Univariate Interpolation-based Modeling of Power and Performance

Jóakim von Kistowski
University of Würzburg
joakim.kistowski@
uni-wuerzburg.de

Samuel Kounev
University of Würzburg
samuel.kounev@
uni-wuerzburg.de

ABSTRACT

Performance and power scale non-linearly with device utilization, making characterization and prediction of energy efficiency at a given load level a challenging issue. A common approach to address this problem is the creation of power or performance state tables for a pre-measured subset of all possible system states. Approaches to determine performance and power for a state not included in the measured subset use simple interpolation, such as nearest neighbor interpolation, or define state switching rules. This leads to a loss in accuracy, as unmeasured system states are not considered. In this paper, we compare different interpolation functions and automatically configure and select functions for a given domain or measurement set. We evaluate our approach by comparing interpolation of measurement data subsets against power and performance measurements on a commodity server. We show that for non-extrapolating models interpolation is significantly more accurate than regression, with our automatically configured interpolation function improving modeling accuracy up to 43.6%.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of Systems—*Performance attributes*

General Terms

Interpolation, Energy Efficiency

Keywords

Power, Performance, Interpolation, Utilization, SERT

1. INTRODUCTION

Computational devices have to run at a great range of device utilization levels, with power and performance scaling non-linearly over the different load levels, as power saving mechanisms, such as dynamic voltage and frequency scaling (DVFS) are being used. A common and highly accurate method for the characterization of power and performance of a workload on a target system at a given load level is the measurement of said characteristics and storing those measurements in a table for re-use during management decisions. The stored measurement results contain power and performance values for a subset of possible system states. Determining the power and performance of the system at other states remains challenging. Currently, several approaches to estimate power consumption at these states exists: Some tools, such as [10] only consider the existing pre-defined states and determine the current state either by nearest neighbor interpolation or through other rule-based mechanisms. Another approach is the training of models based on measured data. Models range from simple models, such as the linear power model [1] and variations thereof [3] to more complex regression models, which also take additional system properties into account [6]. In comparison to approximation, interpolation increases prediction accuracy, as it does not sacrifice or approximate any of the pre-measured results. A great number of different interpolation methods exist. Depending on the system and the power or performance metric under observation, a different interpolation method may be optimal. In addition, some interpolation methods can be configured with varying degrees of freedom.

This paper presents a library[1] for automated selection of interpolation and configuration strategies for performance and power characterization with the goal of minimizing prediction errors for unmeasured performance and power. The major contributions of this paper are as follows:
(1) We present an approach for automated selection and configuration of an interpolation strategy for a given set of performance or power measurements; *(2)* We propose a composition of piece wise polynomial interpolators of varying degrees for the interpolation of a system's power over utilization function; *(3)* We demonstrate that for closed and bounded inputs, interpolation provides superior prediction accuracy on the system of measurement in comparison to approximation techniques, such as regression.

We evaluate our approach based on power and performance measurements using ten of the workloads of the SPEC SERT [5], measuring at 100 load levels per workload. Prediction accuracy is evaluated based on the methods' ability to predict power and performance for all load levels based on a smaller sub-set. We show that for bounded problem spaces interpolation features superior accuracy compared to regression. Our automated interpolation configuration and selection improves modeling accuracy by 43.607% if additional reference data is available and by 31.36% if it is not.

[1]Library: `http://descartes.tools/interpolation`

2. INTERPOLATION

Scattered data interpolation [4] is the reconstruction of a continuous function $f(x)$ from n different sample points $\{(x_1, f_1), (x_2, f_2), ..., (x_n, f_n)\}$. In this paper, we consider univariate functions, where $f(x)$ is our power or performance metric and the input metric x the corresponding system metric (usually utilization). We consider the following interpolation functions:

- **Nearest Neighbor Interpolation**: $f(x) = f(x_i)$ with $x_i \in \{x_1, ..., x_n\}$ being the nearest neighbor to x, meaning that $\forall x_j \in \{x_1, ..., x_n\} : |x - x_i| \leq |x - x_j|$.
- **Linear Interpolation**: Given the two nearest neighbors of x, x_i and x_{i+1}, with $x_i \leq x$ and $x_{i+1} > x$: $f(x) = f(x_i) + (f(x_{i+1}) - f(i)) \frac{x - x_i}{x_{i+1} - x_i}$
- **Shepard Interpolation** [9]: $f(x) = f(x_i)$ if $x = x_i$, otherwise: $f(x) = \frac{\sum_{i=1}^{n} w_i(x) f(x_i)}{\sum_{i=1}^{n} w_i(x)}$ with $w_i(x) = \frac{1}{|x - x_i|^p}$. Parameter p is freely configurable and usually selected based on experience.
- **Polynomial Interpolation**: $f(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_0 x^0$, with the coefficients a_i being the solution to a system of equations that guarantees that the polynomial of degree n passes through all $n + 1$ data points. To avoid oscillation (Runge's Phenomenon [8]), we also split the set into subsets of size m and interpolate these using polynomial functions of degree $m - 1$.
- **Spline Interpolation**: A type of piece-wise polynomial interpolation. Guarantees that the overall function remains continuous in all interpolated data points [2].

3. DETERMINING ACCURACY

During the automated selection and configuration process, we determine the accuracy of an interpolation function using one of these two methods: Interpolation against a reference dataset or cross validation. The latter of these two options is the more common one, as interpolation accuracy improves with additional interpolated data. As a result, all available data is included in the interpolated data, leaving no additional data for referencing. However, a separate reference dataset is most useful when determining the optimal interpolation method for a given problem domain.

If a reference dataset R containing the tuples (x_i, y_i) is available, we calculate a set of absolute errors E with $E = \{e_1, ..., e_n\}$ for interpolation function f as in Eq. 1:

$$\forall (x_i, y_i) \in R : e_i = |f(x_i) - y_i| \tag{1}$$

If no reference dataset exists, we calculate the set of absolute errors E via cross-validation on the interpolated dataset I containing the tuples (x_i, y_i), with $|I| = n$. We create a set of cross-validation-sets V_i as displayed in Eq. 2:

$$\forall i \in \{2, ..., n-1\} : V_i = I \setminus \{(x_i, y_i)\} \tag{2}$$

With f_i being the interpolation function constructed using V_i, we calculate the cross-validation errors as in Eq. 3:

$$\forall i \in \{2, ..., n-1\} : e_{i-1} = |f_i(x_i) - y_i| \tag{3}$$

Our implementation allows the metric for calculation of the final aggregate error of the error set E to be passed using a functional expression. In this paper, we use the arithmetic mean and median.

4. SELECTION AND CONFIGURATION

We allow selection of the best interpolation function for the problem's domain using an independent reference dataset containing a larger set of data points, which describes a similar problem as the dataset to be interpolated. E.g., both datasets describe power per load level measurements, yet they were measured for different workloads on different machines. In such a case, we create a subset from the reference dataset. This subset is of the same size as the set that is to be interpolated and contains data points with the closest possible input values (x-axis values) to the input values of the target data set. Then we select the best configuration and interpolation methods for this subset by comparing the aggregate modeling error of the potential interpolation methods. The function with the minimum aggregate error is selected as the final interpolation function. For functions with a configurable parameter (degree of freedom), this parameter must be auto-configured first. Finally, we transfer the selected method and configuration to the actual set to be interpolated.

In some cases, reference data is not available and selection of a single pre-configured interpolation method is not possible, either due to a lack of sufficient domain knowledge or because of the problem domain's nature. In this case, we select the best interpolation function for a given dataset by calculating the cross-validation error. We compute different cross-validation datasets, each with one data point removed. At least one data point must be removed for cross-validation, as the self-prediction error of an interpolation function is always 0. Consequently, cross-validation using the full dataset is not possible. We evaluate the interpolation method's ability to predict the missing data point for each of the cross-validation datasets. The function with the minimum aggregate error over all cross-validation datasets is selected as the final interpolation function.

Among the existing interpolation functions used in this paper, two function types feature a configurable degree of freedom. We select the final configuration parameter using a hill-climbing approach, as expected values are well known and as the parameters in all of our cases have a specified minimum value. To apply hill-climbing, each parameter must have an initial parameter instance p_0 and a function h so that $p_i = h(p_{i-1})$. With $f(p_i)$ being the parametrized interpolation function and $e(f(p_i))$ being its error metric, we iterate over the parameters p_i in an ascending order (using h) until $e(f(p_{i+1})) \leq e(f(p_i))$.

To improve interpolation accuracy for performance and power measurements, we introduce a new approach to parametrization of piece-wise polynomials. It is designed to minimize the interpolation error due to state changes caused by device power management. These state changes cause non-continuous behavior in a power or performance function. Consequently, it pays to introduce breaks at these points when interpolating polynomials. Breakpoints are detected at the data points featuring the greatest difference between their range value and their successor's range. Meaning that given a set of n breakpoint indices B, with $|B| = n$, the following has to hold true: $\forall i \in B : |y_{i+1} - y_i| < |y_{j+1} - y_j|$, with $j \notin B$. We use these break points for piece-wise polynomial interpolation by interpolating polynomials over the subsets defined within breakpoint boundaries. The amount of breakpoints remains a freely configurable parameter and can be determined using our hill-climbing approach.

Model	10% intervals	20% intervals	25% intervals	scattered
Cubic Spline	0.172%	0.222%	**0.224%**	**0.19%**
Max. degree Polynomial	0.35%	0.249%	0.304%	0.215%
Linear (= Poly., degree 1)	0.169%	0.297%	0.296%	0.175%
Nearest Neighbor	0.704%	1.419%	1.772%	1.117%
Piece-Wise Polynomial (degree 2)	**0.168%**	0.225%	0.255%	0.191%
Piece-Wise Polynomial (degree 3)	0.19%	0.243%	0.317%	0.26%
Piece-Wise Polynomial (degree 4)	0.204%	0.246%	-	0.218%
Shepard (weight 2)	0.353%	0.651%	0.793%	0.58%
Split Polynomial (1 break)	0.35%	0.249%	0.304%	0.215%
Split Polynomial (2 breaks)	0.305%	0.246%	0.317%	0.197%
Split Polynomial (3 breaks)	0.294%	**0.217%**	0.241%	0.191%
Split Polynomial (4 breaks)	0.284%	0.233%	0.296%	0.26%
Linear Power Model	2.757%	2.757%	2.757%	2.757%
Exponentially Corrected Model	4.023%	3.96%	3.938%	4.042%
Linear Regression	0.344%	0.255%	0.317%	0.199%

Table 1: Mean modeling errors for the power over load level function of the SSJ workload.

5. EVALUATION

We evaluate the accuracy of our interpolation methods based on measurements using the SPEC Server Energy-Efficiency Rating Tool (SERT) [5]. We use all of SERT's mini-workloads (called worklets) except for the memory *Capacity* worklet, as it doesn't scale with load levels, and the *XMLvalidate* worklet, which didn't scale correctly for fine-granular target load levels. The used worklets are: six different CPU worklets [11]), two storage worklets, the memory *Flood* worklet, and the hybrid *SSJ* worklet. We exercise each of the worklets at 100 different load levels, with a separate idle power measurement serving as a 101st measurement. For each of these levels, throughput (in s^{-1}) and power consumption (in W) are measured. We select both evenly distributed as well as scattered subsets of measurements from the original 101 results. All models are evaluated based on their ability to accurately reconstruct the entire original measurement for the given workload using no additional data. We compute the relative absolute error ($|p_{model} - p_{referece}|/p_{reference}$) for each data point in the reference measurement, using the mean as overall error metric. A smaller relative error corresponds to a more accurate model. We compare the accuracy of our interpolation functions with three common power modeling approaches:

- **Linear Power Model:**
  The linear power model is a common model in literature [7]. It calculates power consumption at a target load level $u \in [0, 1]$: $p(u) = p_{idle} + (p_{max} - p_{idle})u$
- **Linear Power Model (Exponential Correction):**
  This power model, introduced in [3], modifies the linear power model using an exponential correction factor r, accounting for the curvature in power per utilization functions: $p(u) = p_{idle} + (p_{max} - p_{idle})(2u - u^r)$
- **Polynomial Fitting using Regression:**
  We create polynomials of varying degrees to fit the measured results. The coefficients $a_0, a_1, ..., a_n$ of the polynomial function $p(u) = a_n x^n + ... a_1 x + a_0$ are fitted using OLS multiple linear regression.

5.1 Comparison of Interpolation Methods

A comparison of the mean accuracy of the interpolation and modeling methods for the power per load level function of the hybrid SSJ workload is shown in Table 1. The displayed error metric is the mean of the relative absolute differences between each data point in the reference measurement and the corresponding model prediction. The table shows the error for interpolation methods and the power models. Interpolation with multiple configuration options (such as Shepard weights) is marked with the respective configuration. The best interpolation method changes depending on input dataset size. Compared to the simple power models, interpolation functions are highly accurate, with modeling errors reliably less than 1% for all dataset sizes. The two exceptions are nearest neighbor and Shepard interpolation. Polynomial interpolation provides the greatest accuracy, the optimal configuration depends on the size of the input dataset.

Regression is not as accurate as any of the piece-wise polynomial interpolation methods (equi-distant splits, dynamic splits, or splines) in cases of equi-distant data. For our scattered dataset, regression is only slightly less accurate than cubic spline interpolation.

It is also notable that the dynamically split polynomial interpolation and interpolation using one polynomial of maximum degree feature identical accuracy for this workload. This effect is the result of the largest difference in power consumption for SSJ taking place right before full utilization. Consequently, the first breakpoint is set at full utilization, resulting in no change to polynomial interpolation without breakpoints. However, this observation is specific to SSJ and does not repeat for other workloads. The CPU-bound LU workload's power, e.g., scales differently with increased load, as can be seen in its power scaling behavior. LU has a sharper increase in power consumption beginning at 60% load. This results in a visible impact on interpolation accuracy (see Table 2), as interpolation methods must correctly recognize this sudden increase in power draw.

5.2 Interpolation using Reference Dataset

Next, we evaluate the efficiency of automatic selection of a good interpolation function using an independent reference dataset. We choose the SSJ measurement set as our reference dataset and then choose the corresponding interpolation function for any given subset based on the optimal interpolation function of SSJ for this given subset.

The accuracy of interpolation based on automated interpolation selection and configuration using SSJ as the reference dataset is displayed in Table 3. Compared to using regression, interpolation using an independent reference

Model	10% intervals	20% intervals	25% intervals	scattered
Cubic Spline	0.198%	0.48%	0.901%	0.256%
Linear	0.219%	**0.241%**	0.856%	**0.2%**
Piece-Wise Polynomial (degree 2)	0.226%	0.442%	0.922%	0.268%
Piece-Wise Polynomial (degree 3)	0.239%	0.412%	**0.744%**	0.331%
Split Polynomial (1 break)	0.156%	0.53%	**0.744%**	0.304%
Split Polynomial (4 breaks)	**0.127%**	**0.241%**	0.856%	0.257%
Linear Regression	0.526%	1.305%	1.774%	0.823%

Table 2: Mean modeling errors for the power over load level function of the LU workload.

Worklet	10% intervals Pol. (deg. 2)	20% intervals Split Pol. (3 br.)	25% intervals Cubic Spline	scattered Linear
LU	0.226%	0.46%	0.901%	0.2%
SHA256	0.197%	0.232%	0.181%	0.174%
Flood	0.898%	2.017%	2.529%	0.787%
Sequential	0.123%	0.255%	0.178%	0.109%

Table 3: Mean modeling errors of independent reference based interpolation for representative workloads.

dataset features an improvement of 43.607% (mean over the relative improvements) in accuracy. Compared to always choosing the single, most commonly best interpolation method (dynamically split polynomial interpolation with one breakpoint), automated selection and configuration still improves model accuracy by 20.025% (mean over the relative improvements).

5.3 Interpolation using Cross-Validation

Although cross-validation based configuration and selection lacks the additional data of its reference based counterpart, it is still fairly accurate. Specifically it is 31.36% more accurate than linear regression.

The major drawback of the cross-validation based configuration and selection are the different scales between the cross-validation problem and the final dataset to be interpolated. E.g., when removing a data point from the 10% utilization interval set, the interpolation function has to interpolate a 20% utilization gap. Its ability to do so is then judged to indicate its accuracy when interpolating equidistant data-points at 10% intervals.

6. CONCLUSIONS

This paper introduces an approach to automated configuration and selection for interpolation of power and performance measurements. We show that for closed and bounded problems, interpolation is far more accurate than similar black-box modeling methods, such as regression. Compared to linear regression, we are able to improve accuracy of a power per utilization prediction by 43.607% if additional reference data is available and by 31.36% if it is not. Our automated approach can also improve prediction accuracy by up to 20.025%, compared to always selecting the usually best interpolation function.

The results in this paper enable more accurate predictions of power and performance based on pre-measured results. This, in turn, allows for better decision making in power and performance management. The interpolation approaches, introduced in this paper, can also be used for the generation of additional data for under-fitted models.

For future work, we plan to evaluate the accuracy of the approach in this paper for other problem domains than power and performance modeling, as it is generic enough that is should be possible to use in any space that can be modeled as an interpolation problem.

7. REFERENCES

[1] P. Bohrer, E. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, and R. Rajamony. The Case for Power Management in Web Servers. In *Power Aware Computing*, pages 261–289. Springer US, 2002.

[2] R. Burden and J. Faires. *Numerical Analysis*. PWS-Kent, 1989.

[3] X. Fan, W.-D. Weber, and L. A. Barroso. Power Provisioning for a Warehouse-sized Computer. In *The 34th ACM International Symposium on Computer Architecture*, 2007.

[4] J. Hoschek and D. Lasser. *Fundamentals of Computer Aided Geometric Design*. A. K. Peters, Ltd., Natick, MA, USA, 1993.

[5] K.-D. Lange and M. G. Tricker. The Design and Development of the Server Efficiency Rating Tool (SERT). In *Proceedings of the 2nd ACM/SPEC International Conference on Performance Engineering*, ICPE '11, pages 145–150, New York, NY, USA, 2011. ACM.

[6] A. Lewis, S. Ghosh, and N.-F. Tzeng. Run-time Energy Consumption Estimation Based on Workload in Server Systems. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, HotPower'08, Berkeley, CA, USA, 2008. USENIX Association.

[7] S. Rivoire, P. Ranganathan, and C. Kozyrakis. A Comparison of High-level Full-system Power Models. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, HotPower'08, pages 3–3, Berkeley, CA, USA, 2008. USENIX Association.

[8] X. Shen, F. Mohd-Zaid, and R. Francis. Runge Phenomenon: A virtual artifact in image processing, 2012.

[9] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference*, ACM '68, pages 517–524, New York, NY, USA, 1968. ACM.

[10] A. Verma, P. Ahuja, and A. Neogi. pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems. In *Middleware 2008*, pages 243–264. Springer Berlin Heidelberg, 2008.

[11] J. von Kistowski, H. Block, J. Beckett, K.-D. Lange, J. A. Arnold, and S. Kounev. Analysis of the Influences on Server Power Consumption and Energy Efficiency for CPU-Intensive Workloads. In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE 2015)*, ICPE '15, New York, NY, USA, February 2015. ACM.