimizes the expected cost for getting out of that triangle. If you are point $(i, j)$ and a customer leaves the system then you have two choices. You can either idle the server or shut it down. The optimal choice is to go to the point that has the lowest expected cost to leave the triangle $T(i+j)$. You must not forget to take into account that you have to pay $K^{off}$ to shut down the server.
**Theorem 1.** *If it is optimal to switch the server off at $(i, j)$ and $(j > 0)$ then it's optimal to switch the server off at $(i+1, j-1)$.*

*Proof.* Suppose that you shut down the server at $(i, j)$. That means that the expected cost to leave the triangle $T(i+j)$ are lower if you go to the point $(i, j-1)$ then if you would go to the point $(i+1, j-1)$. Now suppose that you are at point $(i+1, j-1)$. This is the point on the same diagonal as $(i, j)$, but one node below it. You want to know if you need to shut down the server or not when a customer leaves the system and you are at $(i+1, j-1)$. You already know that the expected cost to leave $T(i+j)$ at point $(i, j-1)$ plus $K^{off}$ are lower than the expected cost to leave $T(i+j)$ if you go to $(i+1, j-1)$.

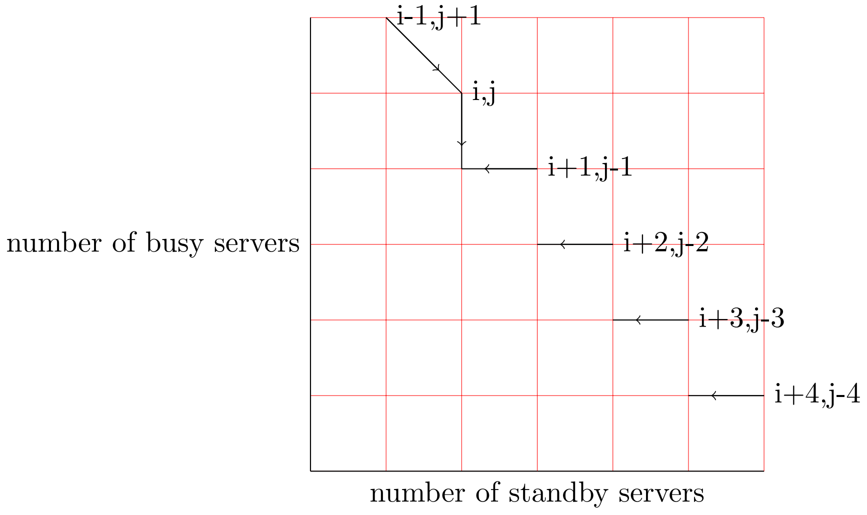


We need to know if the expected cost to leave the triangle are lower if you go to $(i+1, j-2)$ than if you go to $(i+2, j-2)$. The strategy that minimizes the expected cost to leave $T(i+j)$ once you are at point $(i+1, j-2)$ is bounded above by the expected cost of any other strategy you choose at $(i+1, j-2)$. If you have a strategy for the point $(i+1, j-2)$ that is $K^{off}$ cheaper than any strategy at point $(i+2, j-2)$ than it's beneficial to go to $(i+1, j-2)$ instead of $(i+2, j-2)$.
Now at point $(i+1, j-2)$ choose the following strategy: when a customer leaves the system you idle the server until time t. That means that you will stay on diagonal $D(i+j-1)$. We will show that this strategy is $K^{off}$ cheaper then any strategy you can choose if you went to $(i+2, j-2)$. For any strategy at $(i+2, j-2)$ there must be a moment that you go down to a lower diagonal or that you are at point $(i+1, j-1)$. Let's call the moment that this happens time t. Now we are going to look at the strategy that we have chosen at point $(i+1, j-1)$ until time t. There are two possibility's either at time t you are at the same point as if you would have started at point $(i+2, j-2)$ or you are at point $(i, j-1)$. If you started at $(i+2, j-2)$ you have one more server standby than if you started at $(i+1, j-2)$ at least until time t. So until time t you have paid more if you started at point $(i+2, j-2)$ than if you started at point

$(i+1, j-2)$.
If are at the same point at time t then you have paid more if you started at point $(i+2, j-2)$ then if you started at point $(i+1, j-2)$ on top of that you needed to pay $K^{off}$ to shut down the server. The expected cost are equal after time t since you can choose the same strategy. But since you paid more than $K^{off}$ more if you started at point $(i+2, j-2)$ then if you started at point $(i+1, j-2)$ the expected cost for $(i+1, j-2)$ must be lower then at point $(i+2, j-2)$. If you are at $(i, j-1)$ at time t then the expected cost if you started at $(i+1, j-1)$ must also be $K^{off}$ lower since the expected cost at $(i, j-1)$ are $K^{off}$ lower then at $(i+1, j-1)$ and you've paid less until the moment t. So in both cases the expected cost at $(i+1, j-1)$ is more then $K^{off}$ lower then at $(i+2, j-2)$. So if you shut down the server at $(i, j)$, because the expected cost at $(i, j-1)$ are $K^{off}$ lower than it's also cheaper to shut down the server at $(i+1, j-1)$.

□

You can use this lemma repeatedly to show that if it's cheaper to shut down the server at $(i, j)$ it's cheaper to shut down at any point on $D(i+j)$ below $(i, j)$. If it's cheaper to shut down the server at $(i, j)$ then it's cheaper to shut down a server that's on standby if you are at $(i+1, j-1)$ since the expected cost to leave $T(i+j)$ are more than $K^{off}$ lower at $(i, j-1)$ than at $(i+1, j-1)$. So there $i+j+1$ possible strategies on $D(i+j)$. Either you put everything on standby or there is a point where it's cheaper to shut down the server and below that point you shut down a idle server. See for example this picture were you shut down the server at $(i, j)$



# 7 Comparing strategies

Suppose you are in a triangle $T(k)$ then an optimal strategy is a strategy that has the lowest expected cost for getting out of that given triangle. In this chapter we will show a way to calculate the expected cost for getting out of a given triangle. This way we can compare strategies and choose the optimal one.
Suppose we are given a strategy in triangle $T(k)$ and a starting node $(i, j)$. You already know the probability of going up and down. But for the expected total cost we are interested in the expected cost you make before you are in the next state given that you go up or given that you go down. You can easily see that if you are on the y-axis and you go up that your expected cost is equal to $K^{on}$. Define Y as the time that a customer leaves the system and X as the time that a customer enters the system.

$$E(\text{cost}|(i,j) \to (i-1, j+1)) = c * i * E(\text{time in } (i,j)|(i,j) \to (i-1, j+1)).$$

We are now going to calculate the cumulative distribution function.

$$\begin{aligned}
P(X \leq t|X < Y) &= 1 - P(X \geq t|X < Y) \\
&= 1 - \frac{P(X < Y|X \geq t)P(X \geq t)}{P(X < Y)} \\
&= 1 - \frac{(1 - P(X \geq Y|X \geq t))P(X \geq t)}{P(X < Y)} \\
&= 1 - \frac{(1 - (P(X \geq Y|Y < t, X \geq t)P(Y < t) + P(Y \geq t)P(X > Y|X, Y \geq t))P(X \geq t)}{P(X < Y)} \\
&= 1 - \frac{(1 - (1 - e^{-\lambda t} + P(Y \geq t)P(X > Y|X, Y \geq t))(e^{-j*\mu t})}{\frac{j*\mu}{j*\mu+\lambda}} \\
&= 1 - \frac{(1 - (1 - e^{-\lambda t} + e^{-\lambda t}\frac{\lambda}{j*\mu+\lambda}))(e^{-j*\mu t})}{\frac{j*\mu}{j*\mu+\lambda}} \\
&= 1 - \frac{(e^{-\lambda t} - e^{-\lambda t}\frac{\lambda}{j*\mu+\lambda})(e^{-j*\mu t})}{\frac{j*\mu}{j*\mu+\lambda}} \\
&= 1 - \frac{e^{-\lambda t}(1 - \frac{\lambda}{j*\mu+\lambda})(e^{-j*\mu t})}{\frac{j*\mu}{j*\mu+\lambda}} \\
&= 1 - \frac{e^{-\lambda t}(\frac{j*\mu}{j*\mu+\lambda})(e^{-j*\mu t})}{\frac{j*\mu}{j*\mu+\lambda}} \\
&= 1 - e^{-(\lambda+j*\mu)t}.
\end{aligned}$$

We can now calculate the probability density function and with that the expected value of the costs given that a customer leaves the system

$$\begin{aligned} c*i*E(\text{time in }(i,j)|(i,j)\to(i-1,j+1)) &= c(i)*E(X|X<Y) \\ &= c*i*\int_0^\infty t*f(t|X<Y)dt \\ &= c*i*\int_0^\infty t(\mu*j+\lambda)e^{-(\lambda+j*\mu)t}dt \\ &= \frac{c*i}{\mu*j+\lambda}. \end{aligned}$$

We can do the same calculation to calculate that $E(Y|Y<X)=\frac{1}{\mu*j+\lambda}$. This means that the time that you stay in a state is independent of whether a customer enters or leaves the system. We now know the expected time you stay in each state, so we can calculate the expected cost you make before entering a new state.

$$\text{cost before entering a new state} = \begin{cases} \frac{c*i}{\lambda+j\mu}, & \text{if a customer enters and } i>0 \\ K^{on}, & \text{if a customer enters and } i=0 \\ \frac{c*i}{\lambda+j\mu}, & \text{if a customer leaves and you idle} \\ \frac{c*i}{\lambda+j\mu}+K^{off}, & \text{if a customer leaves and you shut down} \end{cases}$$

Since were only interested in the average cost optimal policy for controlling a webfarm we now switch to a discrete time markov process. With the cost as stated above, which depend on the beginning and end state. From now on we will look at this discrete time markov chain.

**Definition 6.** *$p_{i,j}(\uparrow)$ is the probability that the number of customers increases if you are in state $(i,j)$ and $p_{i,j}(\downarrow)$ is the probability that the number of customers decreases if you are in state $(i,j)$*

So these are the transition probabilities that will be used in the new markov chain

**Definition 7.** *$E_{(i,j),\uparrow}$ is the expected cost you make in state $(i,j)$ given that the number of customers increases and $E_{(i,j),\downarrow}$ is the expected cost you make in state $(i,j)$ given that the number of customers decreases*

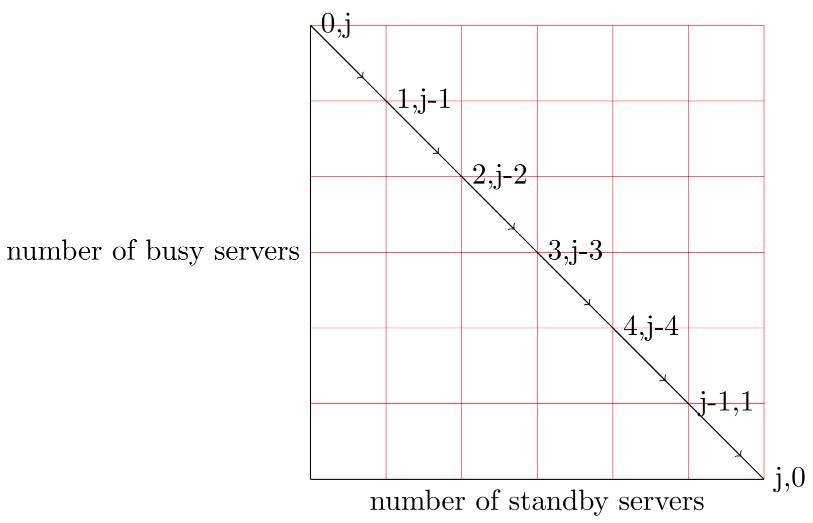
So these are the costs in this new markov chain.

We are now able to calculate the expected cost and the probability of a given path that leads out of the triangle by a path-wise comparison. But there are infinitely many paths that lead out of the triangle so we still need something else to calculate the expected cost for getting out of the triangle for a given strategy.

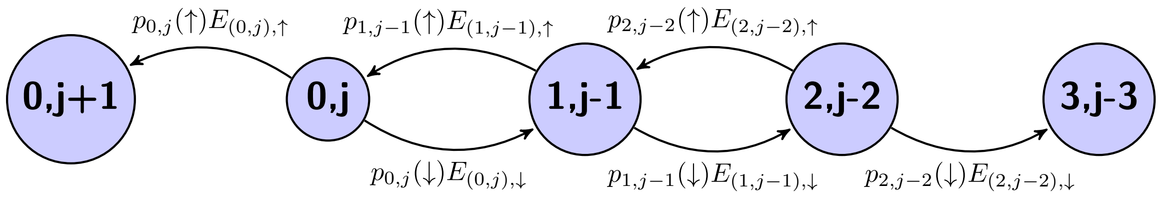
What we are going to do is calculate the expected cost to leave a triangle $T(j)$ and we will use this to calculate the expected cost to leave $T(j+1)$. Calculating the expected cost to leave $T(1)$ is straightforward so if we can calculate if or a next triangle were done. The concept to calculate for other triangles is relatively easy.

There are three possible strategies on a diagonal. You idle every server, you only idle the first server or you idle several servers.

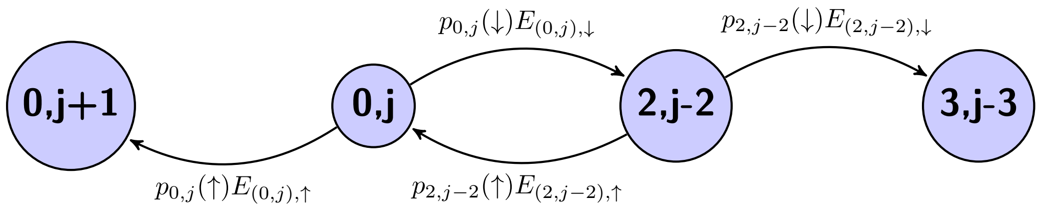
We first start with the simplest case. You are in triangle $T(j)$ and on diagonal $D(j)$ you always idle. We are now going to calculate the expected cost to leave $T(j)$.



Suppose you are on at point $(0, j)$. Then there are two possible things that can happen. Either you go to $(0, j + 1)$ or you go to $(2, j - 2)$. In both cases you can pass $(1, j - 1)$ and $(0, j)$ one or several times. We have this picture:


$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{1,j-1}(\uparrow)E_{(1,j-1),\uparrow}$
$p_{2,j-2}(\uparrow)E_{(2,j-2),\uparrow}$
0,j+1
0,j
1,j-1
2,j-2
3,j-3
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
$p_{1,j-1}(\downarrow)E_{(1,j-1),\downarrow}$
$p_{2,j-2}(\downarrow)E_{(2,j-2),\downarrow}$

We want to reduce it to this picture:


$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
$p_{2,j-2}(\downarrow)E_{(2,j-2),\downarrow}$
0,j+1
0,j
2,j-2
3,j-3
$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{2,j-2}(\uparrow)E_{(2,j-2),\uparrow}$

The new $p_{0,j}(\uparrow)$ is the probability that you go to $(0, j + 1)$ and not to $(2, j - 2)$. $p_{0,j}(\downarrow)$ is the same but then the other way around. $p_{2,j-2}(\uparrow)$ is the probability that you go to $(0, j)$ and not to $(3, j - 3)$. $p_{0,j}(\uparrow)$ is the same but then the other way around. The expected cost are updated in the same manner. So if we can calculate the costs and the probabilities we can eliminate the node $(1, j - 1)$.
In chapter 4 we've already shown how to calculate the probabilities in this case. The important thing is to count the number of loops you make. The costs can be calculated in a similar way. If you go from $(0, j)$ to $(0, j + 1)$ you know you have never visited $(2, j - 2)$ in the mean time, but you could have been to $(1, j - 1)$ one or several times.

**Definition 8.** *$L_{n\uparrow}$ is the path from $(0, j)$ to $(0, j + 1)$ where you visit $(1, j - 1)$ exactly $n$ times and you don't visit $(2, j - 2)$ before $(0, j + 1)$*
*$L_{n\downarrow}$ is the path from $(0, j)$ to $(0, j + 1)$ where you visit $(1, j - 1)$ exactly $n$ times and you don't visit $(0, j + 1)$ before $(2, j - 2)$*

$$\begin{aligned}
P(L_{n\uparrow}|\text{number of customers increases}) &= \frac{P(L_{n\uparrow}\text{ and the number of customers increases})}{P(\text{number of customers increases}} \\
&= \frac{P(L_{n\uparrow})}{P(\text{number of customers increases})} \\
&= \frac{p_{0,j}(\uparrow)(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^n}{\sum_{i=0}^{\infty} p_{0,j}(\uparrow)(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^i} \\
&= \frac{(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^n}{\sum_{i=0}^{\infty}(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^i}
\end{aligned}$$

If you know that you take the path $L_{n\uparrow}$ then the cost you make are equal to:
$E_{(0,j)\uparrow} + n * (E_{(0,j)\downarrow} + E_{(1,j-1)\uparrow})$

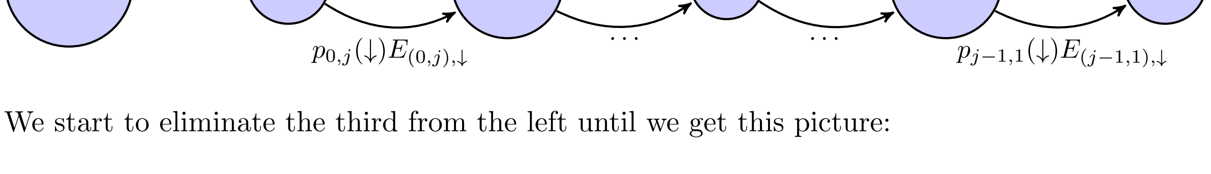
You can now calculate the new $E_{(0,j)(\uparrow)}$ since we know the probability for each of the possible paths and the costs invoked.

$$\begin{aligned}
E^{new}_{(0,j)(\uparrow)} &= \sum_{n=0}^{\infty} P(L_{n\uparrow}|\text{number of customers increases})(E_{(0,j)\uparrow} + n * (E_{(0,j)\downarrow} + E_{(1,j-1)\uparrow})) \\
&= \sum_{n=0}^{\infty} \frac{(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^n}{\sum_{i=0}^{\infty}(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^i}(E_{(0,j)\uparrow} + n * (E_{(0,j)\downarrow} + E_{(1,j-1)\uparrow})).
\end{aligned}$$

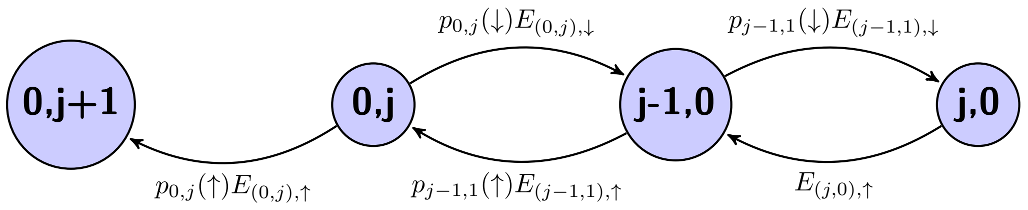
You can calculate $E^{new}_{(0,j)(\downarrow)}$, $E^{new}_{(2,j-2)(\uparrow)}$ and $E^{new}_{(2,j-2)(\downarrow)}$ exactly the same way as $E^{new}_{(0,j)(\uparrow)}$. You then get:

$$\begin{aligned}
E^{new}_{(0,j)(\downarrow)} &= \sum_{n=0}^{\infty} \frac{(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^n}{\sum_{i=0}^{\infty}(p_{0,j}(\downarrow)p_{1,j-1}(\uparrow))^i}(E_{(1,j-1)\downarrow} + E_{(0,j)\downarrow} + n * (E_{(0,j)\downarrow} + E_{(1,j-1)\uparrow})) \\
E^{new}_{(2,j-2)(\uparrow)} &= \sum_{n=0}^{\infty} \frac{(p_{1,j-1}(\downarrow)p_{2,j+2}(\uparrow))^n}{\sum_{i=0}^{\infty}(p_{1,j-1}(\downarrow)p_{2,j+2}(\uparrow))^i}(E_{(1,j-1)\uparrow} + E_{(2,j+2)\uparrow} + n * (E_{(1,j-1)\downarrow} + E_{(2,j+2)\uparrow})) \\
E^{new}_{(2,j-2)(\downarrow)} &= \sum_{n=0}^{\infty} \frac{(p_{1,j-1}(\downarrow)p_{2,j+2}(\uparrow))^n}{\sum_{i=0}^{\infty}(p_{1,j-1}(\downarrow)p_{2,j+2}(\uparrow))^i}(E_{(2,j-2)\downarrow} + n * (E_{(1,j-1)\downarrow} + E_{(2,j+2)\uparrow}))
\end{aligned}$$

We have now found a way to reduce to first picture to the second picture if you start at $(0,j)$ or $(2,j-2)$ by eliminating the middle node. We can repeat this process of elimination until the last node is $(j,0)$. So we start out with this picture:

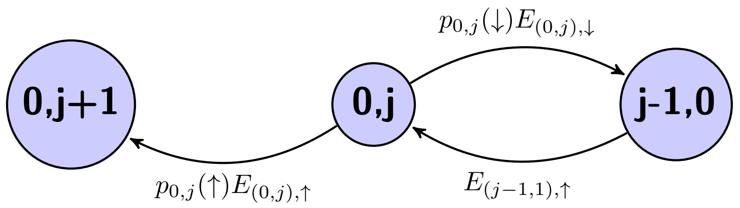

$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{1,j-1}(\uparrow)E_{(1,j-1),\uparrow}$
. . .
. . .
$E_{(j,0),\uparrow}$
0,j+1
0,j
1,j-1
. . .
j-1,1
j,0
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
. . .
. . .
$p_{j-1,1}(\downarrow)E_{(j-1,1),\downarrow}$

We start to eliminate the third from the left until we get this picture:



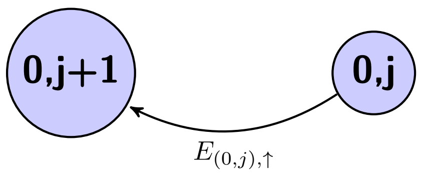
0,j+1
0,j
j-1,0
j,0
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
$p_{j-1,1}(\downarrow)E_{(j-1,1),\downarrow}$
$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{j-1,1}(\uparrow)E_{(j-1,1),\uparrow}$
$E_{(j,0),\uparrow}$

We can continue reducing the picture. If you would start at $(j-1, 0)$ you can go to $(j, 0)$ one or several times but you'll always go to $(0, j)$. You can easily see that the probability that you visit $(j, 0)$ n times before going to $(0, j)$ is equal to $p_{j-1,1}(\uparrow)p_{j-1,1}(\downarrow)^n$. We can now update the picture:

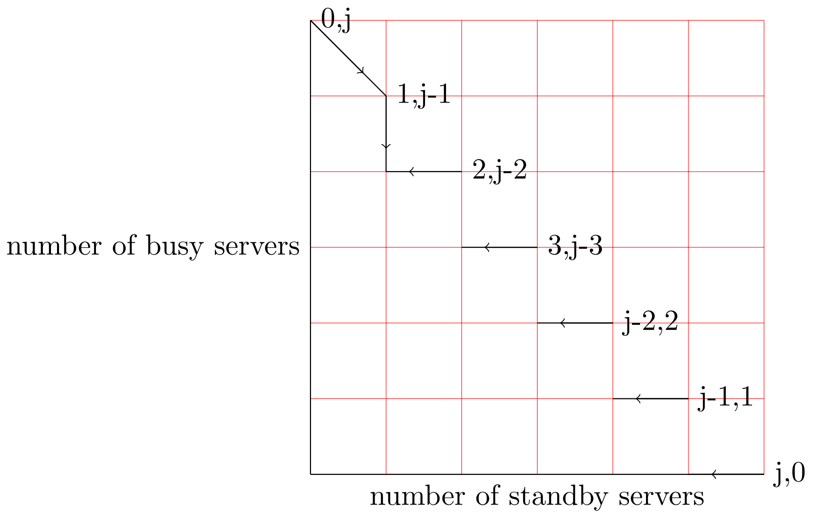


0,j+1
0,j
j-1,0
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$E_{(j-1,1),\uparrow}$

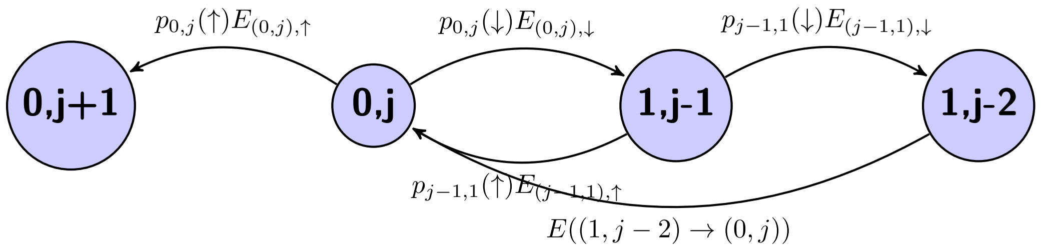
We can repeat the process one last time to get:



We now know the expected cost to get from $(0, j + 1)$ if you start at $(0, j)$ if you idle every server. Of course we want to know the expected cost if you start on any other point on $D(j)$. We can calculate this with the help of our new found knowledge. If you start doing the process backwards you can compute the expected cost of the other nodes. If you don't look at the last picture but the one before that you can easily see how the calculate the expected cost to get from $(j - 1, 0)$ to $(0, j)$. The expected cost is just $E_{(j-1,1),\uparrow}$. If you add this to the value we already have you get the expected cost to get to $(0, j + 1)$. You can repeat this process to calculate the expected cost for any of the nodes on $D(j)$ by putting the nodes back one at the time. We assumed that we already knew the expected cost to get from any node in $T(j - 1)$ to $(0, j)$. We can just add the expected cost of getting from $(0, j)$ to $(0, j + 1)$ to get the value to leave $T(j)$. We now know how to calculate the expected cost to leave a triangle if you idle every server and you already knew the expected cost of the previous triangle.
We started with the simplest case were you idle every server on the diagonal. We will see that for the other cases you can calculate the expected cosy in almost the same way as before. Now suppose you start on the strategy on diagonal $D(j)$ looks like this:

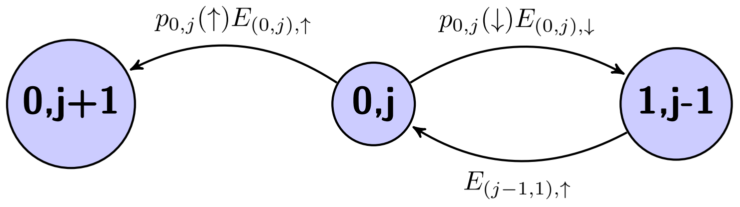


So you idle on the y-axis and you shut down at the $(x = 1) - axis$. Now suppose you start at $(0, j)$. The scenario will then look like this.



$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
$p_{j-1,1}(\downarrow)E_{(j-1,1),\downarrow}$
0,j+1
0,j
1,j-1
1,j-2
$p_{j-1,1}(\uparrow)E_{(j-1,1),\uparrow}$
$E((1, j-2) \to (0, j))$

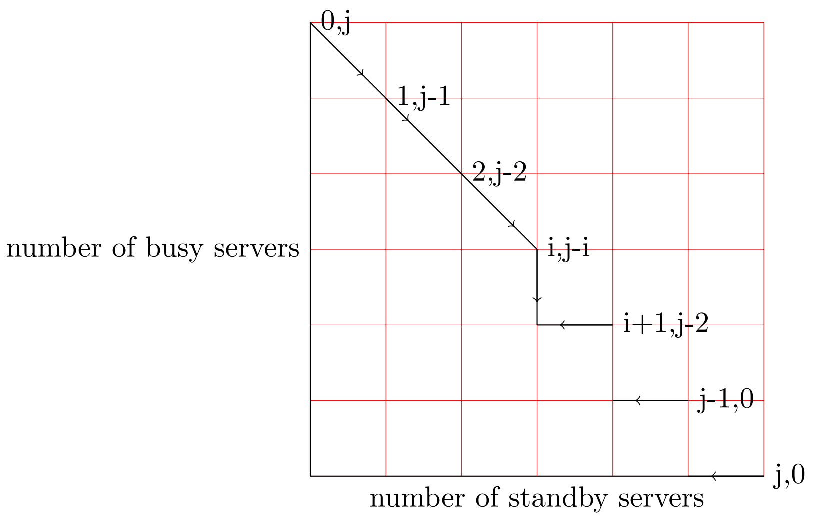
By assumption we already know the expected cost to get from $(1, j - 2)$ to get to $(0, j)$ so we just have the expected cost to get from $(1, j - 2)$ to $(0, j)$. But this picture can be reduced even more. You can calculate the expected cost to get from $(1, j - 1)$ to $(0, j)$. There are two possibilities either you directly go back to $(0, j)$ or you go via $(1, j - 2)$. You already know the probabilities for both cases and the expected cost for both cases. So you get this picture:



$p_{0,j}(\uparrow)E_{(0,j),\uparrow}$
$p_{0,j}(\downarrow)E_{(0,j),\downarrow}$
0,j+1
0,j
1,j-1
$E_{(j-1,1),\uparrow}$

This is a picture we've seen before so we already know how to get the expected cost to get to $(0, j+1)$ from this. We can calculate the expected cost for the other nodes by moving backwards again. We just need to know what to do with nodes were you immediately shut down a server. This is just as easy once you know the expected cost to get out of the triangle at $(1, j - 1)$. If you are $(2, j - 2)$ you immediately shut down a server and that will cost you $K^{off}$. You now are at $T(j - 1)$ and you already know how much it will you cost from there. So once again we can calculate the expected cost to leave $T(J)$ if we know the expected cost to leave $T(j - 1)$.

We will move to the last scenario. This scenario looks like this:



You idle every server until node $(i, j-i)$ and then you shut down the server. This case is just a combination of the previous to cases. You can just eliminate nodes in the same way as in the first case and then you've reduced it to the second case. In all three cases you can calculate the expected cost to leave the triangle if you know the expected cost of the previous triangle.
We know how to calculate the expected cost for $T(1)$ and we now know how to calculate the expected cost to leave triangle $T(j)$ if we know how to calculate the expected cost to leave $T(j-1)$, so we can calculate the expected cost for any larger triangle by induction.
It will be necessary to know the amount computations in the later chapters. To eliminate a node you have to do at most four times a fixed number of calculations. And if you are at diagonal $D(j)$ you have to remove at most j nodes. After that you have to do a fixed number of calculations for each node in $T(j)$ to calculate the expected cost to leave the triangle. This updating will take up the most time since there are $\frac{1}{2}j^2$ nodes. However to know how much it will cost to leave triangle $T(j)$ on $D(j)$ you only need to know the costs to leave triangle $T(j-1)$ at $D(j-1)$. In the future we will only need to know this. And that only takes $j$ times a fixed number number of calculations.
Also note that we are now able to construct a partial optimal solution from the ground up.

## 8 Strategies on the y-axis

Suppose you are in a state on the y-axis and a customer leaves the system. You want to decide whether or not to put the server off or idle it. Then putting the server off is the better strategy if for every strategy that idles the server there is a strategy that puts the server off that has a lower expected cost.
Now we are first going to look at the possible strategy when you are at state $(0,1)$. You can either go to $(0,0)$ by putting the server off or you can go to $(1,0)$ by putting the server on standby. You only go away from these states when a new customer enters the system and then you go back to state $(0,1)$. We can now look at the expected cost for both scenario's. If you put the server off you have to spent $K^{on}+K^{off}$ until you reach $(0,1)$. If you put the server on standby then you have to pay for the time you stay in that state. The expected time you stay in $(1,0)$ is equal to the time you have to wait for a customer to arrive and thats equal to $\frac{1}{\lambda}$. So the expected cost if you put the server on standby is equal to $\frac{c(1)}{\lambda}$. So it's good strategy to put the server off if and only if $K^{on}+K^{off} \leq \frac{c(1)}{\lambda}$.

**Lemma 5.** *If $K^{on}+K^{off} > \frac{c(1)}{\lambda}$ then you idle the server on the y-axis in the optimal strategy*

*Proof.* Suppose that $K^{on}+K^{off} > \frac{c(1)}{\lambda}$ then you put the server on standby at $(0,1)$. Now assume that in the optimal strategy you put the server on standby until at least state $(0,n)$. Now we are going to look at what we are going to do at state $(0,n+1)$. If you put the server of when a customer leaves the system you go to state $(0,n)$. When the next customer leaves the system you will choose to go to $(1,n-1)$ since in the optimal strategy you will put the server on standby at $(0,n)$. If a customer enters the system at $(0,n)$ you go back to $(0,n+1)$.
Now we are going to compare this strategy with a strategy if you idle the server at $(0,n+1)$. If you put the server down there you go to $(1,n)$. When the next customer leaves the system you will choose to go to $(1,n-1)$. If a customer enters the system at $(0,n)$ you go back to $(0,n+1)$. So whether a customer enters or leaves the system you are always on the same spot in both strategies after one step. We are now going to compare the expected cost until they meet.
If you choose to put the server off at $(0,n+1)$ then you go to state $(0,n)$ that will cost you $K^{off}$. If you go to state $(1,n-1)$ you have only paid $K^{off}$. If you go back to state $(0,n+1)$ you have pay $K^{on}$. The probability that you go back to $(0,n+1)$ is equal to $\frac{\lambda}{\lambda+n*\mu}$. So the expected cost for this strategy until they meet is equal to $\frac{(K^{on}+K^{off})\lambda}{\lambda+n\mu}+\frac{K^{off}n\mu}{\lambda+n\mu}$.
If you choose to put the server on standby at $(0,n+1)$ then you go to state $(1,n)$. Then you have to pay $c(1)$ times the time you stay in state $(1,n)$. The expected time you stay in state $(1,n)$ is equal to $\frac{1}{\lambda+n*\mu}$. If a customer leaves the system you also have to pay $K^{off}$ to shut down the server. So the expected cost for this strategy until they meet is equal to $\frac{c(1)}{\lambda+n\mu}+\frac{K^{off}n\mu}{\lambda+n\mu}$.

$$K^{on}+K^{off} > \frac{c(1)}{\lambda} \Rightarrow \frac{(K^{on}+K^{off})\lambda}{\lambda+n\mu}+\frac{K^{off}n\mu}{\lambda+n\mu} > \frac{c(1)}{\lambda+n\mu}+\frac{K^{off}n\mu}{\lambda+n\mu}.$$

This means that the expected cost are lower if you decide to put the server on standby at state $(0,n+1)$ before they meet. After they meet you can choose to do the optimal strategy. This means that idleling a server at $(0,n+1)$ is the better strategy. Since you know that you idle at $(0,1)$ and we have proven that if you idle the server at $(0,n)$ you idle the server at $(0,n+1)$. We now know that you idle the server at any point on the y-axis. □

**Lemma 6.** *If $K^{on}+K^{off} < \frac{c}{\lambda}$ then you shut down on the y-axis in the optimal strategy*

*Proof.* Suppose you are at $(0,n)$ and a customer leaves the system. We are going to compare the strategy of idling with the strategy of shutting down the server. If shutting down is cheaper it's enough to give a strategy that's cheaper than any strategy if you idle. So the strategy that we use is mimicking the strategy were you idle (obviously excluding the first step). The moment the two strategies meet is when one is on the y-axis and the other is on the (x=1)-axis and a customer enters the system. In the first step you have to pay $K^{off}$ extra if you shut down the server and in the last step you have to pay $K^{on}$ extra. Then the rest of the time before they meet you always have an extra idle server if you initially idled the server and for the rest you make the same costs. The time it will take for the strategies to meet must be at least the time it takes for a customer to enter the system and the expected time for that is $\frac{1}{\lambda}$. So the extra cost you make by idling is at least $\frac{c}{\lambda}$ on average. But we know that $\frac{c}{\lambda} > K^{on} + K^{off}$ this means the lower bound for costs you make by idling are larger then the costs extra costs you make for shutting down. This means you will shut down the on the y-axis. And this means you will shut down an idle server if you're not on the y-axis. This gives you a global strategy if $K^{on} + K^{off} < \frac{c}{\lambda}$ □

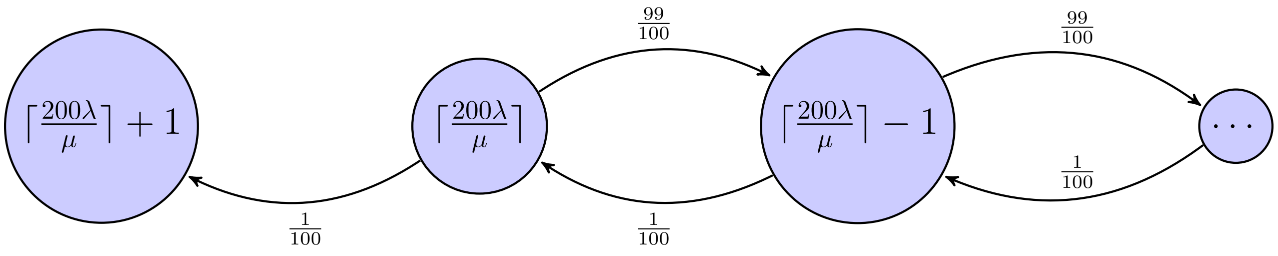
Since we already know what happens if $K^{on} + K^{off} < \frac{c}{\lambda}$ we will always assume from now on that $K^{on} + K^{off} > \frac{c}{\lambda}$

## 9 Bounding $a(j)$

For reasons that will become clear in the next chapters it's necessary to know how big $a(j)$ is for different $\mu$ and $\lambda$. For the computation time it's important that $1 - a(j)$ doesn't become much smaller if $\lambda$ increases. It will be shown that $1 - a(\frac{200\lambda}{\mu})$ is bounded from below.
We will give a bound on $1 - a(\frac{200\lambda}{\mu})$ in three steps. First show that it's really likely that the amount of customers in the system is halved, then show that it's really likely that you get $\frac{\lambda}{\mu}$ in the system and finally that once you have that many in the system that's pretty likely to get an empty system.
**Definition 9.** *$b(j)$ is the probability that there are $j + 1$ customers in the system before there are $\lceil \frac{j}{2} \rceil$ customers in the system if you start with j customers.*

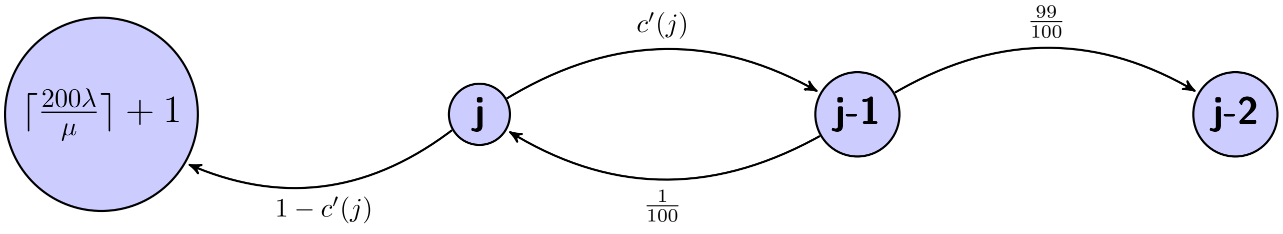
$b(j)$ is the probability that the amount of customers in your system has halved without ever having had more customers in the system. We will need some information about $b(j)$ in order to give a lower bound for $1 - a(j)$.
**Lemma 7.** $b(\lceil \frac{200\lambda}{\mu} \rceil) = b(j) < \frac{1}{99}$

*Proof.* We are only interested in an upper bound for $b(j)$ so we will change some probabilities to make calculations easier. We say that the probability that a customer leaves the system before another enters the system is $\frac{99}{100}$ and that the probability that a customer enters the system before another leaves the system is equal to $\frac{1}{100}$. In reality the probability that customer leaves the system before another enters is larger, since we are looking at $\lceil \frac{\lceil \frac{200\lambda}{\mu} \rceil}{2} \rceil$ and above. So we get an upper bound for $b(j)$.


$$\begin{tikzcd}
\lceil \frac{200\lambda}{\mu} \rceil + 1 & \lceil \frac{200\lambda}{\mu} \rceil \arrow[l, bend left, "\frac{1}{100}"] \arrow[r, bend left, "\frac{99}{100}"] & \lceil \frac{200\lambda}{\mu} \rceil - 1 \arrow[l, bend left, "\frac{1}{100}"] \arrow[r, bend left, "\frac{99}{100}"] & \cdots \arrow[l, bend left, "\frac{1}{100}"]
\end{tikzcd}$$

**Definition 10.** *$c(i)$ is the probability that there are $i - 1$ customers in the system before there are $\lceil \frac{200\lambda}{\mu} \rceil + 1$ in the system*

We can calculate $c(i)$ for the model were every probability is $\frac{99}{100}$. To make clear that the $c(i)$ that we calculate are an upper bound we will refer to them as $c'(i)$. Clearly $c'(\lceil \frac{200\lambda}{\mu} \rceil) = 0.99$. We can now make the following picture:


$$\begin{tikzcd}
\lceil \frac{200\lambda}{\mu} \rceil + 1 & j \arrow[l, bend left, "1 - c'(j)"] \arrow[r, bend left, "c'(j)"] & j-1 \arrow[l, bend left, "\frac{1}{100}"] \arrow[r, bend left, "\frac{99}{100}"] & j-2
\end{tikzcd}$$

You can now see that we can calculate $c'(i-1)$ using the geometric series we have used before.

$$\begin{aligned} c'(i-1) &= \frac{99}{100} + \frac{1}{100}c(i)c(i-1) \\ &= \frac{99}{100-c(i)}. \end{aligned}$$

Now let $d(i) = c'(\lceil\frac{200\lambda}{\mu}\rceil - i)$. This means that $d(0) = c(j)$ and $d(1) = c(j-1)$ and so on. Then you get:

$$c'(j+1) = \frac{99}{100-d(i)}.$$

You can easily proof by induction that this gives the following formula for $d(i)$:

$$d(i) = 1 - \frac{98}{99^{2+i}-1}.$$

Now we can give the desired bound for $b(j)$.

$$\begin{aligned} 1 - b(\lceil\frac{200\lambda}{\mu}\rceil) &= \prod_{i=\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil+1}^{\lceil\frac{200\lambda}{\mu}\rceil} c(i) \\ &> \prod_{i=\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil+1}^{\lceil\frac{200\lambda}{\mu}\rceil} c'(i) \\ &= \prod_{i=0}^{\lceil\frac{200\lambda}{\mu}\rceil-\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil-1} d(j) \\ &> \prod_{i=0}^{\infty} d(i) \\ &= \prod_{i=0}^{\infty} 1 - \frac{98}{99^{2+i}-1} \\ &= \frac{98}{99}. \end{aligned}$$

□

We now want to know what the probability is that you get $\lceil\frac{\lambda}{\mu}\rceil$ customers in the system without getting more then $\lceil\frac{200\lambda}{\mu}\rceil$. If you have more then $\lceil\frac{\lambda}{\mu}\rceil$ in your system then it's more likely that a customers will leave than that a customer will enter the system. If you start at $\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil$ then you are further away from $\lceil\frac{200\lambda}{\mu}\rceil$ then from $\lceil\frac{\lambda}{\mu}\rceil$. This means that there are more states between $\lceil\frac{200\lambda}{\mu}\rceil$ and $\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil$ then between $\lceil\frac{\lambda}{\mu}\rceil$ and $\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil$. The combination of these two gives that it's more likely to get to $\frac{\lambda}{\mu}$ before $\lceil\frac{200\lambda}{\mu}\rceil + 1$ if you start at $\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil$. So we can safely say that the probability to get $\lceil\frac{\lambda}{\mu}\rceil$ customers before you have $\lceil\frac{200\lambda}{\mu}\rceil$ customers if you start with $\lceil\frac{\lceil\frac{200\lambda}{\mu}\rceil}{2}\rceil$ customers is at least a $\frac{1}{2}$. So we can give a lower bound to get to $\lceil\frac{\lambda}{\mu}\rceil$ before going to $\lceil\frac{200\lambda}{\mu}\rceil + 1$ if you start at $\lceil\frac{200\lambda}{\mu}\rceil$

$$\frac{98}{99}\left(\frac{1}{2} + \frac{\frac{1}{2}\frac{1}{2}\frac{98}{99}}{1 - \frac{1}{2}\frac{98}{99}}\right) = \frac{49}{50}.$$

Suppose you start with $\lceil\frac{200\lambda}{\mu}\rceil$ customers in the system. The average amount of times you visit $\lceil\frac{200\lambda}{\mu}\rceil$ before having $\lceil\frac{\lambda}{\mu}\rceil$ customers in the system is bounded above by $1 + \frac{\frac{1}{50}}{1-\frac{1}{50}}$.

**Lemma 8.** $1 - a(\lceil\frac{200\lambda}{\mu}\rceil) \geq \epsilon > 0.$

*Proof.* Suppose you have an empty system. How many times will the system get empty before there are $\frac{\lambda}{\mu}$ customers in the system. $a(j)$ is increasing in j as long there are less then $\frac{\lambda}{\mu}$ customers in the system. The probability that the system gets empty again before $\frac{\lambda}{\mu}$ is reached is:

$$\begin{aligned}\prod_{j=1}^{\lceil\frac{\lambda}{\mu}\rceil-1} a(j) &> \prod_{j=1}^{\lceil\frac{\lambda}{\mu}\rceil-1} a(1)\\ &= \frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}}\end{aligned}$$

We will only need to check what happens when we change $\lambda$ since if $\mu$ goes to zero that's the same if $\lambda$ goes to infinity and vice versa.

$$\begin{aligned}\frac{\delta}{\delta\lambda}\frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}} &= \log(\frac{\lambda}{\lambda+\mu})\frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}}\frac{\delta}{\delta\lambda}\frac{\lambda}{\lambda+\mu}\\ &\log(\frac{\lambda}{\lambda+\mu})\frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}}\frac{\mu}{(\mu+\lambda)^2}\\ &< 0.\end{aligned}$$

$$\begin{aligned}\frac{\delta}{\delta\lambda}\frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}} < 0 &\Rightarrow \frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}} > \lim_{\lambda\to\infty}\frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}}\\ &\Rightarrow \frac{\lambda}{\lambda+\mu}^{\frac{\lambda}{\mu}} > \frac{1}{e}\\ &\Rightarrow \prod_{j=1}^{\frac{\lambda}{\mu}-1} a(j) > \frac{1}{e}.\end{aligned}$$

Suppose you start with an empty system. The average amount of times you have an empty system. before having $\lceil\frac{\lambda}{\mu}\rceil$ customers in the system is bounded above by $1 + \frac{\frac{1}{e}}{1-\frac{1}{e}}$.

The stationary distribution for the $M|M|\infty$ queue is a Poisson distribution.

$$\pi_k = \frac{\frac{\lambda}{\mu}^k e^{\frac{-\lambda}{\mu}}}{k!} \ , \ k \geq 0.$$

With the stationary distribution you know the ratios between the average amount of time you stay in each state. You can also calculate the ratio between the average amount of times you are in each state per time unit. You can do this by dividing the ratio for the average amount of time you stay in each state with the expected time you stay in that given state.

$$\frac{\frac{\lambda}{\mu}^k e^{\frac{-\lambda}{\mu}}}{k!}(\mu k + \lambda) \ , \ k \geq 0.$$

We want compare $\pi_{\frac{200\lambda}{\mu}}$ with $\pi_0$.

$$\begin{aligned}
\pi_{\frac{200\lambda}{\mu}} &= \frac{\frac{\lambda}{\mu}^{\frac{200\lambda}{\mu}} e^{\frac{-\lambda}{\mu}}}{\frac{200\lambda}{\mu}!} \\
&= \pi_0 \frac{\frac{\lambda}{\mu}^{\frac{200\lambda}{\mu}}}{\frac{200\lambda}{\mu}!} \\
&= \pi_0 \frac{x^{200x}}{200x!} \\
&< \pi_0 \frac{x^{200x}}{\frac{200x}{e}^{200x}} \\
&= \pi_0 \frac{1}{\frac{200}{e}^{200x}} \\
&< \pi_0 \frac{1}{2}^{200x} \\
&= \pi_0 \frac{1}{2}^{\frac{200\lambda}{\mu}}.
\end{aligned}$$

We used Stirlings approximation for the factorial. As you can see it's far more likely to see the system empty then that there are $\frac{200\lambda}{\mu}$ customers in the system, if $\frac{\lambda}{\mu} > 1$.

$$\begin{aligned}
\pi_{\frac{200\lambda}{\mu}}(\frac{200\lambda}{\mu}\mu + \lambda)100 > \pi_0(\lambda + \mu) &\Rightarrow \pi_0 \frac{1}{2}^{\frac{200\lambda}{\mu}}(\frac{200\lambda}{\mu}\mu + \lambda)100 > \pi_0(\lambda + \mu) \\
&\Rightarrow \frac{1}{2}^{\frac{200\lambda}{\mu}}(\frac{200\lambda}{\mu}\mu + \lambda)100 > (\lambda + \mu) \\
&\Rightarrow \frac{1}{2}^{\frac{200\lambda}{\mu}}(20100\lambda) > (\lambda + \mu) \\
&\Rightarrow \frac{1}{2}^{\frac{200\lambda}{\mu}} > \frac{\lambda + \mu}{20100\lambda} \\
&\Rightarrow \frac{1}{2}^{\frac{200\lambda}{\mu}} > \frac{1}{20100} \\
&\Rightarrow \frac{1}{2}^{200} > \frac{1}{20100}, \lambda > \mu.
\end{aligned}$$

So this means that $\frac{\pi_{\frac{200\lambda}{\mu}}}{\frac{200\lambda}{\mu}\mu+\lambda}100 < \frac{\pi_0}{\lambda+\mu}$ if $\lambda > \mu$. This means that the system is at least 100 times more empty then that there are $\frac{200\lambda}{\mu}$ in the system.
So now suppose you have $\frac{\lambda}{\mu}$ customers in the system and you want to know if it's more likely to get an empty system before there are $\frac{200\lambda}{\mu}$ customers in the system.
If the system gets empty then the system gets at most $1+\frac{\frac{1}{e}}{1-\frac{1}{e}}$ times empty on average before reaching $\frac{\lambda}{\mu}$ customers in the system and if the system has $\frac{200\lambda}{\mu}$ customers in the system you have $1+\frac{\frac{1}{50}}{1-\frac{1}{50}}$ times $\frac{200\lambda}{\mu}$ customers before having $\frac{\lambda}{\mu}$ customers in the system in the system. The difference between these to is less then a factor 2. So it's not true that once you get an empty system that gets empty a lot of times before returning to having $\frac{\lambda}{\mu}$ in the system (if you compare it to having $\frac{200\lambda}{\mu}$ customers in the system). And as we've seen it's 100 times more likely to have an empty system. If it's more likely that you have $\frac{200\lambda}{\mu}$ customers in the system before the system is empty then you would be more often in $\frac{200\lambda}{\mu}$ then that the system gets empty. This contradicts what we've already seen. This means that it's more likely to get an empty system once you have $\frac{\lambda}{\mu}$ in the system. And we've already seen that the probability to get $\frac{\lambda}{\mu}$ customers in the system is at least 98%. So we can conclude that $1 - a(\lceil\frac{200\lambda}{\mu}\rceil) \geq \epsilon > 0$.

$\square$

## 10 Strategies on the (x=1)-axis

Suppose you are in a state on the (x=1)-axis and a customer leaves the system. You want to decide whether or not to shut down the server. Then putting the server off is the better strategy if, for every strategy that idles the server there is a other strategy that puts the server off that has a lower expected cost. In this section it will be shown that there is a point on the (x=1)-axis where you shut down the server and if you get above that point you will also shut down the server. This will be shown for every possible combination of $\lambda$, $\mu$, $c(i)$, $K^{on}$ and $K^{off}$.

First we need to know the diagonal where there is at least one point where the server will be put off. Suppose you are on a diagonal where you put idle every node and you start at the x-axis (so no customers are in the system). You can now look at the expected time before you leave this diagonal.

**Lemma 9.** *If $i > max(\frac{(K^{on}+K^{off})\mu}{\epsilon * c}, \frac{\lceil 200\lambda \rceil}{\mu})$ then you shut down at least one server on $D(i)$*

*Proof.*

**Definition 11.** *$k(i)$ is the expected time to get i customers in the system if you start with an empty system*

So if you are on a diagonal were you idle all the time $k(i)$ is the time to get to the y-axis if you start on the x-axis. Now if you are on the y-axis two things can happen. You can either go up to higher diagonal before the system gets empty or the system gets empty. Remember that the probability that the system gets empty first is $1 - a(i)$. Now if you started on the x-axis the expected time to get to the y-axis is $k(i)$. Now with probability $1 - a(i)$ you get back to the x-axis before going to a higher diagonal. And with probability $(1 - a(i))^{10}$ you go to the x-axis at least 10 times before you go to the higher diagonal. Then you can easily see that the expected time before have $i + 1$ customers in the system is at least:

$$k(i)\sum_{n=0}^{\infty}(1 - a(i))^n = \frac{k(i)}{a(i)}$$

The expected time is more than this since we ignore the time it takes to go from the y-axis to the x-axis. But we will only need a lower bound so this is not important. We know that $a(i)$ goes to zero when $i$ increases. In chapter 9 we've seen that $a(\lceil \frac{200\lambda}{\mu} \rceil) > \epsilon$ and in chapter 5 we've seen how $a(i)$ will decrease if we've got an upper bound for $1 - a(i)$. Since $k(i)$ must be increasing the time to leave a diagonal goes to infinity if i increases. There will be a point such that the expected time to get to y-axis times the cost per time c is larger then $K^{on} + K^{off}$. That means the expected cost would have been lower if you were on a lower diagonal. If you would've been on this lower diagonal and you idle every server on that diagonal you would always have one less idle server. And since c times the expected time to get to y-axis is larger then $K^{on} + K^{off}$, you will have less cost. This means that if the property holds that the expected time to get to y-axis times the cost per time c is larger then $K^{on} + K^{off}$ you will put the server of if you are on the $(y = 1)$-axis. So we only need to calculate the point were you shut down a least one server. Now let i be at least then $\frac{\lceil 200\lambda \rceil}{\mu} + 1$.

$$\begin{aligned}
c\frac{k(i)}{a(i)} &\geq c\frac{k(1)}{a(i)} \\
&= \frac{c}{\lambda}\frac{1}{a(i)} \\
&= \frac{c}{\lambda}\frac{1}{\frac{1}{\frac{i\lambda}{\mu}(1-a(i-1))+1}} \\
&\geq \frac{c}{\lambda}\frac{1}{\frac{1}{\frac{i\lambda}{\mu}\epsilon+1}} \\
&= \frac{i*c}{\mu}\epsilon + \frac{c}{\lambda} \\
&\geq \frac{i*c}{\mu}\epsilon.
\end{aligned}$$

$$\frac{i*c}{\mu}\epsilon > K^{on} + K^{off} \Rightarrow i > \frac{(K^{on}+K^{off})\mu}{\epsilon * c}.$$

□

**Lemma 10.** *If $j > max\{\frac{(K^{on}+K^{off})\mu^2(\mu+\lambda)}{c\lambda^2\epsilon^2}, \frac{200\lambda}{\mu}, \frac{(K^{on}+K^{off})\mu}{\epsilon * c}\} + 2$ then you shut down the server on the $(x=1)$-axis of $D(j)$*

*Proof.* Now choose a diagonal such that you put the servers off on at least one place on the diagonal. This means that you put the server off on at least the $(y=1)$-axis. Now suppose you are in a state on the $(x=1)$-axis and a customer leaves the system. We are going to compare a strategy were you put the server off with all the possible strategies were you idle the server. If you idle the server first then either there is a moment were you go to a lower diagonal or there is a moment were you are on the y-axis. In the case that you shut down the server you use the following strategy: you idle the server when a customer leaves the system until the moment that you meet the strategy were you initially idled. The strategies meet when the first one goes to a lower diagonal or when it reaches the y-axis. From that point on you can choose to do the optimal strategy. So if putting the server off at the first step and then always idle every server is cheaper then first idling the server we know that it's the best strategy to put the server off, since a strategy were you put the server off is better then any strategy were you initially idle the server.
You can easily compare the cost of the strategies. If you idle the server then you always have an extra idle server if you compare it to the case were you first put server off. If you started on the j-th diagonal the expected time before the strategies meet is at least: $\frac{1}{\mu*(j-1)+\lambda}$. Since this is the expected time you stay on the node $(1, j-1)$ and $(2, j-1)$. So the extra cost you make by idling the server is at least: $\frac{c}{\mu*(j-1)+\lambda}$. If the strategies meet on the y-axis and not on the x-axis then you pay $K^{on}$ extra if you put the server off first plus the initial $K^{off}$ for shutting down the server. The probability that you meet on the y-axis and not on the x-axis is equal to $a(j-2)a(j-1)$. If you meet on the x-axis you've paid $K^{off}$ in both cases. This means that it's cheaper to put the server off if:

$$a(j-1)a(j-2)(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda}$$

Now we will show when this property holds. The implication below only hold if $j > \frac{200\lambda}{\mu}$ and $j > \frac{(K^{on}+K^{off})\mu}{\epsilon * c}$. This means that you shut down at least one server and you know a lower bound for $1 - a(j)$.

$$
\begin{aligned}
j - 2 > \frac{(K^{on} + K^{off})\mu^2(\mu+\lambda)}{c\lambda^2\epsilon^2} &\Rightarrow \\
\frac{\mu+\lambda}{(j-2)} < \frac{c\lambda^2\epsilon^2}{(K^{on}+K^{off})\mu^2} &\Rightarrow \\
\frac{\lambda}{(j-1)(j-2)} + \frac{\mu}{(j-2)} < \frac{c\lambda^2\epsilon^2}{(K^{on}+K^{off})\mu^2} &\Rightarrow \\
\frac{\mu*(j-1)+\lambda}{(j-1)(j-2)} < \frac{c\lambda^2\epsilon^2}{(K^{on}+K^{off})\mu^2} &\Rightarrow \\
\frac{\mu*(j-1)+\lambda}{(j-1)(j-2)} < \frac{c\lambda^2\epsilon^2}{(K^{on}+K^{off})\mu^2} &\Rightarrow \\
\frac{1}{\frac{(j-1)(j-2)\lambda^2}{\mu^2}\epsilon^2}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
\frac{1}{\frac{(j-1)(j-2)\lambda^2}{\mu^2}\epsilon^2}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
\frac{1}{\frac{(j-1)\mu}{\lambda}\epsilon}\frac{1}{\frac{(j-2)\mu}{\lambda}\epsilon}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
\frac{1}{\frac{(j-1)\mu}{\lambda}(1-a(j-2))}\frac{1}{\frac{(j-2)\mu}{\lambda}(1-a(j-2))}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
\frac{1}{\frac{(j-1)\mu}{\lambda}(1-a(j-2))+1}\frac{1}{\frac{(j-2)\mu}{\lambda}(1-a(j-2))+1}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
\frac{1}{\frac{(j-1)\mu}{\lambda}(1-a(j-2))+1}\frac{1}{\frac{(j-2)\mu}{\lambda}(1-a(j-2))+1}(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda} &\Rightarrow \\
a(j-1)a(j-2)(K^{on}+K^{off}) < \frac{c}{\mu*(j-1)+\lambda}&.
\end{aligned}
$$

□

## 11 The global optimal strategy

Now we can put everything together to construct the global optimal strategy.
**Theorem 2.** *If $\frac{c}{\lambda} > K^{on} + K^{off}$ the optimal strategy is to never have an idle server.*

*Proof.* This has already been proven in lemma 6 □

**Definition 12.** $(max\{\frac{(K^{on}+K^{off})\mu^2(\mu+\lambda)}{c\lambda^2}, \frac{\lambda}{\mu}, \frac{(K^{on}+K^{off})\mu}{c}\}) + 2 = d$
**Theorem 3.** *If $\frac{c}{\lambda} < K^{on} + K^{off}$ the optimal strategy is to*

- *Idle on every state on the y-axis*
- *Shut down the server on the $(x = 1)$-axis if there at least $k$ customers in the system*
- *If you switch off at $(i, j)$ you switch of at $(i + 1, j - 2)$*

Lemma 5 proves the first part, the second part is proven in lemma 10 and the third part is proven in theorem 1.
**Theorem 4.** *You can calculate the optimal strategy in $O(d^3)$.*

*Proof.* In chapter 6 we've seen all the possible strategies on a diagonal that you need to check. The amount of strategies that you need to check on $D(j)$ is at most $j + 1$. This means that below $D(d)$ there are less than $d^2$ possible strategies to compare. In chapter 7 we've seen that if you know the strategy on $D(j)$ you need to do at most $j+1$ times a fixed number of calculations (independent of $j$) to calculate the expected cost of a strategy on $D(j + 1)$. This means that each strategy will cost you at most $d$ times a fixed number of calculations (independent of $d$). So there are $d^2$ at most strategies that you need to check and each one will cost at most $d$ times a fixed number of calculations (independent of $d$). So you can find the optimal strategy in $O(d^3)$. □

## 12 Conclusion

We've shown that there exists an expected average cost optimal policy for controlling a webfarm that can be determined in polynomial time as a function of the given parameters. Furthermore we have derived an explicit upper bound ($d$) on the region where idling may be optimal.

## References

[1] Sandra C.C. van Wijk, *Creation of pooling in inventory and queueing. Eindhoven University of Technology*, PHD-thesis, pp. 123–136, Eindhoven, 2012.

[2] I.J.B.F. Adan, V.G. Kulkarni, A.C.C. van Wijk *Optimal control of a server farm. Eindhoven University of Technology*, Eindhoven, 2 October 2012 (preprint).

[3] A. Gandhi, V. Gupta, M. Harchol-Balter, M.A. Kozuch *Optimality Analysis of Energy-Performance Trade-off for Server Farm Management. Carnegie Mellon University*, Pittsburg, 23 January 2010.