On Cost Modeling for Hosted Enterprise Applications

Hui Li and Daniel Scheibli

SAP Research, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany
hui.li@computer.org

Abstract. In enterprises nowadays typical business-critical processes rely on OLTP (online transaction processing) type of applications. Offering such applications as hosted solutions in Clouds rises many technical and non-technical challenges, among which TCO (Total Cost of Ownership) is one of the main considerations for most on-demand service/Cloud providers. In order to reduce TCO, a first step would be to analyze and study its cost components in depth. In this paper we adopt a quantitative approach and model two tangible cost factors, namely, server hardware and server power consumption. For server hardware, on one hand, a pricing model for CPU is proposed as a function of per-core performance and the number of cores, which also manifests the current multi-/many-core trend. Server power consumption, on the other hand, is modeled as a function of CPU utilization (as a main indication of system activity). By using published results from both vendor-specific and industry-standard benchmarks such as TPC-C, we show that a family of *Power functions* is successfully applied in deriving a wide range of cost models. Such analytic cost models, in turn, prove to be useful for the Cloud providers to specify the Service Level Agreements (SLAs) and optimize their service/infrastructure landscapes.

1 Introduction

Cloud computing represents the next wave of IT industry transformation by delivering services and computing as utilities over the Internet [1]. When the services and infrastructure are available in a pay-as-you-go manner to the general public, it is called a *Public Cloud*. The *Private Cloud*, on the other hand, refers to the internal services and resources of ITO (IT Organization) in a business which are not available to the public. Public cloud, such as Amazon Web Services, proves to be a sustaining business model for applications such as Web 2.0, testing and development, and certain data-intensive/HPC applications. ITOs can also outsource some of its non-critical processes from its Private Cloud to a Public one for elasticity and cost-saving considerations.

Despite the success of on-demand solutions for certain functionalities such as HR and CRM, business/mission critical applications remain largely to be deployed on-premise, especially for large organizations. For small and medium enterprises (SMEs), however, there is a market that the whole suite of business

D.R. Avresky et al. (Eds.): Cloudcomp 2009, LNICST 34, pp. 261–269, 2010.

applications be offered as hosted solutions. Apart from the challenges arise from security and multi-tenancy, TCO (Total Cost of Ownership) is one of the main considerations for any on-demand provider for such applications. This applies to both SaaS/Public Clouds for general offerings and Private Clouds that serve the LoBs (Line of Business).

For the Cloud providers to specify the Service Level Agreements (SLAs) and optimize their service/infrastructure landscapes [4], it is of crucial importance to analyze, understand, and model cost components within the TCO. This paper focuses on the cost modeling for hosted OLTP applications on both public and private Clouds. TCO is intrinsically complex and involves a great deal of tangible/intangible factors. Rather than providing a comprehensive TCO model, this paper focuses mainly on the quantitative aspects and models two tangible cost components, namely, server hardware and server power consumption. Firstly, a pricing model for CPU is proposed as a function of per-core performance and the number of cores. The per-core performance is based on the published results of industry-standard OLTP benchmark TPC-C [11] on Intel DP/MP platforms. The fitted CPU pricing model also manifests the current multi-/many-core trend. Secondly, server power consumption is modeled as a function of CPU utilization using a customized Power function. By combining the fitted models for both CPU costs and power consumption, we have developed a simplified analytic model for hosted OLTP applications that incorporates hardware and operation costs.

The rest of the paper is organized as follows: Section 2 develops a CPU cost model based on the certified results of TPC-C benchmarks on Intel DP/MP platforms. Section 3 conducts customized performance tests and models the server power consumption in relationship to the CPU utilization as the main indicator for system activity. Section 4 presents the combined cost model for OLTP applications in a hosted environment, and discusses its context and applicability. Conclusions and future work are presented in Section 5.

2 Modeling CPU Costs for OLTP Applications on Multi-core Platforms

Among the many components of server hardware, namely CPU, memory, storage, and network, we focus on the CPU costs in this paper and make simplified assumptions that costs of other components remain constants or scale with the CPU costs. We are particularly interested in the price-performance relationship on multi-/many-core platforms, as the general trend in processor development has been from single-, multi-, to many cores. Our goal is to investigate and model the relationship between the objective, namely the price per-CPU (C_{cpu}) or price per-core (C_{core}), and the two related parameters: number of cores (N_{core}) and benchmark results per-core (T_{core}). T_{core} also corresponds to the processing speed of the core, and thus the resource demands of the measured OLTP applications. If we model the application system as a closed multi-station queuing center, T_{core} is theoretically bounded by $1/D$, where D is the resource demand

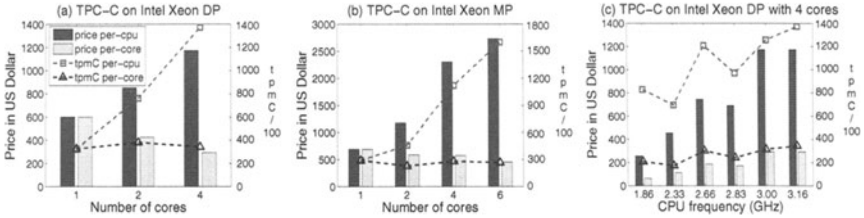

Fig. 1. 117 certified TPC-C benchmark results run on Intel Xeon DP/MP platforms within the timeframe between 7/2002 and 12/2008. TPC-C is measured in transactions per minute (tpmC). Such a throughput measure is defined as how many New-Order transactions per minute a system generates while executing other transactions types.

(minimum response time) of the application on the server. This gives a general idea on the relationship between the performance model and the cost model, whose objectives are conflicting with each other. In this section we focus on modeling the CPU costs P given the number of cores and benchmark results per-core for OLTP applications.

We examine the certified TPC-C [11] benchmark results on Intel DP/MP platforms and associate them with CPU price information [7], which are shown in Figure 1[1]. As there are two independent parameters (N_{core} and T_{core}) involved, we study one of them by fixing the value of the other, and vice versa.

Firstly let us look at the price versus the number of cores given a similar per-core performance. In 1(a), we can see that the per-core price decreases as the number of cores per CPU increases on the Intel Xeon DP platform. As the per-core performance of TPC-C remains the same, the price/performance ratio improves by adding more cores. Generally this trend also applies to TPC-C on Intel MP as shown in Figure 1(b). We notice that the per-core tpmC decreases slightly as the number of cores increases. This is because that the core frequency scales down as the number of cores scales up, which is shown in Table 1. Nevertheless, as the chip design becomes better and more efficient, the per-core performance/frequency ratio (r) improves along the evolution of generations. From a customer perspective this does not mean that the response time of a single application can improve as the resource demand decreases only by increasing the core speed. The main benefit is on the much improved throughput numbers per CPU price.

Secondly let us examine the price versus the per-core performance given the same number of cores. In Figure 1(c), as predicted, we can see that the price increases as the CPU frequency and throughput numbers increase. Some abnormal behavior happens between 2.33 GHz and 2.83 GHz. This may be explained

[1] Disclaimer: The performance in tpmC is influenced by additional factors like machine architecture, cache sizes, memory size/latency/bandwidth, operating system, storage system characteristics, DBMS, TPC-C version/settings as well as other factors not mentioned here. Vendor-specific benchmarks [9] and certified results [10] are also studied and the results are not published here.

Table 1. CPU frequency and the performance/frequency ratio: $r = T_{core}/GHz$

Benchmark	1-core	2-core	4-core	6-core
tpcc/DP (GHz)	3.4	3.0	3.16	-
tpcc/DP (r)	9.5	12.7	10.9	-
tpcc/MP (GHz)	3.33	3.0	2.93	2.66
tpcc/MP (r)	8.7	7.6	9.6	10.0

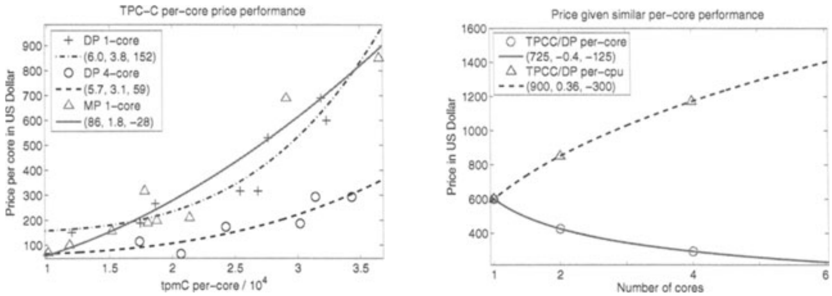

Fig. 2. Fitted power function parameters are (c_1, c_2, c_3) as appeared in Equation 1

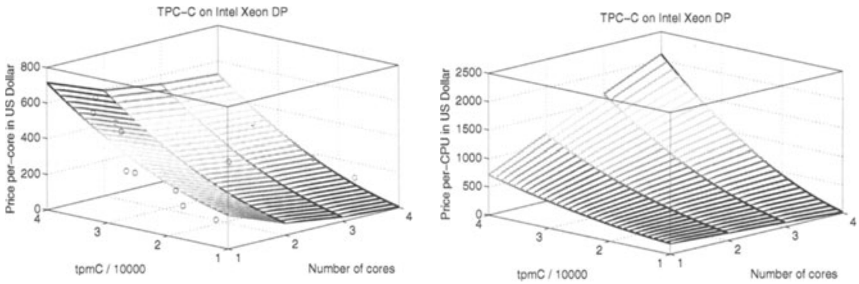

Fig. 3. The fitted cost models for price per-core (C_{core}) and price per-CPU (C_{cpu})

partially by the noise in the data as there is only one available measurement each for CPU frequency at 2.33 GHz and 2.83 GHz. Nevertheless, the general trend of price increasing with speed (core frequency) still holds. Figure 2 gives a better view on the pattern of how price changes with the per-core performance for TPC-C. On both DP and MP platforms with different cores, the per-core price scales with the per-core throughput like a power function. We studied different functions for curve fitting, including polynomial, exponential, power, and other custom functions. It is found that the power function, shown in Equation 1, gives the overall best fit for different data sets.

$$f(x) = c_1 x^{c_2} + c_3 \tag{1}$$

Table 2. CPU cost model parameters for TPC-C on Intel Xeon DP (Equation 3)

model param.	c_1	c_2	c_3	c_4	c_5
TPCC/DP	36	2.0	261	-0.9	-105

Figure 2 also shows that the price per-core decreases like a power function while increasing the number of cores per-CPU. This indicates that the power function (Equation 1) can be used to model the relationships between price per-core and throughput performance/number of cores individually.

The next step is to study per-core performance (T_{core}) and number of cores (N_{core}) jointly and model their relationship with price. Since the power function is the best fitted model for T_{core} and N_{core} individually, we can extend this model to a multi-variable case[2]. A power function with two variables can be formulated as follows:

$$C_{core} = g(T_{core}, N_{core}) = c_1 T_{core}^{c_2} + c_3 N_{core}^{c_4} + c_5, \tag{2}$$

where $(c_1, ..., c_5)$ are the parameters to be fitted. The price per-CPU C_{cpu} is readily obtained by multiplying price per-core with the number of cores:

$$C_{cpu} = N_{core} C_{core} = N_{core} g(T_{core}, N_{core}). \tag{3}$$

Figure 3 shows the fitting of TPC-C/DP data with the cost models C_{core} and C_{cpu}. A non-linear least-squares method in the Matlab Optimization toolbox is used for curve fitting, and the fitted parameters are shown in Table 2. We can see that the fitted model gives an overall good interpolation of real benchmark results. The trend/relationship between price and the two factors, namely performance per-core and number of cores, is well captured. Although different benchmarks on different platforms may yield different parameters[3], the model shown in Equation 3 is general and flexible enough for estimating a wide range of CPU cost information.

It should be noted that the power-function based model for CPU costs developed in this section depends on the Intel pricing schemes for its multi-/many-core platforms. Our contribution is to fit such price information with mathematical models, in relationship to real OLTP benchmark results. This gives the planners/architects at the provider side a convenient tool for estimating hardware costs given the desired performance level of their applications.

[2] An informal proof for this extension can be described as follows: When x or y is constant, either $f(x)$ or $f(y)$ takes the form $ax^b + c$. This means there is no x or y components of any form in the function other than x^b or y^d. So $f(x, y)$ can be written as $ax^b + cy^d + e$.

[3] There are no sufficient data for curve fitting of TPC-C benchmark on Intel MP platform.

Table 3. Power consumption model parameters for a customized OLTP application (Equation 5)

model param.	c_1	c_2	c_3	c_4	c_5
OLTP app.	276.7	15	7	2.1	1.1

3 Modeling Power Consumption

Power consumption and associated costs become increasingly significant in modern datacenter environments [6]. In this section we analyze and model the server power consumption of OLTP applications. We study the relationship between system power consumption (P_{sys}, measured in Watts) and CPU utilization (U), which is used as the main metric for system-level activity. Our experimental methodology and tooling are largely similar to the ones in [5,6], except that we focus on OLTP-like workloads. We run a customized OLTP application similar to sales and distribution business processes, on a 64-bit Linux server with 1 Intel dual-core CPU and 4 GB main memory. The system power is measured using a power meter connected between the server power plug and the wall socket. The CPU utilization data is collected using Linux utilities such as *sar* and *iostat*. Monitoring scripts in SAP performance tools are also used for correlating power and CPU utilization data.

Before data fitting and modeling we first perform a data pre-processing step called *normalization*. Instead of directly modeling P_{sys} we use a normalized power unit P_{norm}, which is defined as follows:

$$P_{norm} = \frac{P_{sys} - P_{idle}}{P_{busy} - P_{idle}}, \tag{4}$$

where the measured P_{idle} ($U = 0$) and P_{busy} ($U = 1$) for our test system are 42W and 84W, respectively. The normalized measurement results are shown in Figure 4.

Generally speaking the server power consumption increases as the CPU utilization grows. One particular important finding from the measurement data is the so-called power capping behavior [6], which means there are few times that the highest power is consumed by the server. Additionally we find that such highest power points are drawn mostly when the CPU utilization is higher than 80% and they have very similar peak values. Most of the common functions, such as quadratic polynomial, power, exponential, and Gaussian, cannot fit such flat curve of power values in the high-utilization interval (see the quadratic fitting in Figure 4).

We developed a model that can fit such power-capping behavior well. The model is inspired by the frequency response curve of a linear filter called Butterworth filter. It has such desired "flat" behavior in the passband of the frequency. We replace the polynomial part of the transfer function with the following customized power function which has two U components:

$$h(U) = c_1 U^{c_2} + c_3 U^{c_4} + c_5, \tag{5}$$

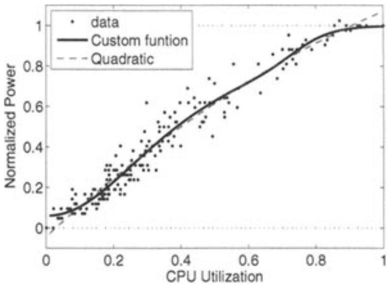

Fig. 4. Normalized system power relates to CPU utilization. The custom function is shown in Equation 6.

where $(c_1, ..., c_5)$ are the parameters to be fitted. The model that relates normalized power (P_{norm}) and CPU utilization U can be formulated as follows:

$$P_{norm}(U) = 1 - h(U)^{-1}. \tag{6}$$

The fitting result is shown in Figure 4 and the fitted model parameters are listed in Table 3. We can see that the proposed power model fits the measurement data well, especially during the high utilization period. Given the measurements for P_{idle} and P_{busy}, the overall system power consumption P_{sys} can be obtained by substituting P_{norm} (Equation 6) in Equation 4.

4 A Cost Model for Enterprise Applications

By combining the cost models for CPU and power consumption in previous sections (equations (3), (4), and (6)), we developed a cost model for business applications:

$$\begin{aligned} &Cost(T_{core}, N_{core}, U, I) = \\ &p_0 + p_1 C_{cpu} + p_2 \int_{t \in I} P_{sys}(U(t))dt, \end{aligned} \tag{7}$$

where t is the measurement time, I is the measurement period ($t \in I$), p_0 is an adjusting constant, p_1, and p_2 are the weighting parameters that scale the individual model outputs. If during the measurement period only average utilization is available, the output can be written as $P_{sys}(\overline{U})I$. The model in (7) uses an additive form to combine server hardware costs and operational costs, in which parameters p_1 and p_2 have to be set properly to reflect different cost structures.

To summarize from a mathematical modeling perspective, we can conclude that the power function $(c_1 x^{c_2} + c_3)$ and its variants have attractive properties for fitting a wide range of curves, including both single- and multi-variable case. Thus, the power function family represents a general and flexible modeling library from which different cost models can be fitted and derived.

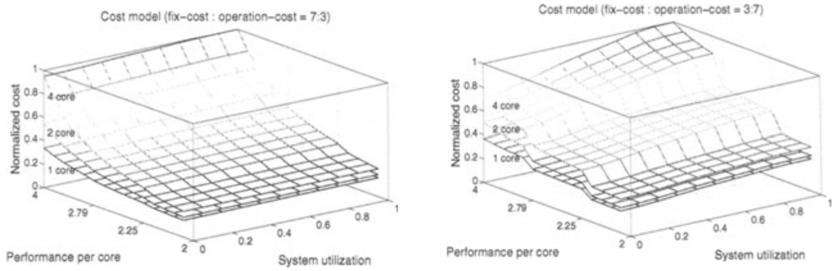

Fig. 5. Cost model structures: For a typical "classical" data center, the ratio of fixed cost versus operational cost (r) is set to 7 : 3. For a modern commodity-based data center, the ratio r is set to 3 : 7

In practice when using the cost model for the optimization of enterprise systems, we need to determine the weighting parameters p_1 (fixed cost) and p_2 (operational cost). These parameters are chosen in a way to reflect the real numbers obtained in case studies in [3]. There are two situations under study in this paper. On one hand, for a typical "classical" data center the ratio of fixed cost versus operational cost (r) is set to 7 : 3, which indicates that the high server capital costs dominate overall TCO by 70%. For a modern commodity-based data center, on the other hand, the ratio r is set to 3 : 7. This means operational costs including power consumption and cooling become the dominating factor. The cost model outputs of (7) for these two situations are illustrated in Figure 5, where differences can be clearly identified. For instance, the total cost increases significantly with the increasing system utilization for the high operational cost situation ($r = 3 : 7$), which is not the case for the high fixed cost counterpart($r = 7 : 3$). We also observe that the discontinuity of cost model outputs along the performance/core axis in the $r = 3 : 7$ situation. This is because the settings of P_{idle} and P_{busy} take discrete values like a piecewise constant function. The CPU performance per core is divided into three ranges and the values of P_{idle} and P_{busy} are set accordingly. For instance, for a 2-core system from low to high performance, P_{idle} and P_{busy} have been set to $[40, 60, 80]$ and $[65, 95, 150]$, respectively. Such settings are made in accordance to the CPU power consumption characteristics on Intel platforms. In the $r = 7 : 3$ situation, however, such effects is dramatically reduced as the operational cost is no longer dominant. In our ongoing research we investigate both situations in the optimization phase to see how different cost structures impact the SLA-driven planning on the service provider side.

5 Conclusions and Future Work

In this paper we developed a analytic cost model that consists of two tangible cost components: server hardware and power consumption. The CPU price is modeled as a function of number of cores and per-core throughput performance for OLTP applications. The server power consumption is modeled as a function of CPU utilization. Both models include Power function or its variants as components, which indicates that Power function as a mathematical form is suitable to fit a wide range of cost structures.

Cost modeling is one important enabling component in our ongoing work on SLA-driven planning and optimization of hosted business applications [8]. Service-Level Agreements (SLA) are bidding contracts between service consumer and service provider on guarantee terms such as performance and cost. In our view well-specified SLAs are important, even indispensable components for making utility-driven SOA and Cloud computing a success. SLAs can also be applied between layers and IT stacks in a provider's landscape. For enabling SLA-aware planning and optimization studies on the provider side, practical models are needed to encapsulate performance information, cost information, and other factors. The proposed cost model is utilized in our studies in optimizing a system landscape running OLTP applications by taking multiple conflicting objectives into account.

References

1. Above the clouds: A berkeley view of cloud computing. Tech. Rep. UCB/EECS-2009-28, University of California, Berkeley (2009)
2. Barroso, L.: The price of performance: An economic case for chip multiprocessing. ACM Queue 3(7), 48–53 (2005)
3. Barroso, L.A., Hölzle, U.: The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. Morgan & Claypool, San Francisco (2009)
4. Chase, J.S., Anderson, D.C., Thakar, P.N., Vahdat, A., Doyle, R.P.: Managing energy and server resources in hosting centres. In: Proc. of SOSP, pp. 103–116. ACM, New York (2001)
5. Economou, D., Rivoire, S., Kozyrakis, C., Ranganathan, P.: Full-system power analysis and modeling for server environments. In: Proc. of Workshop on Modeling, Benchmarking and Simulation, MOBS (2006)
6. Fan, X., Weber, W.-D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: Proc. of the 34th Intl. Sym. on Computer Architecture (ISCA 2007), pp. 13–23. ACM Press, New York (2007)
7. Intel. Intel processor pricing, 2007-2009, `http://www.intc.com/priceList.cfm` (accessed March 2009)
8. Li, H., Theilmann, W., Happe, J.: SLA Translation in Multi-layered Service Oriented Architectures: Status and Challenges. Tech. Rep. 2009-08, University of Karlsruhe, Germany (2009)
9. Marquard, U., Goetz, C.: SAP Standard Application Benchmarks - IT Benchmarks with a Business Focus. In: Kounev, S., Gorton, I., Sachs, K. (eds.) SIPEW 2008. LNCS, vol. 5119, pp. 4–8. Springer, Heidelberg (2008)
10. SAP. The Sales and Distribution (SD) Benchmark, Two-tier Internet Configuration (2009), `http://www.sap.com/solutions/benchmark/sd.epx` (accessed March 2009)
11. TPC. TPC-C: on-line transaction processing benchmark V5 (2009), `http://www.tpc.org/tpcc/` (accessed March 2009)