Journal of Universal Computer Science, vol. 21, no. 13 (2015), 1790-1809
submitted: 4/5/15, accepted: 6/7/15, appeared: 28/12/15

Queuing Theory-based Latency/Power Tradeoff Models for Replicated Search Engines

Ana Freire
(University of A Coruña, A Coruña, Spain
ana.freire@udc.es)

Craig Macdonald
(University of Glasgow, Glasgow, United Kingdom
craig.macdonald@glasgow.ac.uk)

Nicola Tonellotto
(National Research Council of Italy, Pisa, Italy
nicola.tonellotto@isti.cnr.it)

Iadh Ounis
(University of Glasgow, Glasgow, United Kingdom
iadh.ounis@glasgow.ac.uk)

Fidel Cacheda
(University of A Coruña, A Coruña, Spain
fidel.cacheda@udc.es)

Abstract: Large-scale search engines are built upon huge infrastructures involving thousands of computers in order to achieve fast response times. In contrast, the energy consumed (and hence the financial cost) is also high, leading to environmental damage.

This paper proposes new approaches to increase energy and financial savings in large-scale search engines, while maintaining good query response times. We aim to improve current state-of-the-art models used for balancing power and latency, by integrating new advanced features. On one hand, we propose to improve the power savings by completely powering down the query servers that are not necessary when the load of the system is low. Besides, we consider energy rates into the model formulation. On the other hand, we focus on how to accurately estimate the latency of the whole system by means of Queueing Theory.

Experiments using actual query logs attest the high energy (and financial) savings regarding current baselines. To the best of our knowledge, this is the first paper in successfully applying stationary Queueing Theory models to estimate the latency in a large-scale search engine.

Key Words: Information Retrieval, Search Engines, Power Consumption, Green IR, Queueing Theory

Category: H.3.3, H.3.4, G.3

1 Introduction

Large-scale search engines consume high amounts of energy to sustain the necessary infrastructure to handle the incoming query traffic. To give an idea of this power consumption, the energy consumed by Google in 2013 was reported to be 3,712,865 MWh [Google, 2015]. The power consumption is directly related to electricity costs, forming an important part of the operational costs of search engine companies [Hoelzle and Barroso, 2009].

The incoming query flow received by a commercial search engine varies through the course of the day [Silvestri, 2010]. To provide sub-second latencies to user queries, search engines deploy multiple replicas of their data on several machines, distributing the incoming queries among such machines and processing them concurrently. Thus, it is natural to adapt the resources of a search engine according to the variations of the query load: maintaining acceptable query latencies while minimising the number of machines used to process the queries.

[Freire et al., 2014a] proposed a mathematical model for replicated search systems that establishes a trade-off between latency and power consumption in terms of the number of replicated query servers required as query load varies throughout the day. When the incoming query traffic is high, the model automatically activates the number of necessary machines in the system. In the contrary, when the number of incoming query traffic decreases, the system automatically switches some machines to a STANDBY state, leading to power savings. The decision about how many machines should be turned ON/STANDBY is taken considering previous and current query traffic, at the same time it balances query latency and power consumption. As these factors are key in the process, we will study both of them in order to achieve better performance.

Therefore, the main contributions of this paper are as follows:

1. We propose a **new power cost function** that allows the model to completely turn off the unnecessary machines, in order to achieve higher power savings while maintaining good latency values. This function also allows the model to be price-driven, by including hourly variable electricity rates into the model.

2. We model the search engine behaviour with a queueing system and we use quereing theory [Cooper, 2003] to propose a **new latency cost function** to estimate the waiting time of queries. We aim to study how this model work into a large-scale search engine. To the best of our knowledge, this is the first time that stationary models (see Section 3.3.2) are applied in representing the latency of search engines.

This paper is structured as follows: Section 2 discusses the existing literature on *Green Information Technologies* focusing on *Green Information Retrieval*. In

Section 3, we describe in depth the already published dynamic system and the general cost function, noting the main weaknesses this paper addresses. Section 4 proposes the new cost functions, including the new latency cost function based on Queueing Theory (QT) as well as the new power cost function considering energy pricing and OFF state of servers. In Section 5, we concretely state the research questions that we investigate, as well as detailing the baselines and experimental setup. Section 6 reports our experimental results, with concluding remarks following in Section 7.

2 Green Information Retrieval

Several efforts have been made in the existing literature for reducing the energy consumption of general-purpose data centres. In 2005, [Mastroleon et al., 2005] defined a mathematical model that varies the utilized CPUs for job processing by establishing a trade-off between the load of the system and the power consumed. A step forward was achieved by [Economou et al., 2006], who studied component-level (i.e. CPU, memory and disk) power consumption, and then developed a model – named Mantis – for predicting temporal variations in power, as well as peak and average values. The approach of power calculation at component-level is also followed by [Khargharia et al., 2008], where the authors proposed a framework and methodology for autonomic power and performance management in data centres.

Although general-purpose data centres present similar features to search engines such as natural fluctuations and spikes, search engines have particular characteristics (i.e.: query distribution and frequency) that necessitates separate research on the sustainability of Information Retrieval datacentres. The term Green IR was firstly introduced in 2012 by [Chowdhury, 2012] with the aim of encouraging the building of sustainable IR systems.

However, few Green IR works have been published. In [Kayaaslan et al., 2011] authors care about power savings by distributing queries between geographically distant data centres based on workload and electricity prices. Recently, [Sazoglu et al., 2013] propose a novel metric for result caching that considers the financial cost of a cache miss.

In 2014, [Freire et al., 2014a] proposed the first intra-data centre model that turns the servers on or standby depending on the incoming query traffic needs. This model can examine the historical and current query traffic patterns to predict the number of query server replicas now needed, and obtain power savings within a single data centre by eliminating query servers that are not currently needed. Later in 2014, [Freire et al., 2014b] proposed the use of Queueing Theory into their model to represent the latency of the whole system, but the results attested that QT was not suitable for this kind of scenario.

Our contribution aims to solve the weaknesses of both versions of the self-adapting model ([Freire et al., 2014a, Freire et al., 2014b]) by re-defining their power and latency cost functions. The next section describes the self-adapting model in order to introduce later the disadvantages we try to address.

3 Dynamic Optimisation Model

The mathematical model proposed in [Freire et al., 2014a] establishes a trade-off between latency and power into a replicated search engine processing user queries. This approach splits the current day into slots of 15 minutes and does likewise for a previous day (usually the same day of the previous week). Based on historical data, the model estimates the number of queries that will arrive in the system at each slot of the day and computes the necessary number of machines to process those queries in a timely fashion. This way, when the load of the system is expected to be low, some machines are turned to a STANDBY state. On the other hand, when the query traffic increases, the system automatically turns on the necessary number of machines to maintain acceptable response times.

In the remainder of this section, we provide a short introduction to dynamic model formulation to later introduce the improvements we propose.

3.1 General Definition

The self-adapting mathematical model for reducing the power consumption of a search engine consists in defining the following global cost function (Eq. 1) and the objective is to minimize this function at each slot of the day (i.e. at the beginning of each slot k, the number of machines (u_k) is chosen to minimize the function $g(u_k)$):

$$g(u_k) = \lambda P(u_k) + (1 - \lambda)L(u_k) \tag{1}$$

where $\lambda \in [0, 1)$ is the trade-off parameter and varies its value between $[0, 1)$. For $\lambda = 0$, the cost function represented by Equation (1) ignores any power cost, and leads to the maximum number of available processing nodes being used in every time slot. For $\lambda = 1$, the cost function ignores any latency cost, maximising power savings but leading to infinite waiting times. Varying λ in $[0, 1)$, we can achieve any average query latency from infinite to the minimum possible traded off against the corresponding power consumption of the search system. We note that both cost functions assume values in the same range. Without loss of generality, later in this section, we devise particular cost functions ranging in the [0,1] interval, where 0 means no cost and 1 means maximum cost.

$P(u_k)$ represents the power cost function, depending on the maximum number of replicas in the system M.

Lastly, $L(u_k)$ is a cost function to represent the latency of the search engine in responding to queries. In [Freire et al., 2014a], the authors represented the latency using a deterministic approach. Instead, in [Freire et al., 2014b], the authors define $L(u_k)$ by using Queueing Theory.

The next sections describe in detail both power and latency cost functions.

3.2 Power Cost Function

The power cost function represents the electric power consumption of the whole search engine and it is directly proportional to the energy costs of operating the search engine. [Freire et al., 2014a] distinguishes between three states that a node can be in:

1. ON. The node is busy processing a query and consumes power at a rate of P_{on}.
2. STANDBY. The node is available, but is currently sleeping. The node consumes power at a rate of P_{standby}.
3. OFF. The node is off, and it consumes no power (we will later consider that OFF state does consume power, although almost negligible).

Given these costs, at a given time slot k, the total energy consumed by a search engine with u_k active processing nodes out of a possible M is:

$$P_{\text{on}} T_s u_k + P_{\text{standby}} T_s (M - u_k)$$

By normalising this quantity by the maximum consumable energy for M machines, we obtain the following expression for the power cost function $P_k(\cdot)$:

$$P_k(\cdot) = P(u_k) = \frac{1}{M P_{\text{on}}} \left[P_{\text{on}} u_k + P_{\text{standby}} (M - u_k) \right] \tag{2}$$

While the switching time ON $\leftrightarrow$ STANDBY is almost instantaneous, the time required to switch between ON and OFF and vice-versa is not negligible for most data centres [Gandhi and Harchol-Balter, 2011]. Both [Freire et al., 2014a, Freire et al., 2014b] avoid the use of OFF state as they argue that the switching time can negatively impact on the latency of the queries to be processed by the node. However, [Liu et al., 2009], showed that the most effective and aggressive power saving comes from turning off components that are not used, such as CPU, disk, and memory, which consume substantial power when they are turned on, even with no active workload. This way, it should be interesting to study a way of how to power off the servers without impacting the latency.

3.3 Latency Cost Function

3.3.1 Deterministic Approach

The latency cost function proposed by [Freire et al., 2014a] represents the cost incurred when the time required to process queries increases. In order to provide a simple analytic expression for this cost, they consider the following situation: at the beginning of time slot k, we have x_k queued queries, waiting to be processed by u_k nodes with an average service time per node of $\bar{v}_k$ seconds. During the k-th time slot, we receive $\bar{w}_k$ new queries to process. We want to compute the average latency of $x_k + \bar{w}_k$ queries. The first batch of u_k queries can be processed by a single replica after $\bar{v}_k$ seconds, the second batch of u_k queries is processed after $2\bar{v}_k$ seconds, and so on. We have a total of $B = (x_k + \bar{w}_k)/u_k$ batches of queries, so the last batch of at most u_k queries is processed after $B\bar{v}_k$ seconds. Hence, at a given time slot k, the query completion time T_k of $x_k + \bar{w}_k$ queries by u_k replicas can be computed by:

$$T_k = \frac{x_k + \bar{w}_k}{u_k}\bar{v}_k \tag{3}$$

To normalise this completion time in the [0,1] interval, we adapt the latency metric from [Wang et al., 2010]:

$$L_k(\cdot) = L(u_k) = 1 - \exp(\alpha T_k) \tag{4}$$

[Freire et al., 2014a] considered different approaches for the number of queries arriving during the kth time slot. The first one, called LongTerm, estimates the number of incoming queries with the actual number of incoming queries in the same time slot of a previous day, i.e.: $\bar{w}_k = w_{k-SN}$ (where S represent the number of previous days, and N is the total number of time slots in a day). The second one, called ShortTerm, adjusts that value with the current trend of arrivals [Radinsky et al., 2012] experienced in the last two time slots, such that:

$$\bar{w}_k = w_{k-SN} + (w_{k-1} - w_{k-2})$$

We will focus only on ShortTerm, as it demonstrated higher power savings.

3.3.2 Queueing Theory Approach

Recently, Queueing Theory has been increasingly used for achieving energy savings in data centres and for estimating their workload [Jeon and Prabhu, 2013, Meisner et al., 2011, Parolini et al., 2008]. Queueing Theory is considered one of the standard methodologies (together with linear programming, simulation, etc.) of operations research and management science and is standard fare in academic programs in industrial engineering, telecommunications or computer science. For this reason, [Freire et al., 2014b] addressed the following research question: *Is Queueing Theory suitable for representing the latency of a search engine in order to achieve power savings?*. Their aim was to modify the latency function proposed by [Freire et al., 2014a] looking for a more general and formal way of representing the queries' waiting time, by using well-proven QT models (instead of just considering historical data). In order to portray the results of the previous research, we now introduce some basic concepts about Queueing Theory.

In general, a queue can be defined as a waiting line (like customers waiting at a bank office) [Cooper, 2003]. Queueing Theory deals with the analysis of waiting lines where customers wait to receive a service [Bunday, 1996, Cao, 2002]. More generally, Queueing Theory is concerned with the mathematical modeling and analysis of systems that provide service to random demands. A queueing model is an abstract description of such a system.

A general queueing model can be mainly characterized by the following parameters (later it will be instantiated for an seach engine):

- Arrival rate (λ): mean number of arrivals per time unit. Interarrival rate: $\frac{1}{\lambda}$.
- Service rate (μ): mean number of customers that are served per time unit.
- Service capacity (s): number of servers helping the customers.
- Service discipline: First Come First Served (FCFS), Random, Last Come First Served (LCFS), etc.

Kendall [Kendall, 1953] introduced a shorthand notation to characterize a range of these queueing models. Its simplest form consists of a three-part code in the form $a/b/c$. The first letter specifies the interarrival rate distribution and the second one the service rate distribution. For example, for a exponential distribution the letter M is used, and D for deterministic times. The third and last letter specifies the number of servers. Some examples are $M/M/1$ or $M/D/2$. [Freire et al., 2014b] showed that a search engine could be represented by means of a $M/M/s$ model.

The universal notation of Queueing Theory also includes the following parameters, instantiated as follows for $M/M/s$ model:

- Service time:
$$\rho = \frac{\lambda}{s \cdot \mu} \tag{5}$$
  If $\rho < 1$ the system is said to be stationary and the model is able to calculate a solution based on the following formulas. Otherwise, the system is said to be non-stationary and no solution can be found.

– Probability of n customers to be in the system (in a stationary state) (P_n).

$$p_0 = (\sum_{n=0}^{s-1} \frac{(\frac{\lambda}{\mu})^n}{n!} + \frac{(\frac{\lambda}{\mu})^s}{s!(1-\rho)})^{-1} \tag{6}$$

$$p_n = \frac{(\frac{\lambda}{\mu})^n \cdot p_0}{n!}, 0 \leq n \leq s \qquad p_n = \frac{(\frac{\lambda}{\mu})^n \cdot p_0}{s! s^{n-s}}, n > s \tag{7}$$

– Estimated number of customers in the queue (L_q).

$$L_q = \frac{(\frac{\lambda}{\mu})^s \cdot p_0 \cdot \rho}{s! \cdot (1-\rho)^2} \tag{8}$$

– Mean waiting time in the system (W).

$$W = W_q + \frac{1}{\mu} \tag{9}$$

– Mean waiting time in the queue (W_q).

$$W_q = \frac{L_q}{\lambda} \tag{10}$$

Eq. 9 allows us to calculate the mean waiting time in the system based on the values of λ and μ. As the latency function represents the time that the system will spend in solving all the queries within a slot, latency (normalized in the [0,1] interval) is formulated as follows:

$$L_k(\cdot) = L(u_k) = 1 - e^{-(W \cdot T_s \cdot \bar{w}_k / u_k)} \tag{11}$$

[Freire et al., 2014b] concluded that the previous formulation has some deficiencies for calculating the latency of a large-scale search engine. These systems receive a high amount of queries that the usual model of Queueing Theory can not deal with. In periods of high contention, the system achieves a non-stationary state and the formulas of Queueing Theory are not able to compute the waiting time in the system. They addressed this problem by turning on all the machines in the system. The benefit of this approach is that the system always achieves good response times. Nevertheless, their proposed approach is not able to reach the energy savings of the previous deterministic approach.

4 Proposals

We argue that the main deficiencies of previous approaches in the following items:

- **Partial disconnection of servers**. The two previous works studied in this paper ([Freire et al., 2014a, Freire et al., 2014b]) considered only two possible states for the nodes: ON and STANDBY. They did not consider the OFF state to avoid dealing with the delay occurred at switching on a machine after powering off (we assume no delay in the rest of transitions). However, as we indicated before, Lin et al. [Liu et al., 2009], showed that the most effective and aggressive power saving comes from turning off components that are not used, such as CPU, disk, and memory, which consume substantial power when they are turned on, even with no active workload. Therefore, we will study how switching the machines off when they are not needed can lead to power and financial savings.

- **Non-financial models**. None of the previous works that studied the same model [Freire et al., 2014a, Freire et al., 2014b] considered the energy rates as a parameter of the system. They only based the power cost in kWh. However, recent works have attested that the electricity rates are a key factor when taking decisions regarding the management of a search engine [Kayaaslan et al., 2011]. The fact that many data centers adopt hourly variable electricity rates instead of constant prices, makes us think about the importance of considering this factor into the model.

- **Not suitable Queueing Theory models**. [Freire et al., 2014b] demonstrated that Queueing Theory could not be used to estimate the latency in large-scale search engines. Applying the $M/M/s$ model lead to a non-stationary state ($\rho > 1$) when the arrival rate (incoming query traffic) exceeds the server capacity (active machines in the system). As this situation happens at most of the time slots of the day, they conclude that QT was not a suitable approach for these scenarios.

In the remainder of this section we explain in detail our main contributions (new power and latency functions) that aim to solve the previous aspects.

4.1 Powering off the servers and considering energy rates

The power cost defined in Eq. 2 needs to be changed in order to consider the total powering off of the machines and the hourly-variable energy rates.

At the beginning of each slot, if the system detects that some of the active machines in the system are not necessary, it automatically turns them off just after they finish processing the current queries. All the queries waiting to be processed are scheduled to the remaining active replicas. The power in this slot will count for the power consumed by the ON machines (P_{on}) and also by the ones switched OFF (P_{off}), as it is not zero (see Eq. 12).

In the contrary, when the system needs to activate new machines, it automatically orders the booting process and when the machines are ON (after the booting delay d), they start receiving new queries. In these slots the power will be calculated as the sum of the power consumed by the ON and OFF machines, but it also considers the specific power consuming while booting the new machines P_{boot}.

$$P(u_k) = \begin{cases} P_{\text{on}}(u_{k-1}d + u_k(T_s - d)) + P_{\text{boot}}(u_k - u_{k-1})d + P_{\text{off}}(M - u_k)T_s & u_k > u_{k-1} \\ P_{\text{on}}u_kT_s + P_{\text{off}}(M - u_k)T_s & u_k <= u_{k-1} \end{cases} \tag{12}$$

Eq. 13 represents the normalized equation of Eq. 12.

$$P_{norm}(u_k) = P(u_k)/(MP_{on}(T_s - d) + M \cdot P_{boot} \cdot d) \tag{13}$$

Now, in order to consider hourly variable electricity rates, Eq. 12 will be replaced by Eq. 14 (and its normalized version - Eq. 15).

$$Price(u_k) = price_k \cdot P(u_k) \tag{14}$$

$$Price_{norm}(u_k) = Price(u_k)/maxPrize \tag{15}$$

4.2 Modeling the latency using stationary Queueing Theory models

In solving the problem of non-stationary general queueing systems, Stolletz [Stolletz, 2008] demonstrated how the so-called Stationary Backlog-Carryover approach is applicable to systems that frequently reach an overload state. This technique splits the time scale into small slots and applies a dependent stationary queueing model to each slot. The method is called the **stationary backlog-carryover (SBC)** approach, as a backlog b_i of work is measured in each period i (k in our scenario) and carried over into future periods $j > i$. This approximation allows queues to build up in overloaded periods and waiting jobs (queries in our scenario) can be transferred to a subsequent period.

Our aim is to apply this method into the power/latency model in order to achieve a formal way of estimating the latency of the system using Queueing Theory. This way, we will re-define the latency cost function.

The stationary backlog-carryover (SBC) approach can be divided into two main steps (note: as the formulation is straightforward to follow, we use the notation from the general definition in [Stolletz, 2008]):

– Approximation of the expected utilization. In order to get a constant arrival and service rate as well as a constant number of servers in each period i, the original time-dependent arrival rate function $\lambda(t)$ is replaced by a constant one (Eq.16):

$$\lambda_i = \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t} \lambda(s)ds \quad \forall i \in 1 \ldots T \tag{16}$$

And thus, the same applies to the service rates μ_i (Eq. 17):

$$\mu_i = \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t} \mu(s)ds \quad \forall i \in 1 \ldots T \tag{17}$$

Then, each period is approximated by means of a $M/M/s_i/s_i$ Erlang-loss system [Gross and Harris, 1985] using an artificial arrival rate:

$$\tilde{\lambda}_i = \lambda_i + b_{i-1} = \lambda_i + \tilde{\lambda}_{i-1} \cdot P_{i-1}(B) \tag{18}$$

where λ_i is the original arrival rate, b_{i-1} corresponds to the backlog generated through artificial blocking in period $i-1$ and $P_{i-1}(B)$ represents the steady-state probability of blocking for the $M/M/c_{i-1}/c_{i-1}$ model in $i-1$ with arrival rate λ_{i-1}.

As we noted before, this approach allows to serve waiting customers (queries) from period $i-1$ in the next period i and customers can arrive continuously. Applying Erlang's loss formula, we get:

$$b_i = \tilde{\lambda}_i \cdot P(B) = \tilde{\lambda}_i \cdot \frac{(\tilde{\lambda}_i/\mu_i)^{c_i}}{c_i! \sum_{k=0}^{c_i} \frac{(\tilde{\lambda}_i/\mu_i)^k}{k!}} \tag{19}$$

Hence, the expected utilization $E[U_i]$ would be:

$$E[U_i] = \frac{\lambda_i + b_{i-1} - b_i}{c_i \mu_i} \tag{20}$$

– Approximation of the time-dependent expected number of customers in the system and the expected queue length. In order to calculate expected queue lengths, a *modified arrival rate* (MAR) should be defined. This new measure will analyse a $M/M/s_i/\inf$ waiting model with the same utilization as the previous loss model. After some derivations, the MAR follows this equation:

$$\lambda_i^{MAR} = \lambda_i + b_{i-1} - b_i \tag{21}$$

Once we have calculated the previously formulated λ_i^{MAR}, we will use this value instead of the classical λ to obtain the mean waiting time in the system W (see Eqs. (5-11)).

5 Experimental Setup

In the next section, we experimentally investigate to determine the suitability of our new power and latency functions in reducing the power consumption of a search engine without negatively impacting on its efficiency. In particular, the following research questions are addressed:

1. Power cost function: Can we achieve higher power savings by totally turning some machines OFF instead of just turning them STANDBY and by using electricity rates and conduct the performance based on financial savings?

2. Latency cost function: Can Queueing Theory be used to estimate the latency of a large-scale search engine?

In the remainder of this section, we define the experimental setup to address these research questions, covering the different methods and baselines we are going to compare (Section 5.1), the evaluation measures (Section 5.2), the query logs used (Section 5.3) and other parameter settings (Section 5.4).

5.1 Baselines and proposed methods

5.1.1 Baselines

We select three reasonable baselines for determining how many machines are active at any slot:

– **Naïve**: consists in choosing the maximum number M of machines in each time slot. It is the same *Naïve* baseline proposed by [Freire et al., 2014a].

– **Threshold$**: consists in fixing a time threshold for the query completion times and derive the decisions u_k for $i = 0, \ldots, N-1$ that will be applied to the current day based on the previous day average arrival times, the average processing times and the number of queued queries in the same time slot. It is based on the *Threshold* baseline proposed by Freire et al.[Freire et al., 2014a], but we added the feature of completely switching off the servers instead of turning them to a STANDBY state. This way, we assume that each of the selected u_k processing nodes for a specific time slot consume maximum power, and the other $M - u_k$ nodes in a off state consume P_{off} power each. Then, if we consider the definition of latency as per Equation (3), and fix the time threshold to T^*, we can compute u_k as:

$$u_k = \frac{\bar{w}_k}{T^*}\bar{v}_k$$

where we assume that in each time slot the choice of u_k was able to process all the incoming queries, so that $x_k = 0$. The value of T^* is determined by the length of the time slot, as we want all the queries of a slot to be processed before proceeding to the next slot.

– **ShortTerm**: proposed by [Freire et al., 2014a], it implements the original power/latency trade-off model, without the modifications we proposed in this paper regarding the latency and power cost functions. This method does not completely turn the machines off neither considers energy rates in the power cost function and it implements a deterministic latency cost function.

5.1.2 Proposed methods

Next, we define the methods we implemented to solve the proposed research questions:

– **ShortTerm$**: based on the ShortTerm proposed by [Freire et al., 2014a], but with the modified power cost function: it completely switches off the servers when they are not necessary (instead of turning them to a STANDBY state) and it also implements the price-driven approach using hourly-variable energy rates.

– **ShortTerm$SBC**: this methods modifies the previous one by changing also the latency cost function definition. It implements the *Stationary Backlog-Carryover* approach, based on Queueing Theory. The aim is to determine if QT can be used to estimate the latency of a search engine, and compare its power with the deterministic approach implemented by ShortTerm$.

5.2 Evaluation Measures

As our work concerns balancing the trade-off between search engine efficiency and power consumption, we measure both aspects. In particular, we measure the mean response and waiting time for queries (denoted ACT and AWT respectively, and measured in milliseconds (ms)). We also report the 90th percentile regarding the ACT. Concurrently, we measure the power usage of the search engine (measured in KWh), as well as the maximum number of machines used at any slot of the day. We note that with some configurations of the search engine when there are insufficient replicas available, the search engine will become backlogged with excessive number of queued queries. To prevent any skew in the results, we drop queries that are not answered within 500 milliseconds, thereby returning an error page to the user of that query. Clearly this is an undesirable scenario, and hence, we count the number of unanswered queries (denoted %UQ). We also report the price of the consumed energy (in dollars).

To summarise, we consider as a success when the power consumption of the search engine can be reduced regarding the baselines, without marked negative impact upon the experience of the search engine users (indicated by the ACT/AWT and the percentage of unanswered queries).

5.3 Search Engine, Documents & Queries

To evaluate the proposed methods, we determine the processing times for real user queries submitted to a search engine platform. In particular, we index 50M Web documents from the TREC ClueWeb09 corpus (category B) using the Terrier IR platform [Ounis et al., 2006, Terrier-Team, 2015] – ClueWeb09 cat. B is intended to reflect the first tier of a commercial Web search engine index. While indexing the corpus, standard stopwords are removed and Porter stemming applied.

We extracted queries from the MSN 2006 query log [ACM, 2009]. As the frequency of queries in this data set is lower than the incoming traffic of current search engines, we proceeded as follows: we combine four days of the 2nd week of May to generate only one day (and we did the same for the 3rd week). Figure 1 presents the number of queries over the course of each day. This way we obtain two days of queries with realistic arrival times, and we can use the first day to predict the incoming traffic for the second one. During retrieval, we use the WAND dynamic pruning technique applying BM25 to rank 1000 documents for each query, recording the processing time of the query by a single replica. All efficiency experiments are made with a quad-core Intel Xeon 2.4GHz, with 8GB RAM, and where the inverted indexes are stored on a 160GB SATA drive. The multiple machines environment need for experimentation was simulated in Java.

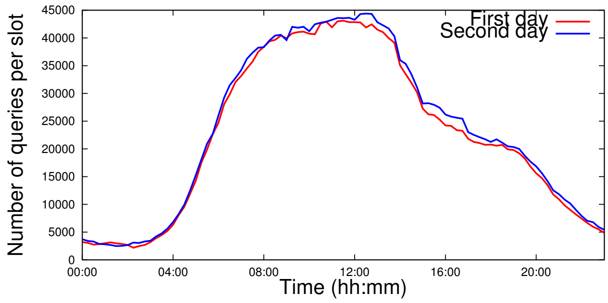


Figure 1: Number of queries arriving per 15 minute slot for both days.

5.4 Parameter Settings

To instantiate our model, we invoke various parameter settings as follows. Firstly, to calculate the power consumption of a replicated processing node, we use the energy ratings from [Edison, 2015] for a small server as follows: $P_{on} = 250W$, $P_{off} = 1.5W$. Peak Power is calculated by dividing the total peak power by the number of blades: $P_{peak} = 300.63W$. Standby power was reported to constitute the 5.6% of the energy consumed at ON state [Star, 2012]: $P_{standby} = 14W$. We assume a boot delay (from OFF to ON) of 20 sec. The energy prices were taken from hourly variable energy rates in Spain [Tarifaluzhora, 2015] in dollars. Within latency cost function (Equation (4)), we follow [Wang et al., 2010] and use $\alpha = -0.01$ for the ClueWeb09 cat. B corpus. For slot duration, we set $T_s = 15$ minutes, reflecting an interval that identifies general changing trends in query volumes that the model can quickly respond to, rather than random fluctuations that might be detected by shorter slot durations. The remaining parameters of our model, namely the power/latency trade-off λ and the number of replica query processors M are experimental variables that we vary within the next section.

6 Results

In this section, we aim to determine if our new power and latency cost functions allow the whole system to save power and money with latency comparable to that achieved by the baselines. To do so, we structure this section as follows: in Section 6.1 and 6.2, we select the values of the trade-off parameter λ and the maximum number of replicas M for the rest of the experimentation. Section 6.3 studies the effect of the new power cost function and Section 6.4 studies the viability of the Stationary Backload-Carryover approach in defining the latency function.

6.1 Effect of the trade-off parameter (λ)

The cost function we want to minimise depends on the value of $\lambda \in [0, 1)$ (see Eq. 1), which balances the power and latency of the system. In order to pick the most suitable value for this parameter, we run ShortTerm$ varying $\lambda \in 0.25, 0.5, 0.75$ and comparing the results regarding the two λ-independent baselines: Naïve and Threshold$.

The top part of Table 1 reports several evaluation measures achieved by ShortTerm$ and the two chosen baselines. The time slot length is set to $Ts = 15$ minutes and the maximum number of machines is $M = 15$.

Comparing the behaviour of ShortTerm$ regarding the different values of λ, it can be seen that $\lambda = 0.75$ adds a marked increase to the ART values (337 ms vs. 208 ms for $\lambda = 0.5$), as it promotes saving money in detriment of increasing the response times. The percentage of unanswered queries is also really high (exceeds 16%). Therefore, we definitely reject $\lambda = 0.75$.

With $\lambda = 0.25$ the model increases the price by 86% regarding $\lambda = 0.5$ (4.23$ versus 2.28$), by improving the latency in only by 0.07%.

If we consider the results with the two baselines, we can observe how Naïve achieves the best latency values (4% better than Threshold$ and 9% better than ShortTerm$), but at the cost of increasing the price (power) around

300%. This is because Naïve maintains all the machines ON during the whole day, while the other approaches try to adapt the number of necessary servers. Looking at the maximum number of machines used at some point of the day, we would like to note how SHORTTERM$ only turns on a maximum of 5 machines, while Threshold uses up to 11.

Thus, we decide to fix $\lambda = 0.5$ for future experiments as it achieves more than 100% of power savings by increasing the latency in 5% regarding the best of the two considered baselines here (Threshold$). With $\lambda = 0.5$ power cost and latency will be balanced equally.

Method	ACT (ms)	AWT (ms)	90thPC	% UQ	Max. Machines	Power (kWh)	Price (dollars)
Naïve	189	5	237	1.63	15	90	9.88
Threshold$	197	14	245	1.86	11	32	3.42
SHORTTERM$ ($\lambda = 0.25$)	195	11	243	1.80	10	40	4.23
SHORTTERM$ ($\lambda = 0.5$)	208	26	257	2.23	5	21.54	2.28
SHORTTERM$ ($\lambda = 0.75$)	337	180	368	16.69	2	10.41	1.11

Table 1: Effect of the trade-off parameter (λ).

6.2 Effect of the maximum number of machines (M)

The number of replicas is considered an important factor in the model. Table 2 reports the results obtained while maintaining the value of $\lambda = 0.5$ and varying $M = 10, 15, 20$. The time slot length is also set to $Ts = 15$ minutes.

Using 10 machines, the AWT of the queues is quite high (180 ms), so we discard $M = 10$. The performance with 15 and 20 machines is really close to the baselines (ART decreased in less than 1%), but with $M = 15$ the power is highly increased. Thus, we select $M = 15$ for the next experiments, as it allows to save energy, by achieving also good latency values (around 200 ms, accepted by commercial search engines).

Note that, SHORTTERM$ is always the one which uses the least number of machines. Although M establishes a maximum number, usually the model selects a much lower value (with $M = 15$ only 5 machines are switched on at the same time, while Naïve and Threshold$ use 15 and 11, respectively).

6.3 Behaviour of the proposed power cost function: considering energy rates and OFF state

Once we have established the optimal values for λ and M, we can compare the behaviour of SHORTTERM$, that includes the new power function, and SHORTTERM, the baseline proposed by [Freire et al., 2014a] that only considers the STANDBY state and it's not price-driven.

Method	ACT (ms)	AWT (ms)	90thPC	% UQ	Max. Machines	Power (kWh)	Price (dollars)
$M = 10$							
Naïve	193	9	241	1.75	10	60	6.57
Threshold$	198	15	246	1.88	10	31.86	3.34
ShortTerm$	337	180	368	16.69	2	10.41	1.11
$M = 15$							
Naïve	189	5	237	1.63	15	90	9.88
Threshold$	197	14	245	1.86	11	32	3.42
ShortTerm$	208	26	257	2.23	5	21.54	2.28
$M = 20$							
Naïve	187	2	235	1.57	20	60	6.57
Threshold$	197	14	245	1.87	11	32.9	3.45
ShortTerm$	195	11	243	1.80	10	40	4.23

Table 2: Effect of the maximum number of machines (M).

Table 3 shows the high power savings when switching the unnecessary machines off (21.54 kWh vs 97 kWh - 77% power savings). The main challenge when switching off unnecessary machines is to deal with the booting delay when reactivating the servers. However, although we have considered 20 sec of booting delay, the latency is only increased by 1.4%. For a better understanding regarding both techniques, see Figure 2 showing the difference in power consumption and latency achieved by these two approaches. Note how both power and latency functions vary according to the incoming query distribution showed in figure 1.

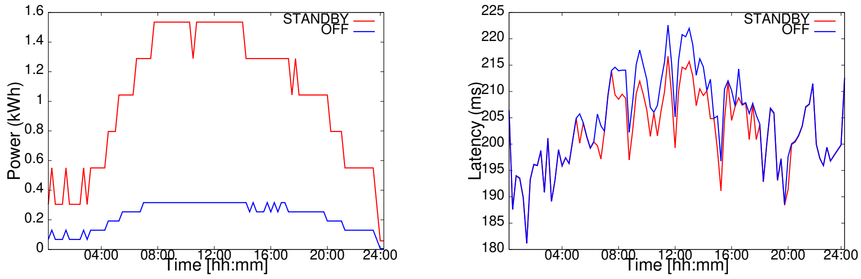
Therefore, we can answer the first research question (stated in Section 5) by saying that the new proposed power cost function allows saving up to 77% of power while increasing the latency by only 1.4%.

Method	ACT (ms)	AWT (ms)	90thPC	% UQ	Max. Machines	Power (kWh)	Price (dollars)
ShortTerm	205	22	253	2.11	6	97	10.35
ShortTerm$	208	26	257	2.23	5	21.54	2.28

Table 3: Effect of the power cost function: ShortTerm$ (OFF mode and price-driven) vs ShortTerm (STANDBY mode and no price-driven).

6.4 Behaviour of the new latency cost: Stationary Backlog-Carryover approach

In this section we experiment by modifying the latency cost function of the ShortTerm$. Instead of estimating the waiting time of queries using the deterministic approach described in Section 3.3.1, we use the Stationary Backlog-Carryover approach described in Section 4.2.



(a) Power (b) Latency

Figure 2: Latency and power values achieved by the new proposed SHORTTERM$ (OFF mode and price-driven) and the original SHORTTERM (STANDBY mode and no price-driven).

Table 4 reports the results obtaining using both approaches. First of all we would like to note that SHORTTERM$SBC is able to deal with the huge amount of incoming queries, while previous works [Freire et al., 2014b] attested that QT was not suitable for applying in large scale search engines for estimating the waiting time of queries. In [Freire et al., 2014b] they experimented that during the busiest periods of the day, the system achieved a non-stationary state where the maximum number of machines were activated. Table 4 shows that the maximum number of machines activated at the same time at any time slot is 13, while $M = 15$. So we can answer the second research question (see Section 5) by concluding that Queueing Theory models can suitable represent the latency of large-scale search engines.

Regarding the power savings achieved with SHORTTERM$SBC, we can say that the deterministic function still performs better, obtaining half power consumption and price. Latency values are comparable with the baselines, under 200 ms in average.

In conclusion, this section should encourage researchers to study the application of Queueing Theory in large-scale search engines, as we have proved that there is a chance in this scenario for this classical and well-tested models.

Method	ACT (ms)	AWT (ms)	90thPC	% UQ	Max. Machines	Power (kWh)	Price (dollars)
SHORTTERM$	208	26	257	2.23	5	21.54	2.28
SHORTTERM$SBC	193	9.33	241	1.73	13	44	4.7

Table 4: Effect of the latency cost function: deterministic approach (SHORTTERM$) vs Stationary Backlog-Carryover approach (SBC).

7 Conclusions

This paper has improved the current state-of-the art regarding *Green Information Retrieval*, by redefining previous power/latency trade-off models for large-scale search engines. We were able to manage the total disconnection of the unused severs with power savings up to 77% with a latency degradation of 1.4%. We have also considered hourly-variable energy rates.

Besides, although previous works have discarded the power of Queueing Theory in representing the latency of a large-scale search engine, we attested that stationary QT models can sucessfully be applied in this kind of scenarios, avoiding non-stationary states at periods of high contention.

As the setup parameters depend mostly on the number of machines and the power/latency trade-off parameter, and both of them are easy to set up, our future research will be focused in how to improve both the latency and power cost functions. More complex queueing theory models as well as the inclusion of renewable energy sources in the equation could be the key for achieving higher energy savings, while maintaining the latency and the sustainability.

References

[ACM, 2009] ACM (2009). Wscd '09: Proceedings of the 2009 workshop on web search click data. New York, NY, USA. ACM.

[Bunday, 1996] Bunday, D. (1996). *An Introduction to Queueing Theory.* Arnold.

[Cao, 2002] Cao, R. (2002). *Introducción a al Simulación y a la Teoría de Colas.* Carlos Iglesias, España, 1st edition.

[Chowdhury, 2012] Chowdhury, G. (2012). An agenda for green information retrieval research. *Inf. Process. Manage.*, 48(6):1067–1077.

[Cooper, 2003] Cooper, R. B. (2003). Queueing theory. In *Encyclopedia of Computer Science*, pages 1496–1498. John Wiley and Sons Ltd., Chichester, UK.

[Economou et al., 2006] Economou, D., Rivoire, S., and Kozyrakis, C. (2006). Full-system power analysis and modeling for server environments. In *Workshop on Modeling Benchmarking and Simulation (MOBS).*

[Edison, 2015] Edison, G. (2015). Blade server power study. http://goo.gl/kUY9wg.

[Freire et al., 2014a] Freire, A., Macdonald, C., Tonellotto, N., Ounis, I., and Cacheda, F. (2014a). A self-adapting latency/power tradeoff model for replicated search engines. In *Proceedings of WSDM 2014*, pages 13–22, New York, USA. ACM.

[Freire et al., 2014b] Freire, A., Macdonald, C., Tonellotto, N., Ounis, I., and Cacheda, F. (2014b). Towards green information retrieval: studying the suitability of queueing theory in reducing power consumption. In *CERI 2014: 3rd Spanish Conference on Information RetrievaL.*

[Gandhi and Harchol-Balter, 2011] Gandhi, A. and Harchol-Balter, M. (2011). How data center size impacts the effectiveness of dynamic power management. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 1164–1169.

[Google, 2015] Google (2015). Google green. https://www.google.com/green/.

[Gross and Harris, 1985] Gross, D. and Harris, C. M. (1985). *Fundamentals of Queueing Theory (2Nd Ed.).* John Wiley & Sons, Inc., New York, NY, USA.

[Hoelzle and Barroso, 2009] Hoelzle, U. and Barroso, L. A. (2009). *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers, 1st edition.

[Jeon and Prabhu, 2013] Jeon, H. and Prabhu, V. (2013). Modeling green fabs, a queueing theory approach for evaluating energy performance. In *Advances in Production Management Systems.*, volume 397, pages 41–48. Springer Berlin Heidelberg.

[Kayaaslan et al., 2011] Kayaaslan, E., Cambazoglu, B. B., Blanco, R., Junqueira, F. P., and Aykanat, C. (2011). Energy-price-driven query processing in multi-center web search engines. In *Proc. of SIGIR 2011*, pages 983–992, New York, NY, USA. ACM.

[Kendall, 1953] Kendall, D. G. (1953). Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chain. *Annals of Mathematical Statistics*, 24(3):338–354.

[Khargharia et al., 2008] Khargharia, B., Hariri, S., and Yousif, M. S. (2008). Autonomic power and performance management for computing systems. *Cluster Computing*, 11(2):167–181.

[Liu et al., 2009] Liu, J., Zhao, F., Liu, X., and He, W. (2009). Challenges towards elastic power management in internet data centers. *2012 32nd International Conference on Distributed Computing Systems Workshops*, 0:65–72.

[Mastroleon et al., 2005] Mastroleon, L., Bambos, N., Kozyrakis, C., and Economou, D. (2005). Automatic power management schemes for internet servers and data centers. In *GLOBECOM*, page 5. IEEE.

[Meisner et al., 2011] Meisner, D., Sadler, C. M., Barroso, L. A., Weber, W.-D., and Wenisch, T. F. (2011). Power management of online data-intensive services. In *Proc. of ISCA 2011*, pages 319–330, New York, NY, USA.

[Ounis et al., 2006] Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Lioma, C. (2006). Terrier: A high performance and scalable information retrieval platform. In *Proc. of the OSIR Workshop 2006*, pages 18–25, France. École Nationale Supérieure des mines de Saint-Etienne.

[Parolini et al., 2008] Parolini, L., Sinopoli, B., and Krogh, B. H. (2008). Reducing data center energy consumption via coordinated cooling and load management. In *Proc. of HotPower 2008*, pages 14–14, Berkeley, CA, USA.

[Radinsky et al., 2012] Radinsky, K., Svore, K., Dumais, S., Teevan, J., Bocharov, A., and Horvitz, E. (2012). Modeling and predicting behavioral dynamics on the web. In *Proc. of the 21st international conference on World Wide Web*, WWW '12, pages 599–608, New York, NY, USA. ACM.

[Sazoglu et al., 2013] Sazoglu, F. B., Cambazoglu, B. B., Ozcan, R., Altingovde, I. S., and Ulusoy, O. (2013). A financial cost metric for result caching. In *Proc. of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 873–876, New York, NY, USA. ACM.

[Silvestri, 2010] Silvestri, F. (2010). Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 4(1-2):1–174.

[Star, 2012] Star, E. E. (2012). Energy calculator. http://eu-energystar.org/.

[Stolletz, 2008] Stolletz, R. (2008). Approximation of the non-stationary m(t)/m(t)/c(t)-queue using stationary queueing models: The stationary backlog-carryover approach. *European Journal of Operational Research*, 190(2):478–493.

[Tarifaluzhora, 2015] Tarifaluzhora (2015). Tarifaluzhora. http://tarifaluzhora.es/.

[Terrier-Team, 2015] Terrier-Team (2015). Terrier. http://www.terrier.org.

[Wang et al., 2010] Wang, L., Lin, J., and Metzler, D. (2010). Learning to efficiently rank. In *Proc. of SIGIR 2010*, pages 138–145, New York, NY, USA. ACM.