On Optimal Policies for Energy-Aware Servers

V. J. Maccio[a], D. G. Down[a]

[a]*McMaster University, Hamilton, Ontario*

Abstract

As energy costs and energy used by server farms increase, so does the desire to implement energy-aware policies. Although under some metrics, optimal policies for single as well as multiple server systems are known, a number of metrics remain without sufficient knowledge of corresponding optimal policies. We describe and analyse a model to determine an optimal policy for on/off single server systems under a broad range of metrics that are based on expected response time, expected energy costs, and expected wear and tear costs. We leverage this model in the problem of routing jobs to one of two servers to show a range of non-trivial optimal routing probabilities and server configurations when energy concerns are a factor.

Keywords: Green computing, energy-aware server, queueing analysis, routing

1. Introduction

The relative as well as absolute energy consumed by servers have been steadily increasing in North America and has become a problem of considerable interest [4]. As systems grow and expand, energy concerns become a major factor for server farm managers from both environmental and economic viewpoints. However, the task of creating feasible optimal or near optimal policies is a daunting problem due to the sheer complexity these systems exhibit. Even for single server systems, when energy is a factor, optimal policies remain unknown for a number of metrics considered in the literature. We focus on developing a model in the context of and using tools and results from queueing theory, that allows one to determine an optimal policy for a single server system under a broad range of metrics. In particular, we consider cost functions constructed from the expected response time of a job in the system ($\mathbb{E}[R]$), the expected energy consumed by the system ($\mathbb{E}[E]$), and the steady state rate that the server cycles between two states, i.e. turning off and on ($\mathbb{E}[C]$), where the expected cycle rate can be thought as the expected wear and tear on the server. This paper extends the work [13].

By now, the field of green computing has a rich literature. We will focus our discussion on work which is concerned with moving servers to different energy states to increase (or decrease) performance. For example, the work in [2, 14, 18] looked at determining the optimal configuration of a server farm when the job sizes are known at arrival, and the decision to turn servers off or keep them on is made at discrete time intervals. This is then formulated as an optimization problem and solved. The article [2] was the first to appear and accounts for wear and tear cost on the servers by allowing for a term similar to $\mathbb{E}[C]$ in the cost function. The article [11] added the variation that jobs can be routed to different geographical locations where energy costs may differ. The work in [14] took a different viewpoint where customers pay a cost, based on a function of the response time of that job. The work of [15] looked at a similar problem where jobs are routed to separate on/off queues, and the problem was solved using Markov Decision Processes (MDPs). In this field the issue of speed scaling also arises, where one can use more energy to improve the performance (response time) of the system. This is examined in [18, 19]. While these frameworks use stochastic models and results, our model puts much heavier emphasis on analysing these systems in a queueing theory context.

When analysing these systems as continuous time Markov chains (CTMCs), they can often be viewed as or reduced to some form of a vacation model, where a vacation is interpreted as the server being off. Many of these models are considered in [1, 5, 17]. It will be seen that our work has elements that are of a similar spirit, i.e. the decomposition of system metrics. The work of [20] looked at specialised vacation models capturing multi-server behaviours, although this is done for specific policies which in general do not capture the optimal policy. To the best of our knowledge, the vacation model which most closely relates to the model analysed in this work is that described in [3]. In general the vacation model presented there is not optimal under the previously mentioned metrics. However, due to a higher coupling to the arrival process, our model allows one to describe the optimal policy.

While all previously mentioned work is related to this paper, the research which is most similar is [6, 7, 8, 9, 16]. They model these systems as CTMCs in both the single server and multi-server settings and analyse them under the specific metric which they refer to as the Energy Response Product (ERP), which unsurprisingly is defined to be the product of the expected response time, and expected energy consumed. With several new techniques (such as the RRR technique described in [6]) and observations, they are often able to arrive at closed form expressions even for multi-server systems, albeit under specific policies. Furthermore, they are able to arrive at the optimal policy for single server systems, however this is due to some convenient properties of the ERP cost function.

Our contributions offer a deeper understanding of the optimal policy for single server energy-aware systems, and are as follows.

- We perform our analysis under a large family of cost functions, based on the expected response time, expected number of jobs in the system, expected energy used, the expected turn-off rate of the server, and also the expectations of the product of these metrics. Furthermore, our analysis allows one to determine the optimal policy under any of these previously mentioned cost functions.
- We give an explicit solution to the underlying CTMC for our model. To the best of our knowledge, this CTMC has not been previously solved.
- We extend our analysis to considerable generality with respect to the underlying distributions. That is, we offer closed form solutions for all of our cost function metrics, under completely general server setup times, and job processing time distributions. We also offer several insightful observations pertaining to these metrics and how different system configurations relate to them.
- We offer several applications of our model. This includes applying our results to a multi-server system with random routing to show that in general when energy concerns are a factor, classical load balancing may be far from optimal.

The organization of this paper is as follows. In Section 2 a formal model of the system is presented. We continue by giving a detailed analysis of this model in Section 3. We firstly impose the assumption that all underlying distributions are exponential and therefore the model can be analysed as a CTMC. We progressively relax these assumptions and analyse the model under almost complete generality, offering a variety of insights and results. Section 4 gives several applications of our model while Section 5 shows how the model can be applied to a two server random routing setting.

2. Model

We wish to capture the behaviour of a single server system, where the server can be dynamically set to a low or high state. Furthermore, we wish to add the restriction that jobs may only be processed when the server is in its higher state. Such a system is modelled as being in one of four energy states: *LOW, SETUP, BUSY,* or *IDLE.* Each of these energy states has a corresponding rate of energy consumption E_{Low}, E_{Setup}, E_{Busy}, and E_{Idle}, respectively. For simplicity of analysis and understanding, if $E_{Low} = 0$, we rename *LOW* to *OFF.* We will see that optimal policies typically depend on the ratios of the energy costs rather than the values themselves. Therefore, we take these ratios with respect to E_{Busy}, and denote them as r_{Low}, r_{Setup}, and r_{Idle}, where $r_{Idle} < 1$. For the remainder of this paper we will often refer to moving to a higher or lower state as turning the server on or off, respectively. For further ease of reading, we often abuse our nomenclature for the energy states. For example we may refer to the server or system being *IDLE, OFF* etc., where we implicitly mean that the server or system is in the energy state *IDLE, OFF* etc.

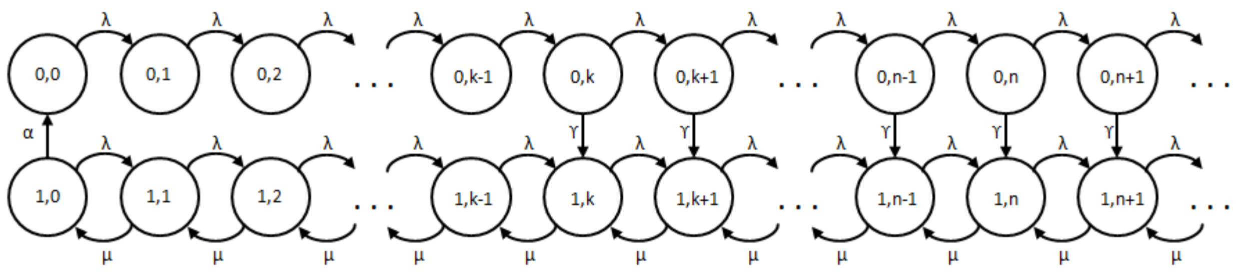


Figure 1: $M/M/1 \circ \{M, M, k\}$ queue Markov Chain

Jobs arrive to a FIFO queue according to a Poisson process of rate λ. If the system is *LOW/OFF* when a job arrives, it checks how many jobs are currently waiting in the queue. If the number in the queue plus the arriving job is equal to a given threshold k, the system moves into *SETUP*. This corresponds to the server turning on. The time it takes to make this transition is exponentially distributed with rate γ. Once the server has completed making its transition, it leaves *SETUP* and becomes *BUSY*. Once *BUSY*, the server begins to process the accumulated jobs. The job processing times are exponentially distributed with rate μ. When a job is completed and no jobs remain in the queue, the system becomes *IDLE*. Once *IDLE*, the system begins to accumulate idling time, since the last time it was turned on. If no job arrives to the system once the server has accumulated a given amount of idling time, the system becomes *LOW/OFF*. If a job arrives while the system is *IDLE*, it becomes *BUSY*. The amount of idling time which will be accumulated before the system becomes *LOW/OFF*, is exponentially distributed with rate α. It is important to note that the time spent in *IDLE* before transitioning to *OFF* is not the time which it takes for a server to turn off, but rather the amount of time it is willing to wait before deciding to turn off. For the purpose of our model, we assume that the time taken for the system to move from its high to low state is negligible, i.e. the transition occurs instantaneously. It is thought that the manager in charge of such a system has no control over λ, μ, and γ, but is free to choose α and k.

Due to the exponential assumptions, this system can be modelled as a CTMC and is depicted in Figure 1, where the system state (n_1, n_2) means that the server is off when $n_1 = 0$, on when $n_1 = 1$, and there are n_2 jobs in the system. One should note that the four previously defined energy states can be seen as a complete and disjoint set of the system states, (n_1, n_2). This interpretation is seen explicitly as:

$$OFF = \bigcup_{i=0}^{k-1} \{(0, i)\}, \quad SETUP = \bigcup_{i=k}^{\infty} \{(0, i)\}, \quad BUSY = \bigcup_{i=1}^{\infty} \{(1, i)\}, \quad IDLE = \{(1, 0)\}.$$

To denote these systems we use a composition of two sets of parameters i.e. $\{\} \circ \{\}$. The first set of parameters is given in classical Kendall notation to describe the non-energy-aware portions of the system. The set of parameters listed after the composition symbol are all parameters which are incorporated due to energy concerns. The first of these parameters is the setup time distribution of the server, the second is the idling time distribution, and the last is the number of jobs allowed to accumulate before the server begins to turn on. For example, the queue in Figure 1 is an $M/M/1 \circ \{M, M, k\}$ system while if the job processing times along with the server setup times follow general distributions, the system would be an $M/G/1 \circ \{G, M, k\}$. The reason for denoting the systems in this way, as we will show later, is that their metrics can often be written as a decomposition where one of the terms will be the corresponding metric of the non-energy-aware counterpart (the first set of parameters).

2.1. Justification of Assumptions and Parameter Summary

The model includes several assumptions in order to be tractable. Firstly, arrival times, setup times, processing times, and idling times are initially all assumed to be exponentially distributed. The assumptions on the arrival and processing times are quite standard for approximating systems of this kind. It will be shown later that the exponential assumption for the idling times is mitigated by properties of the optimal policies. However, in general, the assumption that the setup times of the servers as well as the job processing times are exponentially distributed can be problematic in practical settings. Later in our analysis we relax these assumptions on the distributions and analyse the system under general settings.

There are several constraints imposed on the model to ensure stability and that the model is non-trivial:

$$0 < \lambda < \mu, \quad 0 < \gamma, \quad 0 \leq \alpha, \quad 1 \leq k.$$

The parameters of the model are summarized in Table 1.

Table 1: Parameter Summary

Parameter(s)	**Explanation**
E_{Low}, E_{Setup}, E_{Busy}, E_{Idle}	The energy costs associated with the different energy states.
r_{Low}, r_{Setup}, r_{Idle}	The ratios between the energy costs and E_{Busy}.
λ	The arrival rate of jobs to the system.
μ	The server's processing rate.
γ	The rate at which the server turns on
α	The rate at which a server waits in energy state *IDLE* before moving to energy state *OFF*.
k	The number of jobs the system allows to accumulate in the queue while in energy state *OFF*, before moving to energy state *SETUP*.

3. Analysis

The goal of our analysis is to arrive at closed form expressions for a range of system metrics. Namely, we wish to solve for the expected number of jobs in the system, the expected response time of a job, the expected energy cost of the system, and the steady state rate of cycling between the low and high states (rate of turning off/on). We denote these quantities as $\mathbb{E}[N]$, $\mathbb{E}[R]$, $\mathbb{E}[E]$, and $\mathbb{E}[C]$, respectively. Once we derive these expressions, we can solve for optimal values of the parameters which the system manager has control over, α and k.

3.1. Set of Optimal Policies

We first define what we mean by an optimal policy. We define our cost to be a function of M weighted terms each containing $\mathbb{E}[R]$, $\mathbb{E}[E]$, $\mathbb{E}[C]$, each raised to given powers. We leave out the system metric $\mathbb{E}[N]$ since we can always obtain it by weighting $\mathbb{E}[R]$ by $1/\lambda$ via Little's Law. Formally, our cost function $f(\beta, w)$ is,

$$f(\beta, w) = \sum_{i=1}^{M} \beta_i \mathbb{E}[R]^{w_{R,i}} \mathbb{E}[E]^{w_{E,i}} \mathbb{E}[C]^{w_{C,i}}, \tag{1}$$

where $\forall i.\ 0 \leq \beta_i, w_{R,i}, w_{E,i}, w_{C,i}$ are of the appropriate units. Our model makes two assumptions about the optimal policies.

- The decision to start transitioning between *OFF* and *SETUP* is made at the moment a job arrives to the system.
- It is never optimal to transition to the energy state *OFF*, while the server is *BUSY*, i.e. it is never optimal to turn the server off if there are jobs in the system.

The first assumption is made without loss of generality due to the memoryless property of the arrival stream (the same decision would be made at any point in time between arrivals). The second assumption is a property of the optimal policy due to the nature of the cost function. If the system were to turn the server off while a job(s) remains in the system, $\mathbb{E}[R]$ will increase, since the job(s) that was in the system when it turned off must now wait until the system turns on before it can be completed. At the same time, the system does not gain any benefit with respect to the $\mathbb{E}[E]$ component since it will still have to expend energy to complete the job(s) in the system at some point in the future. So, as the weights in the cost function are positive we know that in the optimal policy the server will only be turned off while the server is idling. Similar assumptions are made in the model used in [7]. Knowing that these two assumptions are valid, we know that any optimal policy can be instantiated using the model we have described, under the model's assumptions.

Similar to the argument made to justify the servers beginning to turn on only when an arrival occurs to the system, the decision to turn a server off or keep it on is made when a job departs the system and leaves it idle. This would imply that in our model, in any policy which minimizes the cost, $\alpha = 0$ or $\alpha \to \infty$. We leave α as part of our model for several reasons. Firstly, it gives us insight on how scaling between these two extremes affects the system. Secondly, it allows us to easily determine where in the parameter space the optimal policy switches between $\alpha = 0$ and $\alpha \to \infty$. Thirdly, it allows for easier extensions of the model where this property may not necessarily hold. For example, this property does not hold when the arrivals do not follow a Poisson process, or in a multi-server setting. Lastly when optimizing under different conditions, i.e. minimizing a linear function of $\mathbb{E}[E]$ with a constraint on $\mathbb{E}[R]$, the optimal α could lie anywhere on the positive real line.

3.2. Steady State

Theorem 1. *The steady state distribution for an $M/M/1 \circ \{M, M, k\}$ queue, depicted by the Markov chain in Figure 1 is given by the set of equations* (2)-(6)*.*

$$\pi_{0,n} = \pi_{0,0} \qquad (0 \le n < k) \quad (2)$$

$$\pi_{0,n} = \pi_{0,0}\left(\frac{\lambda}{\lambda+\gamma}\right)^{n-(k-1)} \qquad (k \le n) \quad (3)$$

$$\pi_{1,n} = \pi_{0,0}\left(\frac{\lambda}{\alpha}\rho^n + \frac{\lambda}{\mu-\lambda}(1-\rho^n)\right) \qquad (0 \le n < k) \quad (4)$$

$$\pi_{1,n} = \pi_{0,0}\left[\left(\frac{\lambda}{\alpha} - \frac{\lambda}{\mu-\lambda}\right)\rho^n + \frac{1}{\mu-\lambda-\gamma}\left((\lambda+\gamma)\left(\frac{\lambda}{\lambda+\gamma}\right)^{n-(k-1)} - \frac{\gamma}{1-\rho}\rho^{n-(k-1)}\right)\right] \qquad (k \le n) \quad (5)$$

$$\pi_{0,0} = (1-\rho)\frac{\alpha\gamma}{k\alpha\gamma + \alpha\lambda + \lambda\gamma} \quad (6)$$

Proof. Each row of the Markov chain depicted in Figure 1 is partitioned into two sections according to $n < k$, or $n \ge k$. The balance equations used to solve for the four different sections of the Markov chain are:

$$\begin{aligned}
\pi_{0,n} &= \pi_{0,0} && (n < k)\\
(\lambda+\gamma)\pi_{0,n} &= \lambda\pi_{0,n-1} && (n \ge k)\\
\mu\pi_{1,n} &= \lambda\pi_{1,n-1} + \lambda\pi_{0,n-1} && (0 < n < k)\\
(\mu+\lambda)\pi_{1,n} &= \lambda\pi_{0,n-1} + \gamma\pi_{0,n} + \mu\pi_{1,n+1} && (n \ge k)
\end{aligned}$$

where π_{n_1,n_2} denotes the steady state probability of being in state (n_1, n_2). We also have the boundary and normalization conditions:

$$\pi_{1,0} = \frac{\lambda}{\alpha}\pi_{0,0} \quad \text{and} \quad \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\pi_{n_1,n_2} = 1.$$

While the first three balance equations can be solved with respect to $\pi_{0,0}$ via simple recursions, the fourth equation requires more effort. However, after solving the non-repeating portion of the Markov chain, we note that we can apply similar methods to those used in [2]. We are able to arrive at a closed form solution as follows. For $n > k$, we fit the steady state distribution to be of the form,

$$\pi_{1,n} = A\rho^{n-(k-1)} + B\left(\frac{\lambda}{\lambda+\gamma}\right)^{n-(k-1)}$$

where with the use of the boundary equations we find that,

$$B = \pi_{0,0}\frac{\lambda+\gamma}{\mu-\lambda-\gamma}$$

and

$$A = \pi_{0,0}\left[\left(\frac{\lambda}{\alpha} - \frac{\lambda}{\mu-\lambda}\right)\rho^{k-1} - \frac{\mu\gamma}{(\mu-\lambda)(\mu-\lambda-\gamma)}\right].$$

With the balance equations solved we use some algebra to yield the steady state distribution for our system model. A full detailed derivation of the steady state distribution can be found in [12]. □

3.3. System Metrics

With the steady state distribution of our model in hand, we wish to arrive at closed form expressions for the system metrics, namely $\mathbb{E}[N]$, $\mathbb{E}[R]$, $\mathbb{E}[E]$, and $\mathbb{E}[C]$. Determining these expectations will allow us to build expressions for our cost function and in turn allow us to arrive at optimal values for α and k.

The simplest expression to solve for is $\mathbb{E}[C]$, the steady state rate at which the server turns off. The only energy state from which the server turns off is *IDLE*, which has steady state probability $\pi_{1,0}$. Therefore the expected cycle rate is just the rate out of *IDLE* going to *OFF*.

$$\mathbb{E}[C] = \alpha\pi_{1,0} = (1-\rho)\frac{\alpha\lambda\gamma}{k\alpha\gamma + \alpha\lambda + \lambda\gamma} \tag{7}$$

The general form of this expression is not unexpected. Firstly, the direct relationship to $(1-\rho)$ is quite intuitive as a heavily loaded system would rarely turn off. Secondly, k only appears in the denominator, giving $\mathbb{E}[C]$ an inverse relationship to k. This is also expected as allowing k jobs to build up slows down the turn on rate of the server as k increases, and the expected turn on rate is equal to the expected turn off rate.

We solve $\mathbb{E}[E]$ by viewing it as a sum of being in energy states *OFF*, *IDLE*, *SETUP*, and *BUSY* weighted by the corresponding energy costs. We sum the states using equations (2)-(6), and exploit our assumption that $E_{Low} = 0$ while the server is *OFF*.

$$\begin{aligned}\mathbb{E}[E] &= E_{Busy}\sum_{n=1}^{\infty}\pi_{1,n} + E_{Setup}\sum_{n=k}^{\infty}\pi_{0,n} + E_{Idle}\pi_{1,0} \\ &= E_{Busy}\left[\rho + \frac{(1-\rho)\lambda}{k\alpha\gamma + \alpha\lambda + \lambda\gamma}(r_{Idle}\gamma + r_{Setup}\alpha)\right]\end{aligned}$$

Letting $\mathbb{E}[E_{M/M/1}]$ denote the expected energy cost in an $M/M/1$ queue, and observing that $\mathbb{E}[E_{M/M/1}] = E_{Busy}\rho + E_{Idle}(1-\rho)$ leads to:

$$\mathbb{E}[E] = \mathbb{E}[E_{M/M/1}] + E_{Busy}\frac{(1-\rho)\alpha}{k\alpha\gamma + \alpha\lambda + \lambda\gamma}(\lambda r_{setup} - (\lambda + k\gamma)r_{idle}).$$

This gives us the true expected energy cost of the system, however since the $\mathbb{E}[E]$ term in in our cost function is weighted by a constant β, we can absorb the constant E_{Busy}, and derive a new metric normalized by this weight:

$$\mathbb{E}[E^N] = \frac{\mathbb{E}[E]}{E_{Busy}} = \mathbb{E}[E^N_{M/M/1}] + \frac{(1-\rho)\alpha}{k\alpha\gamma + \alpha\lambda + \lambda\gamma}(\lambda r_{setup} - (\lambda + k\gamma)r_{idle}). \tag{8}$$

We arrive again at a decomposition, the terms here are scaled by ρ or $(1-\rho)$. As we would expect there is an $r_{busy}\rho$ term present (contained within the term $\mathbb{E}[E^N_{M/M/1}]$), since based on our model assumptions $r_{busy}\rho$ is a lower bound for the expected normalized energy cost. We also note that letting $\alpha = 0$ simply leaves us with $\mathbb{E}[E^N_{M/M/1}]$, the expected normalized energy consumption in an M/M/1 queue, as anticipated. How the rest of the terms arise is at this point not intuitively clear, but in the next section we give a different point of view on $\mathbb{E}[E]$ which allows us to gain much more insight.

To solve for $\mathbb{E}[R]$, we use the traditional method of solving first for $\mathbb{E}[N]$ by weighting the steady state distribution and then applying Little's Law. After quite a bit of algebra we are able to write:

$$\mathbb{E}[N] = \mathbb{E}[N_{M/M/1}] + \frac{\alpha\lambda(\lambda + k\gamma)}{\gamma(k\alpha\gamma + \alpha\lambda + \lambda\gamma)} + \frac{k\alpha\gamma(k-1)}{2(k\alpha\gamma + \alpha\lambda + \lambda\gamma)}.$$

Applying Little's Law gives us:

$$\mathbb{E}[R] = \mathbb{E}[R_{M/M/1}] + \frac{1}{\gamma}\frac{\alpha(\lambda + k\gamma)}{k\alpha\gamma + \alpha\lambda + \lambda\gamma} + \frac{k-1}{2\lambda}\frac{k\alpha\gamma}{k\alpha\gamma + \alpha\lambda + \lambda\gamma}. \tag{9}$$

Both terms yield convenient decompositions. We would expect to find some form of the $M/M/1$ queue embedded within the $M/M/1 \circ \{M, M, k\}$ queue since many of its metrics are optimized when their behaviours are equivalent, that is when $\alpha = 0$. As a sanity check, letting $\alpha = 0$ in the expressions for $\mathbb{E}[N]$ and $\mathbb{E}[R]$ reduces to exact $M/M/1$ expressions, as we would expect. Furthermore, $\mathbb{E}[N_{M/M/1}]$ and $\mathbb{E}[R_{M/M/1}]$ are lower bounds for $\mathbb{E}[N]$ and $\mathbb{E}[R]$, respectively.

To analyse the second term of (9), it is easier to first allow $k = 1$, which eliminates the third term. With $k = 1$ and letting α approach ∞, our system reduces to that of the system described in [7], where $\mathbb{E}[R] = \mathbb{E}[R_{M/M/1}] + 1/\gamma$. We can see from (9) that our result agrees. So the expected response time of a job is bounded below by $\mathbb{E}[R_{M/M/1}]$ and bounded above by $\mathbb{E}[R_{M/M/1}] + 1/\gamma$, when $k = 1$. Moving α along the positive real line scales $\mathbb{E}[R]$ between these two bounds. When $k > 1$ the behaviour is similar, as k appears in both the numerator and denominator, scaled by the same coefficient. Therefore, the second term still adds a value between 0 and $1/\gamma$. However, when $k > 1$ the third term no longer equals 0, so $\mathbb{E}[R_{M/M/1}]$ is no longer bounded above by $\mathbb{E}[R_{M/M/1}] + 1/\gamma$.

The third and last term of (9) quantifies the effect imposed on the response time when k jobs are allowed to accumulate. As k increases, we see a linear increase in the third term. While no clear intuition is available, we can see that the third term is weighted by $\frac{k-1}{2\lambda}$, which is the expected amount of time spent *OFF* after a single job has arrived.

Viewing equations (7), (8), and (9) together, one begins to understand the trade-offs when optimizing a given cost function by choosing α and k. Each individual metric prefers α and k to be either set to their respective upper or lower bounds, but unfortunately they pull in different directions, as seen in Table 2.

Note that to minimize $\mathbb{E}[E]$, $\alpha = 0$ when

$$r_{idle} < \frac{\lambda}{k\gamma + \lambda}r_{setup}$$

Table 2: Optimal Parameters of Metrics

	Optimal Values of	
Metric	α	k
$\mathbb{E}[R]$	0	1
$\mathbb{E}[E]$	0 or $\to \infty$	$\to \infty$
$\mathbb{E}[C]$	0	$\to \infty$

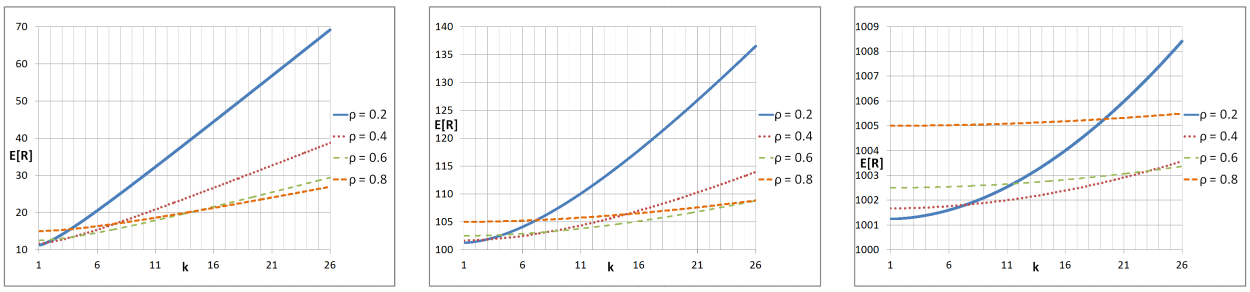
and $\alpha \to \infty$ otherwise.

The impact that the decision variables, k and α, have on a given metric can further be examined with some interesting results. Primarily, we wish to inspect how the choice of k affects the expected response time. While the other metrics are also interesting, as will be seen later these can be looked at in a much more general setting. Furthermore, while the choice of α highly impacts the metric, we know that the arrivals follow a Poisson process, any given cost function is minimized when $\alpha = 0$ or when $\alpha \to \infty$. We also know that if $\alpha = 0$, then the choice of k is irrelevant, since in steady state the server will never turn off. Due to these observations, we can look at (9) as $\alpha \to \infty$,

$$\mathbb{E}[R] = \mathbb{E}[R_{M/M/1}] + \frac{1}{\gamma} + \frac{k-1}{2\lambda}\left(\frac{k\gamma}{k\gamma + \lambda}\right). \tag{10}$$

Figure 2 shows the relationship that $\mathbb{E}[R]$ has to k under several configurations. The reader is reminded that while k is shown on a continuous range (to better understand the mathematical relationship it has to these systems), in practice it must take on discrete values. Here one can see a somewhat counter intuitive result, where as k increases, the expected response time for heavily loaded systems becomes less than that of a lightly loaded system. This leads to our first observation.

Observation 1. *Given any two $M/M/1 \circ \{M, M, k\}$ queues where one has a higher load than the other, there exists a value $k*$ such that for all values of $k > k*$, the expected response time of the lighter loaded queue, is greater than that of the heavier loaded queue.*



(a) $\mu = 1, \gamma = 0.1$ (b) $\mu = 1, \gamma = 0.01$ (c) $\mu = 1, \gamma = 0.001$

Figure 2: $M/M/1 \circ \{M, M, k\}$ expected response time vs k for varying γ values

At first this may seem surprising, as a lighter load is often associated with a lower expected response time, but it is in fact quite intuitive. Consider the system when the server is turned off and there are no jobs in the system, i.e. in the system state $(0, 0)$. The next $k - 1$ jobs which arrive will have to wait for the server to turn on, as well as the remaining time for the server to decide to begin turning on. That is, the jobs will have to wait for the server to move through the energy states *OFF* and *SETUP*. While the expected time the system spends in *SETUP* is completely independent of k and λ, this is not the case for the time spent *OFF*. In fact, the expected amount of time the system spends *OFF* is the expected amount of time it takes for k jobs to arrive, which equals k/λ. For a lightly loaded system this takes a much longer time compared to a heavily loaded one, and part of this extra time is added to the response time for all $k-1$ jobs which arrived during that period. This can further be seen in the third term of (10) which captures this behaviour. This result illustrates the value of carefully choosing k, since it could have surprisingly negative effects. Especially since at first glance it may be appealing to turn the server off and keep it off for a longer period, if the load is light.

3.4. Regeneration Period Analysis

Here we approach the analysis of our system from a different angle. This method allows us to relax several of our assumptions while still arriving at closed form expressions, as well as allowing us to gain deeper insight and intuition into the system behaviour.

Theorem 2. *The proportion of time spent in the energy states of an $M/G/1 \circ \{G, G, k\}$ queue is insensitive to the distributions themselves (beyond the means). As a result,*

$$\mathbb{E}[E^N] = \rho + (1-\rho)\frac{\lambda}{k\alpha\gamma + \alpha\lambda + \lambda\gamma}(r_{Idle}\gamma + r_{Setup}\alpha),$$
$$\mathbb{E}[C] = (1-\rho)\frac{\alpha\lambda\gamma}{k\alpha\gamma + \alpha\lambda + \lambda\gamma},$$

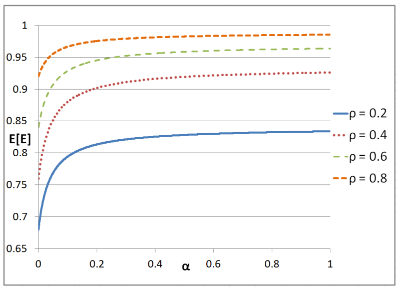
and for any single server system if $r_{idle} < \frac{\lambda}{k\gamma+\lambda} r_{setup}$, then it is always optimal to leave the server on.

Proof. We view the system using the rate at which "regeneration periods" complete. Let $S_{0,0}$ denote the state of the system where the server is off and there are 0 jobs in the system, and let $P_{0,0}$ denote the proportion of time the system spends in $S_{0,0}$ in steady state. We define our regeneration period to start at system state $S_{0,0}$, moving through energy state *OFF* into energy state *SETUP*. Once the server has turned on, it continues to move between energy states *BUSY* and *IDLE* a number of times before it lastly moves from *IDLE* back to $S_{0,0}$. The reader is reminded that in our model, during a regeneration period the total time spent idling is accumulated when moving between the energy states *BUSY* and *IDLE*. Once the system moves back to state $S_{0,0}$ the idling time is reset. Since for every regeneration period the system visits state $S_{0,0}$ exactly once, the rate at which regeneration periods occur in the system is the rate out of state $S_{0,0}$. When in state $S_{0,0}$, the rate out is simply the arrival rate to the system, λ. Therefore in steady state the regeneration period rate is $\lambda P_{0,0}$.

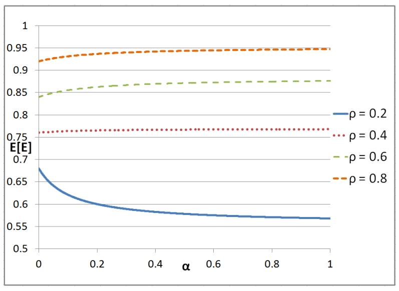
We also make the observation that the expected proportion of time which the system spends in energy states *OFF*, *SETUP*, *BUSY* and *IDLE* (denoted P_{Off}, P_{Setup}, P_{Busy}, and P_{Idle} respectively) over just one of its regeneration periods, is equal to the proportion of time the system spends in those energy states, in steady state. This is a consequence of the renewal reward theorem and the observation that the system regenerates each time it enters state $S_{0,0}$ (since in $S_{0,0}$ all active events are exponentially distributed, i.e. the interarrival times). For each regeneration period the server turns on a single time, therefore P_{Setup} equals the product of the regeneration process rate and the expected setup time of the server, i.e. $(\lambda/\gamma)P_{0,0}$. This same argument can be used for P_{Off}, which is the time it takes for k jobs to arrive to the system multiplied with the regeneration period rate, $\lambda(k/\lambda)P_{0,0} = kP_{0,0}$. We know that the rate into state $S_{0,0}$ must equal the rate out which implies $P_{1,0} = (\lambda/\alpha)P_{0,0}$. However, once again this is also just the product of the expected idling time and the regeneration period rate of the system. Finally, we get the proportion of time the system spends *BUSY* for free since we know it must be ρ. Putting it all together we have:

$$1 = \rho + \frac{\lambda}{\alpha}P_{0,0} + \frac{\lambda}{\gamma}P_{0,0} + kP_{0,0}.$$

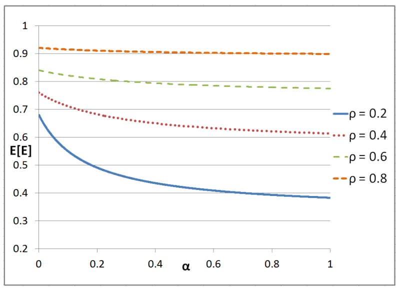
This analysis has been done without imposing assumptions on any of the distributions, except for the arrival stream. This assumption was used when we calculated the rate out of state $S_{0,0}$ to be $\lambda P_{0,0}$, as well as when invoking the renewal reward theorem. Isolating and solving for $P_{0,0}$ we find it is equal to $\pi_{0,0}$ from our previous analysis, and the same can be said for the expected energy used by the system, that is the expected energy used in an $M/M/1 \circ \{M, M, k\}$ system equals that of an $M/G/1 \circ \{G, G, k\}$ system. $\square$



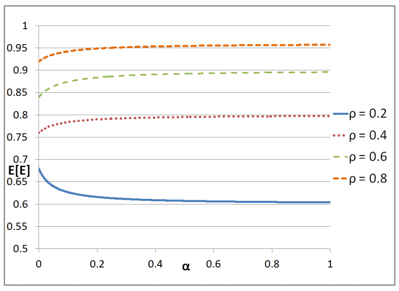
(a) $\mu = 1, \gamma = 0.01, k = 1$



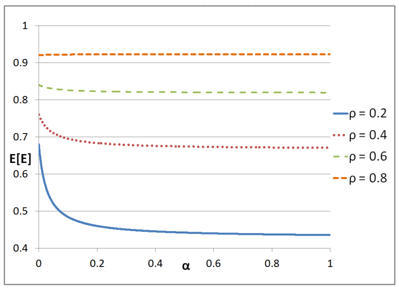
(b) $\mu = 1, \gamma = 0.25, k = 1$



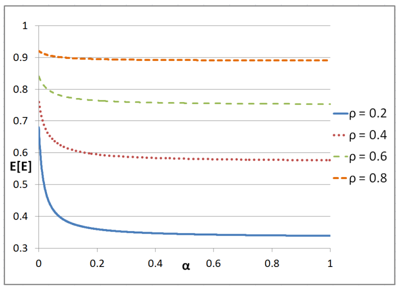
(c) $\mu = 1, \gamma = 1, k = 1$



(d) $\mu = 1, \gamma = 0.1, k = 2$



(e) $\mu = 1, \gamma = 0.1, k = 5$



(f) $\mu = 1, \gamma = 0.1, k = 10$

Figure 3: Expected energy consumption vs α under varying γ and k

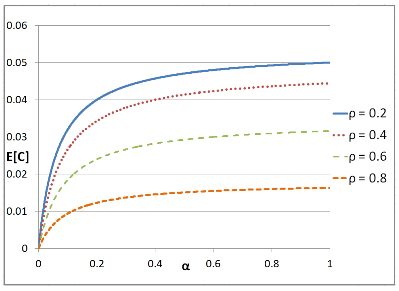
It may be counter intuitive for the proportion of time spent in different energy states to be independent of the underlying distributions of the processing and setup times. However, this is similar to the fact that an $M/G/1$ queue is $BUSY$ and $IDLE$ with probabilities ρ and $(1 - \rho)$, respectively. Here one can see why the exact analysis on the effects which the system parameters have on $\mathbb{E}[E]$ and $\mathbb{E}[C]$ was previously deferred, since here we can perform the analysis in greater generality.

Figure 3 shows the relationship of the expected energy versus α for several configurations with varying values of ρ, γ, and k. Here $r_{idle} = 0.6$ and $r_{setup} = 1$. As was seen in Table 2, $\mathbb{E}[E]$ is the only metric which can be minimized when $\alpha = 0$ or when $\alpha \to \infty$. The transition between optimal values can be seen graphically in Figures 3-(a)-(c). When the setup times are high, it is optimal to always leave the server on, even for lightly loaded systems. As the setup time for the server decreases, the optimal configuration becomes immediately turning the server off when it becomes idle. This transition occurs firstly in lighter loaded systems, as seen in Figure 3-(b), but eventually also occurs in the heavily loaded systems. This relation is also seen in Figures 3-(d)-(f) as k increases, while other parameters are held constant. This interchangeable effect of changes in γ and k can be seen mathematically as well, and leads to an interesting observation.

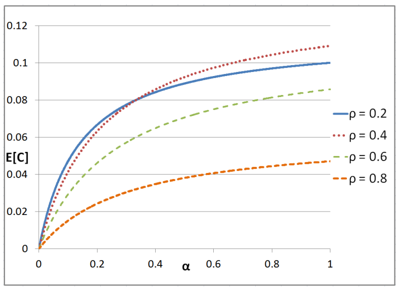
Observation 2. *The expression for the expected energy consumed by a single server system where the server instantly turns off when it idles, i.e.* (8) *as* $\alpha \to \infty$*, is symmetric in* k *and* γ*.*

This result is quite surprising, as the parameters k and γ have no true equivalence in any intuitive sense. The higher the value of k the longer the system will wait before beginning to turn on, while the higher the value of γ the quicker the server turns on. Furthermore, the feasible domain of both variables is not even equal, as $k \in \mathbb{N}$, while $\gamma \in \mathbb{R}^+$. While this relationship of course does not hold with respect to other system metrics such as $\mathbb{E}[E]$, the impact which k and γ have on $\mathbb{E}[C]$ remains interesting.

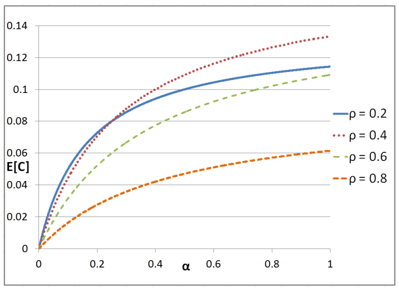
Similar to the preceding discussion of energy costs, Figure 4 shows the relationship which the expected cycle rate, $\mathbb{E}[C]$, has for various configurations. As expected, $\mathbb{E}[C]$ is minimized when $\alpha = 0$, since the server will never turn off, and therefore never turn on in steady state. But as $\alpha \to \infty$, $\mathbb{E}[C]$ begins to exhibit



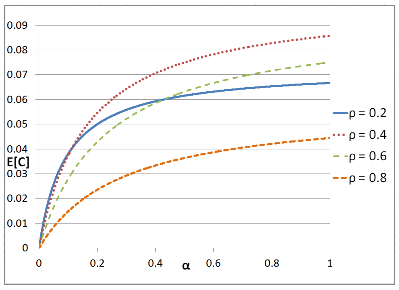
(a) $\mu = 1, \gamma = 0.01, k = 1$



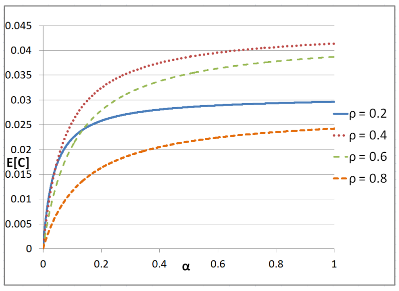
(b) $\mu = 1, \gamma = 0.5, k = 1$



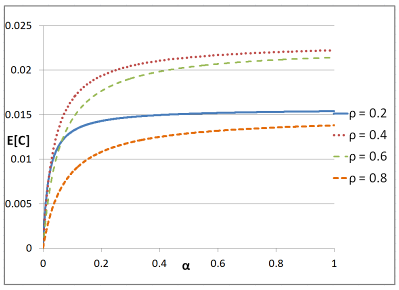
(c) $\mu = 1, \gamma = 1, k = 1$



(d) $\mu = 1, \gamma = 1, k = 2$



(e) $\mu = 1, \gamma = 1, k = 5$



(f) $\mu = 1, \gamma = 1, k = 10$

Figure 4: Expected cycle rate vs α under varying γ and k

some unexpected behaviour. We note that in Figures 4-(a)-(c), as γ increases, the cycle rate for a lightly loaded system ($\rho = 0.2$) begins to fall below that of a more heavily loaded system ($\rho = 0.4$). Since in these figures $k = 1$, one may expect that the lightly loaded system will always have a higher cycle rate since the busy period is much shorter. However, this is not what we observe. This effect is further seen as k is also increased in Figures 4-(d)-(f). Not only do more heavily loaded systems begin to overtake more lightly loaded systems, but certain curves seem to be converging to the same value. In particular the curves where $\rho = 0.2, 0.8$ and $\rho = 0.4, 0.6$ approach the same values as α, k, and γ increase, giving us another unexpected observation.

Observation 3. *As α, γ, and k increase, the expected cycle rate in an $M/G/1 \circ \{G, G, k\}$ for systems with loads equidistant from* 0.5 *approach equal values, and furthermore the greater the value of $|\rho - 0.5|$ the lower the cycle rate will be.*

This observation has quite a bit of intuition behind it. Firstly, it is clear that as $\rho \to 0$ or as $\rho \to 1$, $\mathbb{E}[C] \to 0$ since the server will either never be busy, or never be idle. Furthermore, as we slightly move away from these two extremes, $\mathbb{E}[C]$ must increase as opportunities for the server to turn on and off begin to arise. Secondly, in a lightly loaded system with a large enough k, the time the server waits before turning on is significantly larger than for a heavily loaded system, as was discussed earlier. On the other hand, the time spent clearing an arbitrary number of jobs out of the system once the server turns on can take significantly longer in a heavily loaded system compared to a lightly loaded one. By extending the time of a regeneration period, both of these effects lower the expected cycle rate. Thinking about the system in this way allows us to see why heavily and lightly loaded systems have a lower expected cycle rate. However, it does not offer any direct intuition why a symmetry arises around $\rho = 0.5$, nor why $\rho = 0.5$ gives the highest expected cycle rate as α, γ and k grow.

With energy and cycling analysed, to move forward in our analysis in an attempt to solve for the expected response time under less restrictive assumptions than those for the previously solved $M/M/1 \circ \{M, M, k\}$ system. While the energy and cycle metrics can be solved in almost complete generality ($M/G/1 \circ \{G, G, k\}$), the response time is harder to arrive at. We therefore again impose exponential assumptions upon the idling times of the system, but still allow for general distributions for the processing and setup times. Some generality is lost, but we argue that the exponential assumptions on the idling and interarrival times are not nearly as limiting. For many applications, modelling the arrivals as a Poisson process is a reasonable assumption, while as we have stated before, having the server setup times and job processing times being exponentially distributed can be problematic. We also know that if the arrivals do follow a Poisson process then α is either 0 or approaches ∞, meaning the actual distribution has little impact. With this in mind, we analyse the $M/G/1 \circ \{G, M, k\}$ queue with the goal of determining $\mathbb{E}[R]$.

Theorem 3. *For an $M/G/1 \circ \{G, M, k\}$ queue, the expected number of jobs in the system and the expected response time for a job are given by:*

$$\begin{aligned}\mathbb{E}[N] = \mathbb{E}[N_{M/G/1}] + \alpha\frac{\gamma(k-1)+\lambda}{k\alpha\gamma+\alpha\lambda+\lambda\gamma}\left[\frac{1}{2} - \rho - \rho\frac{\alpha}{\alpha+\lambda}\left(\frac{\gamma(k-1)+\lambda}{\gamma}\right) - \frac{1}{2}\frac{\alpha}{\alpha+\lambda}\Gamma\right] \\ + \rho\frac{\alpha}{\alpha+\lambda}\left(\frac{\gamma(k-1)+\lambda}{\gamma}\right) + \frac{1}{2}\frac{\alpha}{\alpha+\lambda}\Gamma,\end{aligned}$$

where letting $Var(G)$ denote the variance of the setup time distribution,

$$\Gamma = (k-1)^2 + (2k-1)\frac{\lambda}{\gamma} + \lambda^2 Var(G),$$

and

$$\mathbb{E}[R] = \frac{\mathbb{E}[N]}{\lambda}.$$

Proof. We tackle the problem by examining an embedded Markov chain, as is traditionally done for the $M/G/1$ queue. We define N_n to be a random variable denoting the number of jobs left in the system as the n^{th} job departs. As in the $M/G/1$ analysis,

$$N_{n+1} = \begin{cases} N_n + A_{n+1} - 1 & N_n \geq 1; \\ A_{n+1} & N_n = 0, \end{cases}$$

where A_{n+1} denotes the number of arrivals which occurred between the departure of the n^{th} and $(n+1)^{th}$ jobs, not counting the $(n+1)^{th}$ if it arrived during that period. For our model, we have to condition A_{n+1} on N_n,

$$A_{n+1} = \begin{cases} A_{S,n} & N_n \geq 1; \\ A_{S,n} + X_{Off,n}(k-1+A_{\Gamma,n}) & N_n = 0, \end{cases}$$

where $A_{S,n}$ is a random variable denoting the number of jobs which arrive while the n^{th} job is being processed. $A_{\Gamma,n}$ is a random variable denoting the number of jobs which arrive to the system during the server's setup time, given the $(n+1)^{th}$ job is the first to arrive once the server has turned off. $X_{Off,n}$ is an indicator variable that is 1 when the system is *IDLE* and the next energy state it moves to is *OFF* and 0 if the next energy state it moves to is *BUSY*, given that the n^{th} job to depart leaves behind an empty system. We note that the distributions for all three of these random variables are independent of n, and from here on refer to them simply as A_S, A_Γ, and X_{Off}. We can now rewrite the expressions for N_{n+1} and A_{n+1} with the use of the Heaviside step function.

$$\begin{aligned} N_{n+1} &= N_n - \mathcal{U}(N_n) + A_{n+1} \\ A_{n+1} &= A_S + (1-\mathcal{U}(N_n))X_{Off}(k-1+A_\Gamma) \\ \Rightarrow N_{n+1} &= N_n - \mathcal{U}(N_n) + A_S \\ &\quad + (1-\mathcal{U}(N_n))X_{Off}(k-1+A_\Gamma) \qquad (11) \end{aligned}$$

If we let $n \to \infty$ and then take the expectation of both sides, the N_n and N_{n+1} terms cancel out. We also exploit the fact that X_{Off} is independent from A_Γ, since A_Γ is dependent only on the interarrival and setup times. After some algebra we are left with an expression for $\mathbb{E}[\mathcal{U}(N)]$.

$$\mathbb{E}[\mathcal{U}(N)] = \frac{\mathbb{E}[A_S] + \mathbb{E}[X_{Off}](k-1+\mathbb{E}[A_\Gamma])}{1+\mathbb{E}[X_{Off}](k-1+\mathbb{E}[A_\Gamma])}$$

This should not give us any new information about the system, as for an $M/G/1$ queue this would yield $\mathbb{E}[\mathcal{U}(N)] = \rho$. Of course the interpretation of $\mathbb{E}[\mathcal{U}(N)]$ is the steady state probability there is at least one job in the system. From our previous analysis on the proportion of time in the energy states, we know this to be:

$$1 - P_{0,0} - P_{1,0} = \rho + (1-\rho)\alpha\frac{\gamma(k-1)+\lambda}{k\alpha\gamma+\alpha\lambda+\lambda\gamma}.$$

As a sanity check this is what $\mathbb{E}[\mathcal{U}(N)]$ evaluates to when,

$$\mathbb{E}[A_S] = \rho, \quad \mathbb{E}[X_{Off}] = \frac{\alpha}{\lambda+\alpha}, \quad \text{and} \quad \mathbb{E}[A_\Gamma] = \frac{\lambda}{\gamma}.$$

To arrive at $\mathbb{E}[N]$, we use the usual approach: square both sides of (11), let $n \to \infty$, take expectations and exploit the following equalities.

$$\begin{aligned}
\mathcal{U}^2(N) &= \mathcal{U}(N) \\
N\mathcal{U}(N) &= N \\
N(1-\mathcal{U}(N)) &= 0 \\
\mathcal{U}(N)(1-\mathcal{U}(N)) &= 0 \\
\mathbb{E}[X_{Off}A_S] &= \mathbb{E}[X_{Off}]\mathbb{E}[A_S] \\
\mathbb{E}[X_{Off}A_\Gamma] &= \mathbb{E}[X_{Off}]\mathbb{E}[A_\Gamma]
\end{aligned}$$

Substituting those equations into (10) after squaring both sides yields,

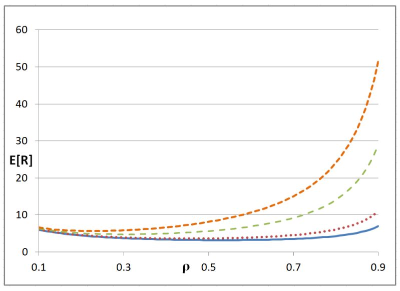
$$\begin{aligned}
2(1-\mathbb{E}[A_S])\mathbb{E}[N] = \mathbb{E}[\mathcal{U}(N)][1+2\mathbb{E}[A_S](1-\mathbb{E}[X_{Off}](k-1+\mathbb{E}[A_\Gamma]))] + \mathbb{E}[A_S^2]\mathbb{E}[X_{Off}](k-1+\mathbb{E}[A_S]) \\
+ (1-\mathbb{E}[\mathcal{U}(N)])\mathbb{E}[X_{Off}]((k-1)^2 + 2(k-1)\mathbb{E}[A_\Gamma] + \mathbb{E}[A_\Gamma^2]).
\end{aligned}$$

After some algebra, we are able to arrive at a relatively clean expression for the expected number of jobs in the system. □

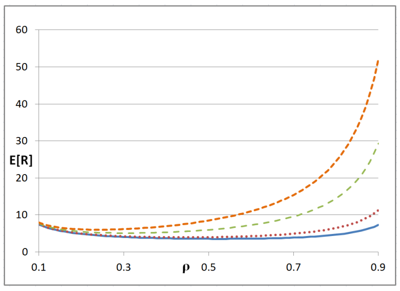
Again we see this recurring decomposition of the energy-aware system into its classical queue counterpart plus additional terms. We would expect to see this result for the same reasons discussed when we solved the $M/M/1 \circ \{M, M, k\}$ queue. Combining Theorem 2 and Theorem 3 now gives us the tools to optimize $M/G/1 \circ \{G, M, k\}$ systems under any metric of the form given in (1).

With the expected response time solved under general distributional assumptions for the processing and setup times, we take this opportunity to gain a deeper insight into this metric than provided in Section 3.3. Figure 5-(a)-(c) shows the relationship which $\mathbb{E}[R]$ has under different system configurations, as well as different processing time variance values. The results here are not surprising as $\mathbb{E}[R]$ can be seen as a decomposition, and the only place the variance of the processing times is present is in the $\mathbb{E}[R_{M/G/1}]$ term, which is well understood. Looking at the impact which the setup time variance has on $\mathbb{E}[R]$ is a different story.

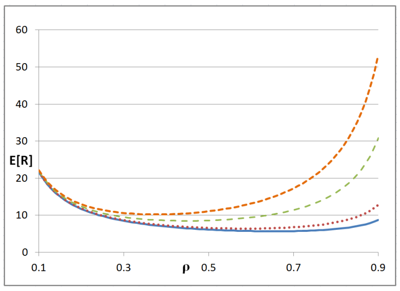
Observation 4. *Although $\mathbb{E}[R]$ is dependent on the setup time variance in an $M/G/1 \circ \{G, M, k\}$ system, it is relatively insensitive to it.*



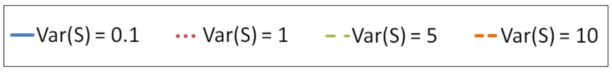
(a) $\mu = 1, \gamma = 0.1, k = 1, Var(G) = 0.01$

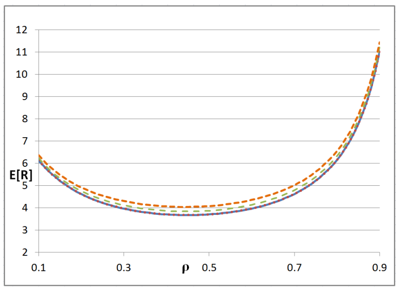


(b) $\mu = 1, \gamma = 0.5, k = 2, Var(G) = 0.05$

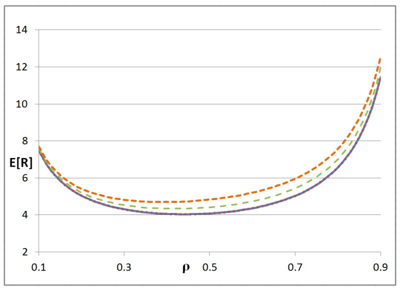


(c) $\mu = 1, \gamma = 1, k = 5, Var(G) = 1$

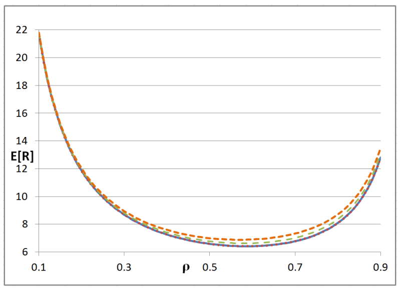




(e) $\mu = 1, \gamma = 0.1, k = 1, Var(S) = 1$



(f) $\mu = 1, \gamma = 0.1, k = 1, Var(S) = 5$



(g) $\mu = 1, \gamma = 0.1, k = 1, Var(S) = 10$

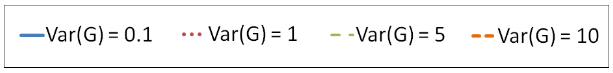


Figure 5: Variance effects: expected response time vs p

This can be seen in Figures 5-(e)-(g) and for the most part is good news for a system manager. This parameter is likely not to be initially known, and due to this low sensitivity, making an assumption or approximation may be a reasonable choice. The reason for this low sensitivity is not exactly clear, but we do note that during a single regeneration period, there is only one corresponding random variable for the setup times. While in contrast, there could be many invocations of of the processing time distribution (at least k).

3.5. Expectation of Products

While (1) encapsulates a large family of cost functions, there is another class of relevant cost functions which previous analysis fails to cover. Cost functions of the form (1) can be instantiated as the product of expectations, for example $\mathbb{E}[R]\mathbb{E}[E]\mathbb{E}[C]$, but cannot be instantiated as the expectation of products, for example $\mathbb{E}[R \cdot E \cdot C]$. This section extends our set of valid cost functions, and shows that, perhaps surprisingly, many of these cost functions are minimized under trivial configurations.

The new set of cost functions is defined as follows,

$$f(w) = \mathbb{E}[R^{w_1} \cdot E^{w_2} \cdot N^{w_3} \cdot C^{w_4}], \tag{12}$$

where, $\forall i. w_i \geq 0$. Note that unlike in (1), N is included, since in this form Little's law can no longer be applied to make the term redundant.

Theorem 4. *Cost functions of the form $\mathbb{E}[R^{w_1} \cdot E^{w_2} \cdot N^{w_3} \cdot C^{w_4}]$, where w_1, w_2, w_3, and w_4 are positive integers, are all trivially minimized when the server always remains on, expect for cost functions of the form $\mathbb{E}[R^{w_1} \cdot E^{w_2}]$ which are minimized when the server always remains on, if $r_{idle} < \frac{\lambda}{k\gamma+\lambda} r_{setup}$ for an optimal value of k.*

Proof. Firstly, we observe that if $w_4 > 0$ then the cost function is trivially minimized, as keeping the server always on guarantees that $C = 0$.

We secondly inspect the cost function $\mathbb{E}[E \cdot N]$, and show this also leads to a trivial solution. From the definition of the state space we know that,

$$\mathbb{E}[N \cdot E] = E_{Setup} \sum_{n=k}^{\infty} n\pi_{0,n} + E_{Busy} \sum_{n=0}^{\infty} n\pi_{1,n}.$$

Substituting (3)-(6) into the previous equations and simplifying, we arrive at,

$$\mathbb{E}[N \cdot E] = \mathbb{E}[(N \cdot E)_{M/M/1}] + E_{Setup}(1-\rho)\frac{\alpha\lambda(\lambda + k\gamma)}{\gamma(k\alpha\gamma + \alpha\lambda + \lambda\gamma)} + E_{Busy}\rho\frac{\alpha\lambda(\lambda + k\gamma)}{\gamma(k\alpha\gamma + \alpha\lambda + \lambda\gamma)}. \tag{13}$$

One should note that since we invoked the steady state equations, we have assumed underlying exponential distributions, and therefore are performing this analysis for an $M/M/1 \circ \{M, M, k\}$ system. However, this assumption will later be relaxed. The optimal value of α can be obtained by taking the the partial derivative of (13) and setting it equal to 0. Taking said derivative yields,

$$\frac{\partial}{\partial\alpha}\mathbb{E}[N \cdot E] = (\rho + r_{Setup}(1-\rho))\frac{\lambda(\lambda + k\gamma)}{\gamma}\left(\frac{\lambda\gamma}{(k\alpha\gamma + \alpha\lambda + \lambda\gamma)^2}\right).$$

This expression does not equal 0 for any value of α. This implies that the optimal value lies at one of the bounds for the feasible range of α. Furthermore, the derivative is always positive. Taking these two facts together, we know that $\mathbb{E}[N \cdot E]$ is always minimized when $\alpha = 0$. In other words, when $\mathbb{E}[N \cdot E]$ is the cost function, the optimal policy is one where the server always remains on.

With regards to the "always on" property of the $\mathbb{E}[E \cdot N]$ cost function, this becomes intuitively clear after several observations, and in fact can be generalized to $\mathbb{E}[E^{w2} \cdot N^{w3}]$.

1. It is known that in a stable system there is no avoiding being $BUSY$ for a proportion of time equal to the system utilization, ρ. Therefore, the energy cost of the system must equal E_{Busy} for a proportion ρ of the time.

2. It is observed that the expected number in the system while in $BUSY$ given that it transitioned from *IDLE*, is less than or equal to the expected number of jobs in the system while in $BUSY$ given that it transitioned from *SETUP*. This is due to the fact that transitioning from *IDLE* implies there is only one job in the system (at the time of the transition), while when transitioning from *SETUP* there are at least k jobs, as well as whatever jobs arrived during the setup time (mean number λ/γ).

3. Due to the two observations immediately preceding this, ignoring the addition of terms to $\mathbb{E}[N \cdot E]$ when the system is *IDLE*, *OFF*, and in *SETUP*, one cannot achieve a lower $\mathbb{E}[N \cdot E]$ than the policy which always keeps the server on.

4. It is noted that when the system is *IDLE*, $N = 0$, which implies $N \cdot E = 0$. Therefore from the third and fourth observation, one can conclude the policy which will always minimize $\mathbb{E}[N \cdot E]$ is the policy which always keeps the server on.

Furthermore, this reasoning can be extended to general underlying distributions and therefore, this always on property for $\mathbb{E}[E^{w2} \cdot N^{w3}]$ also holds in the $M/G/1 \circ \{G, G, k\}$ system.

From our previous observation of C in (12), we inspect cost functions of the form $\mathbb{E}[R^{w_1} \cdot E^{w_2} \cdot N^{w_3}]$. If $w_2 = 0$ it is obvious that keeping the server on will be optimal. Also, if $w_1, w_2, w_3 > 0$ the same trivial solution also arises, due to reasons mentioned before, specifically, when the server is idle the cost function is 0. Reducing the set of cost functions (12) by removing all trivial cases leaves us with

$$\mathbb{E}[R^{w_1} \cdot E^{w_2}].$$

Noting that $R = S + W$ where S denotes the processing time and W denotes the waiting time allows us to rewrite the previous equation as

$$\mathbb{E}[(S+W)^{w_1} \cdot E^{w_2}] = \mathbb{E}\left[\left(\sum_{i=0}^{w_1-1} \binom{w_1}{i} W^{w_1-i} \cdot S^i + S^{w_1}\right) \cdot E^{w_2}\right]. \tag{14}$$

Before completing the next step we note that due to $W = 0$ while the server is idle, we can conclude that all cost functions of the form $\mathbb{E}[W^{w_1} \cdot E^{w_2}]$ are also trivially solved using the same argument for that of $\mathbb{E}[N^{w_1} \cdot E^{w_2}]$. Next we let T equal all terms in (14) except for $S^{w_1} \cdot E^{w_2}$. We also note that the processing time of the next job to be processed is independent from the energy used, and therefore $\mathbb{E}[S^{w_1} \cdot E^{w_2}] = \mathbb{E}[S^{w_1}] \cdot \mathbb{E}[E^{w_2}]$. Substituting this observation and T into (14) gives us,

$$\mathbb{E}[R^{w_1} \cdot E^{w_2}] = \mathbb{E}[T] + \mathbb{E}[S]\mathbb{E}[E].$$

We know that T is a sum of terms of the form $W^m \cdot E^n$ and is therefore minimized when the server remains on. We also know from Theorem 2 that $\mathbb{E}[E]$ is minimized by leaving the server on, when $r_{idle} < \frac{\lambda}{k\gamma+\lambda} r_{setup}$. Therefore it is optimal to keep the server on when $r_{idle} < \frac{\lambda}{k\gamma+\lambda} r_{setup}$ under the cost function $\mathbb{E}[R^{w_1} \cdot E^{w_2}]$. $\square$

4. Applications

In this section, we derive optimal values for the parameters under popular optimization criteria, as well as demonstrating how these results can be used in other settings. We revert back to our model with exponential assumptions for simplification of calculations, however all methods used are still applicable in the general setting.

4.1. Weighted Sum Cost Function

One of the more popular metrics used is a weighted sum of the three system metrics, $\mathbb{E}[R] + \beta_1\mathbb{E}[E] + \beta_2\mathbb{E}[C]$ [2, 15, 16, 19]. Often $\mathbb{E}[C]$ is ignored ($\beta_2 = 0$) and the weights β_1 and β_2 appropriately scale the units of the overall function. This means of course that $\mathbb{E}[R]$ must be scaled by a unit constant. We take the partial derivatives first with respect to α.

$$\begin{aligned}
\frac{\partial}{\partial\alpha}\mathbb{E}[R] &= \frac{\lambda(\lambda+k\gamma)}{(k\alpha\gamma+\alpha\lambda+\lambda\gamma)^2} + \frac{\gamma^2 k(k-1)}{(k\alpha\gamma+\alpha\lambda+\lambda\gamma)^2} \\
\frac{\partial}{\partial\alpha}\mathbb{E}[E^N] &= (1-\rho)\lambda\gamma\frac{r_{setup}\lambda - r_{idle}(\lambda+k\gamma)}{(k\alpha\gamma+\alpha\lambda+\lambda\gamma)^2} \\
\frac{\partial}{\partial\alpha}\mathbb{E}[C] &= (1-\rho)\frac{\lambda^2\gamma^2}{(k\alpha\gamma+\alpha\lambda+\lambda\gamma)^2}
\end{aligned}$$

As expected, α only appears in the denominators. As was previously mentioned, this means that when we take the weighted sum of the derivatives, there is no value of α to make the sum evaluate to 0. In other words, the optimal value of α occurs at one of its bounds, $\alpha = 0$ or $\alpha \to \infty$. However, we already knew this from our previous observation on the Poisson process. What this yields that we did not have before is the point where the preference of α switches. From our cost function we can see that when the following inequality holds, the optimal value is to have $\alpha \to \infty$, while it is 0 otherwise.

$$\beta_1(1-\rho)\lambda\gamma r_{idle}(k\gamma+\lambda) \le \lambda(\lambda+k\gamma) + \gamma k(k-1) + \beta_1(1-\rho)\lambda^2\gamma r_{setup} + \beta_2(1-\rho)\lambda^2\gamma^2 \tag{15}$$

When solving for the optimal value of k, we can simplify by initially having $\alpha \to \infty$ since we know that if $\alpha = 0$ the choice of k is irrelevant since the server never shuts off. Taking the partial derivatives of the metrics with $\alpha \to \infty$ gives us,

$$\frac{\partial}{\partial k}\mathbb{E}[R] = \frac{\gamma}{2\lambda}\frac{k^2\gamma + 2k\lambda - \lambda}{(\lambda + k\gamma)^2},$$
$$\frac{\partial}{\partial k}\mathbb{E}[E^N] = -(1-\rho)\frac{\lambda\gamma r_{setup}}{(\lambda + k\gamma)^2},$$
$$\frac{\partial}{\partial k}\mathbb{E}[C] = -(1-\rho)\frac{\lambda\gamma^2}{(\lambda + k\gamma)^2}.$$

Setting the weighted sum of the above three terms equal to 0, we arrive at the following quadratic,

$$0 = \frac{\gamma^2}{2\lambda}k^2 + \gamma k - (\lambda + (1-\rho)\lambda\gamma(\beta_1 r_{setup} + \beta_2\gamma)). \tag{16}$$

Solving (16) and substituting into (15), one can determine the optimal values of the system parameters. If there exists a solution, k^*, for (16) on the constrained range of k, due to the convexity of our metrics with respect to k, one would just need to check both $\lceil k^* \rceil$ and $\lfloor k^* \rfloor$ to see which yields the best result.

4.2. Optimization with SLA Constraints

Here we consider a constrained optimization problem. We find that the optimal value of α is not necessarily at the bounds of its range. Imagine a server where for simplicity k is fixed at 1 and there is a service level agreement (SLA) that the expected response time for a job must be less than or equal to some constant T, where $\frac{1}{\mu-\lambda} \leq T \leq \frac{1}{\mu-\lambda} + \frac{1}{\gamma}$, and we wish to minimize the expected energy consumed by the system under the assumption that $r_{idle} < \frac{\lambda}{\lambda+\gamma}r_{setup}$. We set (9) equal to T and solve for α:

$$\alpha = \frac{\lambda\gamma^2}{\lambda+\gamma}\frac{T - \mathbb{E}[R_{M/M/1}]}{1 - \gamma(T - \mathbb{E}[R_{M/M/1}])}.$$

Using this value for α will minimize the expected energy used by the system. This value is optimal due to our assumption on r_{Idle}, which implies $\mathbb{E}[E]$ decreases as α increases.

4.3. Sleep States

Modern servers usually have several different discrete sleep settings which they can be set to. While in these sleep states, the server consumes a lower amount of energy than being idle but it cannot process jobs. We define a class of policies $\mathcal{P}$, which exhibit very similar behaviour to the polices we have been considering. Policies of class $\mathcal{P}$ wait for k jobs to accumulate in the queue while in a lower energy state before beginning to turn on. Once turned on the system processes jobs until it becomes idle. If the system idles for a certain amount of time before a new job arrives, it moves to the same lower energy state that it started in, and repeats its behaviour. The key difference here is now we have different lower energy states (the sleep states), and we allow the server to only use one of them. We show that our model can be used to find the optimal policy contained in $\mathcal{P}$.

We add the following variation to our previous model: the system now has I different sleep states it can be set to, where each of the i sleep states is denoted by $SLEEP_i$. As stated before, jobs cannot be processed while the server is in state $SLEEP_i$, $\forall i : 0 < i \leq I$. For each state $SLEEP_i$, there is a corresponding energy cost, denoted $E_{Sleep,i}$ (along with an energy ratio with respect to E_{Busy}, $r_{Sleep,i}$), as well as a corresponding setup time, with rate γ_i. Typically, $\forall i : 0 < i < I.E_{Sleep,i} \leq E_{Sleep,i+1}$ and $\gamma_i \leq \gamma_{i+1}$.

Our original model can describe a system where instead of turning off after a given idling time, it instead transitions to some state $SLEEP_i$. Here the steady state probabilities of $\pi^i_{0,0}$ to $\pi^i_{0,k-1}$ now correspond to the steady state probabilities of being in state $SLEEP_i$ rather than OFF, and γ is replaced with γ_i. To analyse this system, we must also replace each instance of γ in our equations for $\mathbb{E}[R]$, $\mathbb{E}[E^N]$, and $\mathbb{E}[C]$

by γ_i as well as make a slight addition to the expression for $\mathbb{E}[E^N]$, (8), to account for energy now being consumed in the sleep state,

$$\mathbb{E}[E^N_{Sleep,i}] = \mathbb{E}[E^N] + (1-\rho)\frac{k\alpha\gamma_i}{k\alpha\gamma_i + \alpha\lambda + \lambda\gamma_i} r_{Sleep,i}.$$

From here we can analyse the system, and obtain the optimal values of α and k. Substituting these values into our optimization metric gives us some value, denoted opt_i. Once we have these I optimal values as well as the optimal value for the server turning off, we can take the minimum of them and design our policy to always transition to the corresponding energy state *OFF*, or $SLEEP_i$.

Although accounting for the sleep states of the server allows us to determine improved policies than if we were to ignore them, we can no longer claim that our model can describe the optimal policy of the server, i.e. the optimal policy may not be contained in $\mathcal{P}$. This is due to the fact that the optimal policy may have the server be in some sleep state until k_1 jobs accumulate, then move to a higher sleep state where it waits for k_2 jobs to accumulate before turning on. However, when the optimal values of k are low for any individual sleep state under our analysis, we conjecture that the policy will be close to optimal. For a more in depth analysis of this model under multiple sleep states, we refer the reader to [10].

5. Random Routing

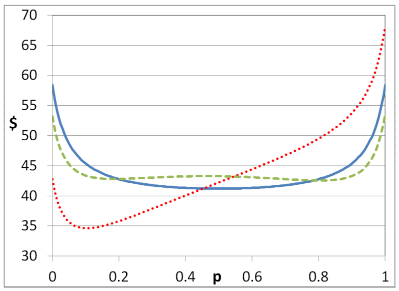
Here we present an application of our model in a random routing setting, where we leverage our single server solutions. Imagine a system with two $M/M/1 \circ \{M, M, k\}$ queues. When a job arrives to the system, it is sent to the first queue with probability p and is sent to the second queue with probability $(1-p)$. If we wish to optimize for some metric, we now have five decision variables, α_1, α_2, k_1, k_2, and p, where the subscripts 1 and 2 denote the values for the first and second server, respectively. We know that the values for α_1 and α_2 will be either set to 0 or approach ∞, which breaks the problem set into three cases (due to symmetry) where we instead look to optimize against k_1, k_2 and p and then take the lowest value from among the three cases. We classify the cases as follows. The first is $\alpha_1 = \alpha_2 = 0$, the second is $\alpha_1 \to \infty$ and $\alpha_2 = 0$, and the third is $\alpha_1 \to \infty$ and $\alpha_2 \to \infty$.

We wish to minimize $\mathbb{E}[N] + \beta\mathbb{E}[E]$. This falls within our class of cost functions, as $\mathbb{E}[N]$ is a scaled version of $\mathbb{E}[R]$ (by Little's Law). We know that for the first case since the servers will always be on and each server will be $BUSY$ for $\frac{p\lambda}{\mu}$ and $\frac{(1-p)\lambda}{\mu}$ proportion of time respectively, that the optimal configuration in that case is to set $p = 0.5$, i.e. balance the loads. As we will see, the other cases provide non-trivial optimal values for p.

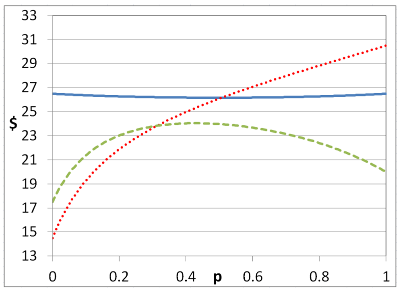
Observation 5. *In multiple server systems where energy provisioning is a concern, traditional load balancing in general is not optimal.*

Figure 6 shows several examples under different parameter configurations of the cost function versus p in the three different cases where the optimal k values are used, and r_{Idle} and r_{Setup} are both set to 0.8. In Figure 2(a), we see a medium loaded system where either server could accept all of the arrivals and still be stable. Here we can see that the optimal server configuration is to have a server which is always on which takes the majority of the system load (89.5%), while a server which turns off when it becomes idle takes a small portion of the system load (10.5%). This means that a lot of the time, the server that turns off will just remain off with up to four jobs waiting in the queue. This may seem unfair to the jobs which are "unlucky" enough to be put into this queue but this is an unfortunate side effect of the cost function being considered.

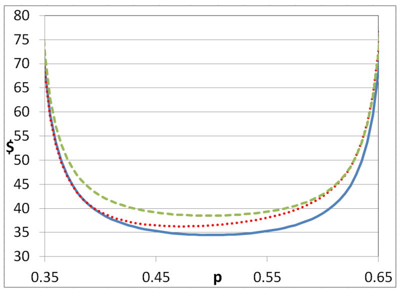
In Figure 6-(b), we see a lightly loaded system and get a result that is not surprising. The optimal configuration is still one server that remains on and one that turns off. However, the server which turns on and off is completely ignored. In other words, the configuration which optimizes the random routing problem is simply an M/M/1 queue. This is somewhat expected since the load on the system is so light it is not advantageous to use the second server.



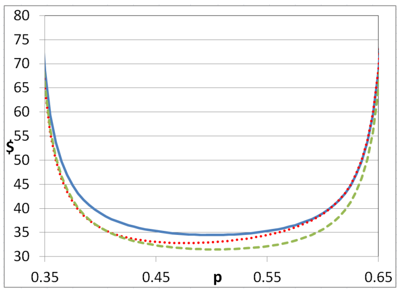
(a) $\mu = 2, \lambda = 1.9, \gamma = 0.2, \beta = 22, k_1 = 5, k_2 = 7$



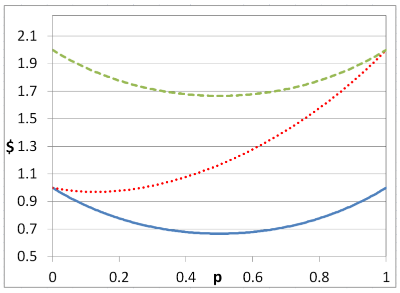
(b) $\mu = 2, \lambda = 1, \gamma = 0.2, \beta = 15, k_1 = 1, k_2 = 4$



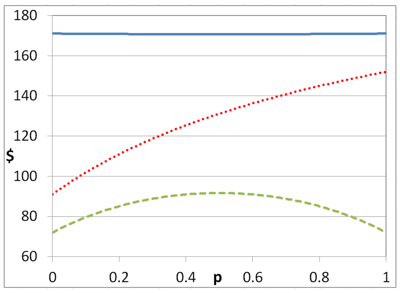
(c) $\mu = 2, \lambda = 3, \gamma = 0.5, \beta = 15, k_1 = 3, k_2 = 2$



(d) $\mu = 2, \lambda = 1.9, \gamma = 3, \beta = 15, k_1 = 1, k_2 = 2$



(e) $\mu = 2, \lambda = 1, \gamma = 1, \beta = 0, k_1 = 1, k_2 = 1$



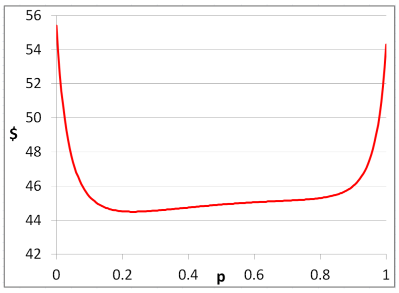
(f) $\mu = 2, \lambda = 1, \gamma = 1, \beta = 100, k_1 = 1, k_2 = 1$

$\alpha_1 = 0, \alpha_2 = 0$	$\alpha_1 \to \infty, \alpha_2 = 0$	$\alpha_1 \to \infty, \alpha_2 \to \infty$
━━━	········	‒ ‒ ‒

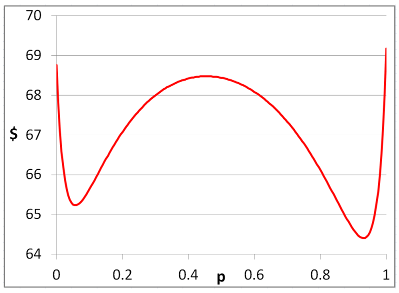
Figure 6: Random routing: cost vs p for three different configurations of a two server system

Figures 6-(c) and (d) show the results for a heavily loaded system where both servers must be used or the system will be unstable. We can see the curves of the three cases approach similar curvatures. In Figure 2(c), where the setup rate is relatively low ($\gamma = 0.5$), the classical load balancing approach gives us the best configuration with both servers always on and $p = 0.5$. We notice that as we increase the setup rate of the server ($\gamma = 3$), both servers being on becomes sub-optimal and the case of both servers turning on and off begins to dominate. In fact, the optimal value is $p = 0.505$ and not $p = 0.5$ as one might expect. This is as we would expect since the faster the server can turn on, the more appealing it is to shut it off.

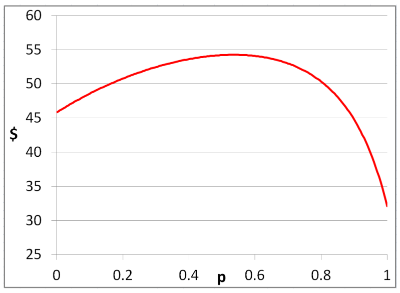
As we see from Figure 6, simple load balancing is not sufficient to arrive at the optimal configuration as we have shown non-trivial values of p that optimize the system. Taking a narrower look at the single case of having both servers able to turn off in Figure 7, shows a similar non-trivial result. Here the graphs also become asymmetric with respect to p, and even the optimal values of k_1 and k_2 are not equal. As in the case of having one server always on, and one server able to turn off, load balancing is not optimal. It is noted that if load balancing were used in Figure 7-(b), i.e. $p = 0.5$, the result would be a disaster, as it is one of the worst configurations possible in this context. Adding energy concerns to these systems greatly impacts the complexity of the analysis as typical load balancing algorithms are no longer optimal. This also raises questions on the implications for other multi-server settings such as round robin routing or in an $M/M/c \circ \{M, M, k\}$ queue. Specifically, there is no reason why in general each server should be homogeneous with respect to the server's α and k values.



(a) $\mu = 2, \lambda = 1.9, \gamma = 0.1, \beta = 15, k_1 = 6, k_2 = 3$



(b) $\mu = 2, \lambda = 1.9, \gamma = 0.1, \beta = 30, k_1 = 3, k_2 = 4$



(c) $\mu = 2, \lambda = 0.5, \gamma = 0.1, \beta = 50, k_1 = 7, k_2 = 1$

Figure 7: Random routing: single case (both servers turn off)

6. Conclusion

As energy costs of servers as well as the relative energy consumed by servers increase, we must put greater emphasis on determining optimal policies. Here we gave a complete analysis of the single server systems $M/M/1 \circ \{M, M, k\}$ and $M/G/1 \circ \{G, M, k\}$, with respect to $\mathbb{E}[N]$, $\mathbb{E}[R]$, $\mathbb{E}[E]$, and $\mathbb{E}[C]$ as well as analysis for an $M/G/1 \circ \{G, G, k\}$ queue with respect to $\mathbb{E}[E]$ and $\mathbb{E}[C]$. This gave us an array of tools and equations to arrive at optimal policies for many single server energy-aware systems under general settings. This analysis allowed us to make several interesting and unexpected observations which give a greater clarity to how these systems behave. We also leveraged our analysis in several other applications, such as SLA optimization, servers with sleep states, and a multi-server systems with random routing. For the latter we showed that typical load balancing algorithms are not enough to arrive at an optimal configuration. Furthermore, this context gives a deeper insight into the analysis of these energy-aware multi-server systems with other routing policies. In particular, heterogeneous servers may be desirable, in contrast to models where energy costs are not considered.

For our future work we plan on extending some of our analysis methods to multi-server systems in hopes at arriving at the optimal policy. However, this is a much greater challenge as many trivialities in the single server system, such as the form of the optimal policy, do not carry over. Specifically, when determining when it is optimal start a setup or turn a server off, one would need to consider the behaviour of the other servers, and potential future states the system could transition to.

Acknowledgement. This research was funded by the Natural Sciences and Engineering Research Council of Canada.

7. References

[1] J. R. Artalejo. A unified cost function for M/G/1 queueing systems with removable server. *Trabajos de Investigacion Operativa*, 7(1):95–104, 1992.

[2] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. *SIGMETRICS Performance Evaluation Review*, 33(1):303–314, June 2005.

[3] G. Choudhury and H. K. Baruah. Analysis of a Poisson queue with a threshold policy and a grand vacation process: an analytic approach. *The Indian Journal of Statistics, Series B*, 62(2):303–316, 2000.

[4] U. EPA. Report to congress on server and data center energy efficiency. Technical report, U.S Environmental Protection Agency, 2007.

[5] S. W. Fuhrmann and R. B. Cooper. Stochastic decompositions in the M/G/1 queue with generalized vacations. *Operations Research*, 33:1117–1129, 1985.

[6] A. Gandhi, S. Doroudi, M. Harchol-Balter, and A. Scheller-Wolf. Exact analysis of the M/M/k/setup class of markov chains via recursive renewal reward. In *ACM SIGMETRICS*, 2013.

[7] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Performance Evaluation*, 67(11):1155–1171, Nov. 2010.

[8] A. Gandhi and M. Harchol-Balter. M/M/k with exponential setup. Technical report, Carnegie Mellon University, 2010.

[9] A. Gandhi, M. Harchol-Balter, and I. Adan. Server farms with setup costs. *Performance Evaluation*, 67(11):1123–1138, Nov. 2010.

[10] M. E. Gebrehiwot, S. Aalto, and P. Lassila. Optimal sleep-state control of energy-aware M/G/1 queues.

[11] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew. Greening geographical load balancing. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS, pages 233–244, 2011.

[12] V. J. Maccio. On optimal policies for energy-aware servers. Master's thesis, McMaster University, 2013.

[13] V. J. Maccio and D. G. Down. On optimal policies for energy-aware servers. In *Proceedings of the IEEE 21st International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2013)*, pages 31–39, Aug 2013.

[14] M. Mazzucco and D. Dyachuk. Optimizing cloud providers revenues via energy efficient server allocation. *Sustainable Computing: Informatics and Systems*, 2(1):1–12, 2012.

[15] A. Penttinen, E. Hyytia, and S. Aalto. Energy-aware dispatching in parallel queues with on-off energy consumption. In *IEEE International Performance Computing and Communications Conference*, pages 1–8, November 2011.

[16] J. Slegers, N. Thomas, and I. Mitrani. Dynamic server allocation for power and performance. In *Proceedings of the SPEC International Workshop on Performance Evaluation: Metrics, Models and Benchmarks*, SIPEW '08, pages 247–261, Berlin, Heidelberg, 2008. Springer-Verlag.

[17] N. Tian and Z. G. Zhang. *Vacation Queueing Models Theory and Applications.* Springer Science, 2006.

[18] A. Wierman, L. L. H. Andrew, and M. Lin. *Handbook on Energy-Aware and Green Computing*, chapter Speed Scaling: An Algorithmic Perspective, pages 385–406. CRC Press, 2012.

[19] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems. In *Proceedings of INFOCOM*, 2009.

[20] X. Xu and N. Tian. The M/M/c queue with (e, d) setup time. *Journal of Systems Science and Complexity*, 21:446–455, 2008.