Mechanistic and Empirical Models for Processing Capacity Prediction

André A. Cesta, Geraldo M. Silva

Eldorado Research Institute
Brasilia - Campinas - Porto Alegre, Brazil

{andre.cesta,Geraldo.magela}@eldorado.org.br

Abstract. *Former works have already demonstrated how processing capacity models, such as Amdahl's Law and Gunther's Universal Scalability Law, can be generalized and extended from functions of the number of processors 'p' to functions of intrinsic microprocessor variables, such as: clock speed 'c' and hardware threads 't', thus being able to predict performance. These previous articles focused on adjusting the processor capacity models to a medium dataset, obtaining prediction errors of around 12%. The present paper focuses on adjusting these models and empirical modeling techniques to lower errors from 13% to 7% for an ERP system performance benchmark, and from 17% to 10% for the SPECInt 2006 benchmark.*

Resumo. *Trabalhos anteriores demostraram como modelos de capacidade de processamento, tais como a Lei de Amdahl e a Lei de Escalabilidade Universal de Gunther, podem ser generalizados e estendidos de funções de número de processadores 'p' para funções de variáveis intrínsecas do microprocessador, tais como: velocidade do clock 'c' e threads do hardware 't', possibilitando a predição do desempenho. Esses artigos focaram-se no ajuste de modelos de capacidade de processadores para um conjunto de dados mediano, obtendo erros de predição em torno de 12%. O presente artigo concentra-se no ajuste desses modelos e técnicas de modelagem empírica para reduzir os erros de 13% para 7% para um benchmark de desempenho de um sistema ERP, e de 17% para 10% para o benchmark do SPECInt 2006 rate.*

1. Introduction

In this paper, we take one of the models researched in [Cesta et al. 2012] and fit it to new challenging datasets while specifying the considerations that had to be made and methodologies that had to be applied. The new results obtained contribute to the amount of scientific evidence supporting these models and to the body of knowledge from performance engineering and performance modeling.

The main equation used in this paper is Eq. 25 from [Cesta et al. 2012], here referred to as Eq. 1:

$$S(t,es,c) = \frac{c \cdot t \cdot es}{c \cdot (1 + \sigma \cdot (t-1)) + (es - c) \cdot (t \cdot i + \pi_i \cdot (1-t))} \quad (1)$$

Equation 1 is a result of multiple generalizations from Amdahl's Law and Gunther's Universal Scalability Law. The steps for its elaboration were:

1. Initially, Amdahl's Law was adjusted for an environment with different thread count and clock speeds. For this generalization it was considered that a thread is equivalent to a serial processor, and a clock speed variation will influence the speed-up, since the clock speed is usually different in the processors. These considerations can be performed by adding '*t*', for thread, instead of '*p*', for processor, in Amdahl's Law; and by adding 'c', for clock speed, assuming it will speed up both serial and parallel program parts. Eq. 2 shows speed-up by Amdahl's Law (S_A) on the left and the generalization for threads and clock speed on the right.

$$S_A = \frac{p}{1+\sigma\cdot(p-1)} \Rightarrow S_A(t,c) = c.\frac{t}{1+\sigma\cdot(t-1)} \quad (2)$$

2. Finally, using the similar speed-up analysis demonstration techniques, but assuming that program tasks are not only serial and parallel, but instead: 1. serial and internal or 2. serial and bound by external resource speed; and 3. parallel and internally executed or 4. parallel but restricted by external resource speed; and considering *t*, *c*, *es* (external resource speed, which if unknown can be set to 1) will influence each program part differently, we arrive at Eq. 1.

The Eq. 1 extends and generalizes Amdahl's Law to cover heterogeneous processor cases. *S(t, es, c)* stands for the speedup from threads, external resource access speed and clock speed, respectively. The speedup can be equated to scalability and throughput [Gunther 2010]. The equation model adjustment parameters are: the average percentage of processor time spent in serial tasks from the program being modeled, 'σ'; the average percentage of processor time spent in internal tasks that can benefit from clock speed increases since they do not rely on external resources' access speeds, '*i*'; and the average percentage of processor time spent in parallelizable and internal tasks that benefit from both clock speed and thread count increases, π_i .

2. Methodology

A. Datasets

Dataset 1: a subset of all machines benchmarked for a well-known ERP system benchmark. Our subset comprised all 143 CISC machines with l2 caches on the core and l3 caches on the processor, for the period from January, 2006 to December, 2012. Five machines were discarded during residual analysis.

Dataset 2: all 3478 CISC machines benchmarked for SPECInt rate 2006 [Henning 2006] using DDR3 memory with l2 caches on the core and l3 caches on the processor or not present. There were only two machines as influential observations during residual analysis.

B. Models

In this paper we fitted predictive models to datasets ranging from highly heterogeneous to highly homogeneous ones. Our models were: mechanistic such as Eq. 1 and computationally evolved equations in ANN models.

Notice that in order to fit an ANN with more than five variables to a dataset of only 143 machines, we had to resort to best practices for low-data situations.: 1. k-fold resampling techniques with an optimized k value; 2. hidden layer size optimization; and

3. network training and epoch choice guided by validation dataset-fitting error. The other features from our networks were the most commonly used ones: hyperbolic tangent function, back-propagation training, etc.

In order to attain more model generalization power in highly heterogeneous dataset cases, Eq. 1 was also complemented with additional empirical regression terms. These terms were the variables isolation such as *l1* cache amount and processor generation

C. Equations Format

In this paper, whenever mentioning fitting Eq. 1 or any other, except for ANNs, we mean fitting an MLR regression model of the following algebraic form: $\hat{y} = \beta_1 \cdot Eq1 + 0$. Notice how β_0 is forced to zero as Eq. 1 has to pass through the origin (zero threads and clock will yield zero performance as in Amdahl, where zero processors will yield zero performance). Notice that before fitting this model through least squares, one has to assume certain values for Eq. 1 parameters σ, i, π_i (we used a brute force, combinatorial search to find the best values).

3. Results

Below, we present a table with results from fitting various equations to various datasets. The "Regression model" column lists regression equation. The "Dataset" column shows which dataset was fitted by the model. It can be either the dataset from Methods Section (1 or 2) or a more homogeneous subset of datasets 1 and 2 (indicated by labels such as 1.1, 1.2) and accompanied by an explanation on how to generate it. The "Cardinality" column lists how many machines were present in the initial dataset. The "% error" column represents the mean value for the absolute value from the regression residual divided by the actual observed or measured machine capacity. This is a better indicator of regression quality than the standard error. The R^2_{adj} column indicates the model's R-squared adjusted value or the determination coefficient.

Table 1. Some combined variables used in result table

Symbol	Description
l1_p_t	The ratio of l1 cache available per thread.
l2_p_t	The ratio of l2 cache available per thread.
l3_p_t	The ratio of l3 cache available per thread.

Table 2. All numerical results by fitting Eq. 1 to dataset 1, dataset 2 and subsets of them

Regression Eq. (c.f.Table 5)	Dataset summarized description (all datasets are derived from datasets 1 or 2)	Dataset Cardinality	%error	R^2adj	σ	*i*	π_i
Eq.1	1 (c.f. Methodology for description)	138	13.1%	0.9742	0.000631	0.526315	0.526315
Eq.1	1.1 Only Intel E5-Generation, coincidentally all have the same *l1_p_t*, *l2_p_t*, but there are two values of *l3_p_t* and memory channels.	16	5.8%	0.9192	0.000510	1.000000	1.000000
ANN	1	138	5.6%	-	-	-	-
Eq.1	2 (c.f. Methodology for description)	3476	17.5%	0.9692	0.001518	0.600000	0.600000
Eq.1	2.1 Only Intel E5- with *l1_p_t* =32k; *l2_p_t* = 128k; *l3_p_t* = 320k or 640k	439	2.2%	0.9988	0.001132	0.945833	0.945833
Eq.1	2.2 Only Intel E7-, with *l1_p_t* =32k; *l2_p_t* = 128k; 300k < *l3_p_t* < 384k	71	2.6%	0.9992	3.00E-04	1	1
Eq.1	2.3 Xeon E series, *l1_p_t* =32k; *l2_p_t* = 128k; 512k < *l3_p_t* < 768k.	549	3.3%	0.9983	0.002539 1	0.972916	0.972916
Eq.1	2.4 Xeon X series, *l1_p_t* =32k; *l2_p_t* = 128k; 512k < *l3_p_t* < 1024k.	814	4.1%	0.9975	0.004055 6	0.978571	0.978571

4. Discussion

In order to validate our results further, we fitted an artificial neural network (ANN) to dataset 1. This ANN, which was the best ANN we could have created, achieved test dataset percent error of 5.6%, a value only slightly higher than the one obtained by our regression on Table 2 row 2. By best ANN we could have created, we mean the network optimized for scarce data situations.

Notice that the Eq.1 regression percent error is 13% for generics processor models. Error rate much higher than presented by ANN. Our results, obviously, show high accuracy in predicting performance of Eq. 1 when applied to a datase more

homogeneous. It is noticed that for different generations of processors (*Intel E5*, *Intel E7*, *Intel Xeon E series* and *X series*) the accuracy achieved an error rate of 2% to 4%.

While being used by a sizing team to reduce experimental costs, this work is applicable currently for our team hardware sizing. We use specifics coefficients, coupled with the ANN, for their respective generations of processors. We achieved a satisfactory error rate, as the results show.

5. Related and Future Works

In [Cesta et al. 2011], [Cesta et al. 2012] and Eyerman's article [Eyerman et al. 2011] provide a good review of the literature around processor performance prediction and around mechanistic, empirical and hybrid models. However, the model we propose has the differential of being top-down and based on Amdahl's Law. Specifically, Eyerman's models are bottom-up and based on variables that we cannot obtain from our benchmark datasets, operating at a much lower level. These include variables such as cache misses and TLB misses, that can only be obtained by running micro-benchmarks.

As future work, it would be interesting to combine both approaches (Eyerman's and the one introduced in this work) in order to find an even more powerful model.

6. Conclusion

This article confirmed our assumption from Section "Methodogy/B. Models" that extraneous variables not captured by Eq. 1, such as: the number of memory access channels or small design changes in microarchitecture were causing the increased errors when fitting Eq. 1 to highly heterogeneous datasets containing multiple machine generations. Such confirmation came mostly via the reduced errors verified when using Eq. 1 to model more homogeneous sub-datasets with variability forcedly concentrated on threads and clock speed.

References

Cesta, A., Takara, A. and Moscheto, D. (2011) "Leveraging diverse regression approaches and heterogeneous machine data in the modeling of computer systems performance", in Proceedings of MSV, Las Vegas, Nevada, USA, pp. 201–207.

Cesta, A., Silva, G. M. and Storch, M. (2012) "Performance Prediction for Processors and External Resources", in Proceedings of CMG, Las Vegas, Nevada, USA.

Eyerman, S., Hoste, K. and Eeckhout, L. (2011) "Mechanistic-Empirical Processor Performance Modeling for Constructing CPI Stacks on Real Hardware", Performance Analysis of Systems and Software (ISPASS), IEEE International Symposium on , vol., no., pp.216,226, 10-12.

Gunther, N. (2010) "Guerrilla Capacity Planning: A Tactical Approach to Planning for Highly Scalable Applications and Services", Springer.

Henning, J. L. (2006) "Spec cpu2006 benchmark descriptions", SIGARCH Comput. Archit. News, vol. 34, pp. 1-17, September.