Optimal Power Allocation in Server Farms

Anshul Gandhi
Carnegie Mellon University
Pittsburgh, PA, USA
anshulg@cs.cmu.edu

Mor Harchol-Balter*
Carnegie Mellon University
Pittsburgh, PA, USA
harchol@cs.cmu.edu

Rajarshi Das
IBM Research
Hawthorne, NY, USA
rajarshi@us.ibm.com

Charles Lefurgy
IBM Research
Austin, TX, USA
lefurgy@us.ibm.com

ABSTRACT

Server farms today consume more than 1.5% of the total electricity in the U.S. at a cost of nearly $4.5 billion. Given the rising cost of energy, many industries are now seeking solutions for how to best make use of their available power. An important question which arises in this context is *how to distribute available power among servers in a server farm so as to get maximum performance.*

By giving more power to a server, one can get higher server frequency (speed). Hence it is commonly believed that, for a given power budget, performance can be maximized by operating servers at their highest power levels. However, it is also conceivable that one might prefer to run servers at their lowest power levels, which allows more servers to be turned on for a given power budget. To fully understand the effect of power allocation on performance in a server farm with a fixed power budget, we introduce a queueing theoretic model, which allows us to predict the optimal power allocation in a variety of scenarios. Results are verified via extensive experiments on an IBM BladeCenter.

We find that the optimal power allocation varies for different scenarios. In particular, it is not always optimal to run servers at their maximum power levels. There are scenarios where it might be optimal to run servers at their lowest power levels or at some intermediate power levels. Our analysis shows that the optimal power allocation is non-obvious and depends on many factors such as the power-to-frequency relationship in the processors, the arrival rate of jobs, the maximum server frequency, the lowest attainable server frequency and the server farm configuration. Furthermore, our theoretical model allows us to explore more general settings than we can implement, including arbitrarily large server farms and different power-to-frequency curves. Importantly, we show that the optimal power allocation can significantly improve server farm performance, by a factor of typically 1.4 and as much as a factor of 5 in some cases.

*Research supported by NSF SMA/PDOS Grant CCR-0615262 and a 2009 IBM Faculty Award.

SIGMETRICS/Performance'09, June 15–19, 2009, Seattle, WA, USA.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Modeling techniques

General Terms

Theory, Experimentation, Measurement, Performance

1. INTRODUCTION

Servers today consume ten times more power than they did ten years ago [3, 21]. Recent articles estimate that a 300W high performance server requires more than $330 of energy cost per year [24]. Given the large number of servers in use today, the worldwide expenditure on enterprise power and cooling of these servers is estimated to be in excess of $30 billion [21].

Power consumption is particularly pronounced in CPU intensive *server farms* composed of tens to thousands of servers, all *sharing workload and power supply.* We consider server farms where each incoming job can be routed to any server, i.e., it has no affinity for a particular server.

Server farms usually have a fixed peak power budget. This is because large power consumers operating server farms are often billed by power suppliers, in part, based on their peak power requirements. The peak power budget of a server farm also determines its cooling and power delivery infrastructure costs. Hence, companies are interested in maximizing the performance at a server farm given a fixed peak power budget [4, 8, 9, 21].

The power allocation problem we consider is: *how to distribute available power among servers in a server farm so as to minimize mean response time.* Every server running a given workload has a minimum level of power consumption, b, needed to operate the processor at the lowest allowable frequency and a maximum level of power consumption, c, needed to operate the processor at the highest allowable frequency. By varying the power allocated to a server within the range of b to c Watts, one can proportionately vary the server frequency (See Fig. 2). Hence, one might expect that running servers at their highest power levels of c Watts, which we refer to as *PowMax*, is the optimal power allocation scheme to minimize response time. Since we are constrained by a power budget, there are only a limited number of servers that we can operate at the highest power level. The rest of

the servers remain turned off. Thus PowMax corresponds to having *few fast* servers. In sharp contrast is *PowMin*, which we define as operating servers at their lowest power levels of b Watts. Since we spend less power on each server, PowMin corresponds to having *many slow* servers. Of course there might be scenarios where we neither operate our servers at the highest power levels nor at the lowest power levels, but we operate them at some intermediate power levels. We refer to such power allocation schemes as *PowMed*.

Understanding power allocation in a server farm is intrinsically difficult for many reasons: First, there is no single allocation scheme which is optimal in all scenarios. For example, it is commonly believed that PowMax is the optimal power allocation scheme [1, 7]. However, as we show later, PowMin and PowMed can sometimes outperform PowMax by almost a factor of 1.5. Second, it turns out that the optimal power allocation depends on a very long list of external factors, such as the outside arrival rate, whether an open or closed workload configuration is used, the power-to-frequency relationship (how power translates to server frequency) inherent in the technology, the minimum power consumption of a server (b Watts), the maximum power that a server can use (c Watts), and many other factors. It is simply impossible to examine all these factors via experiments.

To fully understand the effect of power allocation on mean response time in a server farm with a fixed power budget, we introduce a queueing theoretic model, which allows us to predict the optimal power allocation in a variety of scenarios. We then verify our results via extensive experiments on an IBM BladeCenter.

Prior work in power management has been motivated by the idea of managing power at the global data center level [8, 20] rather than at the more localized single-server level. While power management in server farms often deals with various issues such as reducing cooling costs, minimizing idle power wastage and minimizing average power consumption, we are more interested in the problem of allocating peak power among servers in a server farm to maximize performance. Notable prior work dealing with peak power allocation in a server farm includes Raghavendra et al. [21], Femal et al. [10] and Chase et al. [5] among others. Raghavendra et al. [21] present a power management solution that coordinates different individual approaches to simultaneously minimize average power, peak power and idle power wastage. Femal et al. [10] allocate peak power so as to maximize throughput in a data center while simultaneously attempting to satisfy certain operating constraints such as load-balancing the available power among the servers. Chase et al. [5] present an auction-based architecture for improving the energy efficiency of data centers while achieving some quality-of-service specifications. We differ from the above work in that we specifically deal with minimizing mean response time for a given peak power budget and understanding all the factors that affect it.

Our contributions

As we have stated, the optimal power allocation scheme depends on many factors. Perhaps the most important of these is the specific relationship between the power allocated to a server and its frequency (speed), henceforth referred to as the *power-to-frequency relationship*. There are several mechanisms within processors that control the power-to-frequency relationship. These can be categorized into DFS (Dynamic Frequency Scaling), DVFS (Dynamic Voltage and Frequency Scaling) and DVFS+DFS. Section 2 discusses these mechanisms in more detail. The functional form of the power-to-frequency relationship for a server depends on many factors such as the workload used, maximum server power, maximum server frequency and the voltage and frequency scaling mechanism used (DFS, DVFS or DVFS+DFS). Unfortunately, the functional form of the server level power-to-frequency relationship is only recently beginning to be studied (See the 2008 papers [21, 25]) and is still not well understood. Our first contribution is the investigation of how power allocation affects server frequency in a single server using DFS, DVFS, and DVFS+DFS for various workloads. In particular, in Section 3 we derive a functional form for the power-to-frequency relationship based on our measured values (See Figs. 2(a) and (b)).

Our second contribution is the development of a queueing theoretic model which predicts the mean response time for a server farm as a function of many factors including the power-to-frequency relationship, arrival rate, peak power budget, etc. The queueing model also allows us to determine the optimal power allocation for every possible configuration of the above factors (See Section 4).

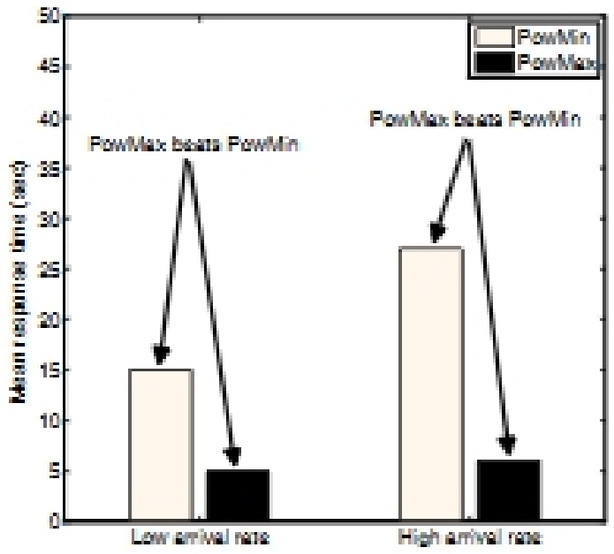
Our third contribution is the experimental implementation of our schemes, PowMax, PowMin, and PowMed, on an IBM BladeCenter, and the measurement of their response time for various workloads and voltage and frequency scaling mechanisms (See Sections 5 and 6). Importantly, our experiments show that using the optimal power allocation scheme can significantly reduce mean response time, sometimes by as much as a factor of 5. To be more concrete, we show a *subset* of our results in Fig. 1, which assumes a CPU bound workload in an open loop setting. Fig. 1(a) depicts one possible scenario using DFS where PowMax is optimal. By contrast, Fig. 1(b) depicts a scenario using DVFS where PowMin is optimal for high arrival rates. Lastly, Fig. 1(c) depicts a scenario using DVFS+DFS where PowMed is optimal for high arrival rates.

We experiment with different workloads. While Section 5 presents experimental results for a CPU bound workload, LINPACK, Section 6 repeats all the experiments under the STREAM memory bound workload, the WebBench web workload, and other workloads. In all cases, experimental results are in excellent agreement with our theoretical predictions. Section 7 summarizes our work. Finally, Section 8 discusses future applications of our model to more complex situations such as workloads with varying arrival rates, servers with idle (low power) states and power management at the subsystem level, such as the storage subsystem.

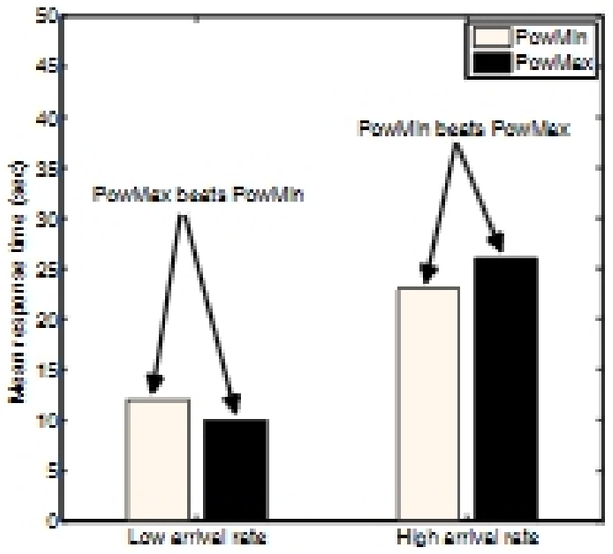
2. EXPERIMENTAL FRAMEWORK

2.1 Experimental setup

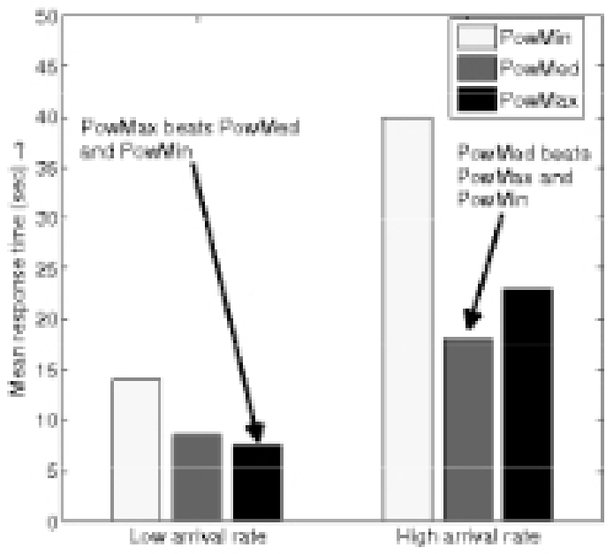
Our experimental setup consists of a server farm with up to fourteen IBM BladeCenter HS21 blade servers featuring two 3.0 GHz dual-core Intel Woodcrest Xeon processors and 1 GB memory per blade, all residing in a single chassis. We installed and configured Apache as an application server on each of the blade servers to process transactional requests. To generate HTTP requests for the Apache web servers, we employ an additional blade server on the same chassis as the workload generator to reduce the effects of network latency. The workload generator uses the web server performance benchmarking tool `httperf` [19] in the open server



(a) PowMax is best. at high arrival rates.



(b) PowMin is best at high arrival rates.



(c) PowMed is best at high arrival rates.

Figure 1: ***Subset of our results, showing that no single power allocation scheme is optimal. Fig.(a) depicts a scenario using DFS where PowMax is optimal. Fig.(b) depicts a scenario using DVFS where PowMin is optimal at high arrival rates whereas PowMax is optimal at low arrival rates. Fig.(c) depicts a scenario using DVFS+DFS where PowMed is optimal at high arrival rates whereas PowMax is optimal at low arrival rates.***

farm configuration and `wbox` [23] in the closed server farm configuration. We modified and extended `httperf` and `wbox` to allow for multiple servers and to specify the routing probability among the servers. We measure and allocate power to the servers using IBM's Amester software. Amester, along with additional scripts, collects all relevant data for our experiments.

2.2 Voltage and frequency scaling mechanisms

Processors today are commonly equipped with mechanisms to reduce power consumption at the expense of reduced server frequency. Common examples of these mechanisms are Intel's SpeedStep Technology and AMD's Cool 'n' Quiet Technology. The power-to-frequency relationship in such servers depends on the specific voltage and frequency scaling mechanism used. Most mechanisms fall under the following three categories:

Dynamic Frequency Scaling (DFS) a.k.a. Clock Throttling or T-states is a technique to manage power by running the processor at a less-than-maximum clock frequency. Under DFS, the Intel's 3.0 GHz Woodcrest Xeon processors we use allow for 8 operating points which correspond to effective frequencies of 12.5%, 25%, 37.5%, 50%, 62.5%, 75%, 87.5%, and 100% of the maximum server frequency.

Dynamic Voltage and Frequency Scaling (DVFS) a.k.a. P-states is a more efficient power-savings mechanism that reduces server frequency by reducing the processor voltage and frequency. Under DVFS, our processors allow for 4 operating points which correspond to effective frequencies of 66.6%, 77.7%, 88.9%, and 100% of the maximum server frequency.

DVFS+DFS attempts to leverage both DVFS, and DFS by applying DFS on the lowest performance state available in DVFS. Under DVFS+DFS, our processors allow for 11 operating points which correspond to effective frequencies of 8.3%, 16.5%, 25%, 33.3%, 41.6%, 50.0%, 58.3%, 66.6%, 77.7%, 88.9%, and 100% of the maximum server frequency.

2.3 Power consumption within a single server

When allocating power to a server, there is a minimum level of power consumption (b) needed to operate the processor at the lowest allowable frequency and a maximum level of power consumption (c) needed to operate the processor at the highest allowable frequency (Of course the specific values of b and c depend on the application that the server is running). Formally, we define the following notation:

Baseline power: b **Watts** The minimum power consumed by a fully-utilized server over the allowable range of processor frequency.

Speed at baseline: s_b **Hertz** The speed (or frequency) of a fully utilized server running at b Watts.

Maximum power: c **Watts** The maximum power consumed by a fully-utilized server over the allowable range of processor frequency.

Speed at maximum power: s_c **Hertz** The speed (or frequency) of a fully utilized server running at c Watts.

3. POWER-TO-FREQUENCY

An integral part of determining the optimal power allocation is to understand the power-to-frequency relationship. This relationship differs for DFS, DVFS, and DVFS+DFS, and also differs based on the workload in use. Unfortunately, the functional form of the power-to-frequency relationship is not well studied in the literature. The servers we use support all three voltage and frequency scaling mechanisms and therefore can be used to study the power-to-frequency relationships. In this section, we present our measurements depicted in Figs. 2(a) and (b) showing the functional relationship between power allocated to a server and its frequency for the LINPACK [13] workload. We generalize the functional form of the power-to-frequency relationship to other workloads in Section 6 (See Figs. 8 and 10). Throughout we assume a homogeneous server farm.

We use the tools developed in [17] to limit the maximum power allocated to each server. Limiting the maximum power allocated to a server is usually referred to as *capping* the power allocated to a server. We run LINPACK jobs