Application Speedup Characterization: Modeling Parallelization Overhead and Variations of Problem Size and Number of Cores.

Victor H. F. Oliveira
Univ. Federal do Rio Grande do Norte
Dep. Eng. Computação e Automação
vhfdoliveira@ufrn.edu.br

Alex F. A. Furtunato
Inst. Federal do Rio Grande do Norte
Dir. Acad. Tecnologia da Informação
alex.furtunato@ifrn.edu.br

Luiz F. Silveira
Univ. Federal do Rio Grande do Norte
Dep. Eng. Computação e Automação
lfelipe@dca.ufrn.br

Kyriakos Georgiou
University of Bristol
Department of Computer Science
Kyriakos.Georgiou@bristol.ac.uk

Kerstin Eder
University of Bristol
Department of Computer Science
Kerstin.Eder@bristol.ac.uk

Samuel Xavier-de-Souza
Univ. Federal do Rio Grande do Norte
Dep. Eng. Computação e Automação
samuel@dca.ufrn.br

ABSTRACT

To make efficient use of multi-core processors, it is important to understand the performance behavior of parallel applications. Modeling this can enable the use of online approaches to optimize throughput or energy, or even guarantee a minimum QoS. Accurate models would avoid probe different runtime configurations, which causes overhead. Throughout the years, many speedup models were proposed. Most of them based on Amdahl's or Gustafson's laws. However, many of those make considerations such as a fixed parallel fraction, or a parallel fraction that varies linearly with problem size, and inexistent parallelization overhead. Although such models aid in the theoretical understanding , these considerations do not hold in real environments, which makes the modeling unsuitable for accurate characterization of parallel applications. The model proposed estimates the speedup taking into account the variation of its parallel fraction according to problem size, number of cores used and overhead . Using four applications from the PARSEC benchmark suite, the proposed model was able to estimate speedups more accurately than other models in recent literature.

CCS CONCEPTS

• **Computer systems organization → Multi-core architectures**;
• **Theory of computation → Parallel computing models**;

KEYWORDS

Parallel Computing; Application Characterization; Performance Modeling

ACM Reference Format:
Victor H. F. Oliveira, Alex F. A. Furtunato, Luiz F. Silveira, Kyriakos Georgiou, Kerstin Eder, and Samuel Xavier-de-Souza. 2018. Application Speedup Characterization: Modeling Parallelization Overhead and Variations of Problem Size and Number of Cores.. In *ICPE '18: ACM/SPEC International Conference on Performance Engineering Companion , April 9–13, 2018, Berlin, Germany.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3185768.3185770

ICPE '18, April 9–13, 2018, Berlin, Germany

ACM ISBN 978-1-4503-5629-9/18/04…$15.00
https://doi.org/10.1145/3185768.3185770

1 THE PROPOSED SPEEDUP MODEL

In this work, we propose a parallel speedup model that combines the effects of the variation of the problem size and the number of processors on the parallel fraction of the code—as similarly addressed in [7], and the effect of parallelization overhead—as approached by the authors of [11] and [5]. Important to note that none of these authors considered both effects in their models, therefore, this is a novelty proposed in our work.

The simplicity of our model allows for a straightforward comparison to previous work in this area. We take Amdahl's speedup as a starting point and add a quota for the parallelization overhead in the parallel execution time and make the parallel fraction vary with the number of processors and problem size, which leads us to

$$S = \frac{1}{(1 - f(p,N)) + \frac{f(p,N)}{p} + Q(p,N)}, \tag{1}$$

where S is the speedup, Q is a quota for the parallelization overhead, and the function $f(p,N)$ denotes the parallel fraction that varies with the problem size N, and the number of processors p, as

$$f(p,N) = max(min(f_1 + \frac{f_2}{p} + f_3 \times f_4^N, 1), 0), \tag{2}$$

where f_1, f_2, f_3 and f_4 are empirical parameters used to characterize the application. This function establishes that f has one fixed part represented by f_1, independent of the other execution parameters; a component that is inversely proportional to p, represented by f_2, which can be explained by the fact that as p changes, the number of instructions that can run in parallel also changes, since p can become even larger than the parallelization capability; and another component that varies exponentially with N, represented by f_3 and f_4, which can be explained by the fact that the parallel and serial parts can vary nonlinearly between each other, as N varies.

For Q, we propose the following function:

$$Q(p,N) = q_1 + \frac{q_2 \times p}{q_3^N}, \tag{3}$$

where q_1, q_2 and q_3 are empirical parameters used to characterize the application. This function establishes that the overhead tends to grow linearly with the increase of p, since more interprocess communication is expected, and tends to decrease exponentially with the increase of N, since less context switch is expected for larger problems.

Table 1: Results - Mean Square Error values and Coefficient of Determination (R^2)

Applications	Model proposed on (1)		Model based on [1]		Model based on [5]		Model based on [7]	
	MSE	R^2	MSE	R^2	MSE	R^2	MSE	R^2
Blackscholes	0.1066%	**0.9823**	**0.1056%**	**0.9823**	0.1062%	0.9822	0.1111%	0.9814
Freqmine	**3.446%**	**0.9898**	260.77%	0.2251	260.58%	0.2256	5.078%	0.9849
Fluidanimate	**0.2604%**	**0.9992**	6.089%	0.9808	1.754%	0.9944	0.395%	0.9987
Vips	**23.48%**	**0.9346**	57.02%	0.8411	43.14%	0.8798	29.51%	0.9178

2 RESULTS

The results showed in this Section were extracted using a shared-memory compute node of the High Performance Computing Center at UFRN (NPAD/UFRN) with 2 CPUs Intel Xeon Sixteen-Core E5-2698v3 2.3 GHz, 40M cache and 128 GB RAM DDR4 2133 RDIMM.

To estimate the parameters of (2) and (3), as all parameters in [1], [7] and [5] used for comparison, we used a Particle Swarm Optimization (PSO) algorithm to minimize the Mean Square Error (MSE) of the model compared to measurements of a training set, composed by the configurations where $p = [2, 4, 8, 16, 32]$ and $N = [1, 2, 4, 5, 7, 8, 10]$ for all the applications.

The variation of N was done by modifying the linear component of each application, as defined in [2], with the exception of freqmine, which has no linear component. In this case, we vary linearly the number of transactions, considering 99.000 transactions for $N = 1$ and 990.000 transactions for $N = 10$. To do those tasks we develop a library in python, which can be accessed in [3].

Once the parameters were estimated, we compared the results of the models with the measurement of a test set, composed by the configurations where $p = [3, 6, 12, 24]$ and $N = [3, 6, 9]$ for blackscholes, freqmine and vips, and $p = [2, 4, 8, 16, 32]$ and $N = [3, 6, 9]$ for fluidanimate, because this application works only with p as a power of two.

The summary of the results is presented in Table 1. The table contains the MSE values, shown as a percentage of the mean speedups of all the measurements of each application, and the coefficient of determination (R^2) for each model and application. It is possible to notice that our model obtained the smaller MSE, except for blackscholes, which all models were able to make very good predictions, because their behavior is almost independent of N and p, with f close to 1. Also, the values of the R^2 of our model to all the applications is very close to 1, higher than all other models, which indicates that almost all the variation of the real measurements can be explained by our model [6]. Fig. 1 shows that our model also achieved the best maximum relative errors for all applications, with the exception of blackscholes, compared to the others models.

3 CONCLUSIONS

The proposed model was able to make good estimations of the speedup, which makes it a suitable candidate for further studying the effects of other execution parameters. The next step is to apply our model to a larger number of applications. The PARSEC has 8 more applications where is possible to manipulate their input sizes. Similarly, for the Splash-2 [9] benchmarks. A possible extension of this model should include the effect of varying the processor's clock frequency. For example, if the frequency decreases, memory

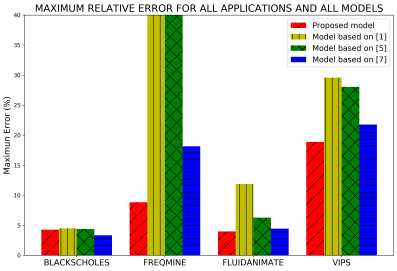

Figure 1: Maximum relative error of all applications and models using the test set.

bounded applications can start to perceive more memory requests attended per processor cycle. This may have a significant positive effect on the speedup. Also, energy-aware approaches to reduce software energy consumption have often the hardware clock frequency as an adjusting parameter, which makes a frequency-aware speedup model very desirable for such techniques [4, 8, 10].

REFERENCES

[1] Gene M Amdahl. 1967. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*. ACM, 483–485.
[2] Christian Bienia. 2011. Benchmarking Modern Multiprocessors. (2011), 153.
[3] Alex Furtunato. 2017. parsecpy. (2017). https://gitlab.com/lappsufrn/parsecpy
[4] Kyriakos Georgiou, Samuel Xavier-de Souza, and Kerstin Eder. 2017. The IoT energy challenge: A software perspective. *IEEE Embedded Systems Letters* (2017).
[5] Siegfried Höfinger and Ernst Haunschmid. 2017. Modelling parallel overhead from simple run-time records. *The Journal of Supercomputing* (2017), 1–17.
[6] Douglas C Montgomery. 2017. *Design and analysis of experiments*. John Wiley & Sons.
[7] Surya Narayanan, Bharath N Swamy, and André Seznec. 2014. Impact of serial scaling of multi-threaded programs in many-core era. In *Computer Architecture and High Performance Computing Workshop (SBAC-PADW), 2014 International Symposium on*. IEEE, 36–41.
[8] Daniele De Sensi, Massimo Torquati, and Marco Danelutto. 2016. A reconfiguration algorithm for power-aware parallel applications. *ACM Transactions on Architecture and Code Optimization (TACO)* 13, 4 (2016), 43.
[9] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta. 1995. The SPLASH-2 programs: characterization and methodological considerations. *Isca '95* June (1995), 24–36. DOI: http://dx.doi.org/10.1109/ISCA.1995.524546
[10] Samuel Xavier-De-Souza, Carlos A. Barros, Marcio O. Jales, and Luiz F.Q. Silveira. 2015. Not faster nor slower tasks, but less energy hungry and parallel: Simulation results. *2015 4th Berkeley Symposium on Energy Efficient Electronic Systems, E3S 2015 - Proceedings* (2015), 3–5. DOI: http://dx.doi.org/10.1109/E3S.2015.7336814
[11] Leonid Yavits, Amir Morad, and Ran Ginosar. 2014. The effect of communication and synchronization on Amdahl's law in multicore systems. *Parallel Comput.* 40, 1 (2014), 1–16.