Quantifying Productivity–Towards Development Effort Estimation in HPC

Sandra Wienke*, Tim Cramer*, Matthias S. Müller* and Martin Schulz†
*IT Center, RWTH Aachen University, Aachen, Germany
Email: {wienke, cramer, mueller}@itc.rwth-aachen.de
†CASC, Lawrence Livermore National Laboratory, CA, USA
Email: schulzm@llnl.gov

***Abstract*—With increasing expenses for future HPC centers, the need to look at their productivity, defined as amount of science per total cost of ownership (TCO), grows. This includes development costs which arise from development effort spent to parallelize, tune or port an application to a certain architecture. Development effort estimation is popular in software engineering, but cannot be applied directly to (non-commercial) HPC setups due to their particular target of *performance*. In our work-in-progress, we illustrate a methodology to qualify and quantify development effort parameters and hence how to estimate development effort and productivity. Here, the main challenge is to account for the numerous impact factors on development effort. We show preliminary results for two case studies: Questionnaires reveal development effort parameters with high impact and statistical tests help us to derive further details (here: comparing programming models). Additionally, we provide an online survey to engage the HPC community.**

I. Introduction

Towards exascale computing, the expenses of HPC centers increase in terms of acquisition, energy, administration, and programming. Thus, a quantifiable benefit-cost metric is needed to make an informed decision on how to invest available budgets, e.g., for the procurement of the next supercomputer. We define this productivity metric [1] as follows:

$$\text{productivity} = \frac{\text{value}}{\text{cost}} = \frac{\text{\# app.runs}}{\text{total cost of ownership (TCO)}} \quad (1)$$

TCO includes (amongst others) the acquisition, energy and development costs to run an application on a certain architecture. Our focus is on the development cost, which is composed of the development effort [person-days] and the salary [$ per person-day]. To define productivity as a metric with predictive power, we need to estimate the development effort in HPC. This work presents preliminary and ongoing work in this area.

II. Development Effort Estimation

Development effort estimation models are used in software companies to predict the cost of writing new software [2]. However, they are often not directly applicable to (non-commercial) HPC centers, since HPC codes usually require spending much development effort into parallelization or tuning. For instance, the 80-20 rule suggests a Pareto-like distribution for the relationship of effort and performance in HPC. Contrary, the popular COCOMO II software cost estimation model [3] predicts effort as function of lines of code and correlates with performance only within a bounded range.

TABLE I
Preliminary ideas on a methodology to estimate development effort and productivity parameters such as the impact of the programming model or numerical algorithm.

Qualifying	**Impact factors & their level of impact**		Quantifying
	Questionnaires: gathering & weighting parameters; correlation analysis	Development diaries, development effort tracking tool, error propagation	
	Impact of programming model		
	Statistical significance of effort & performance; questionnaires	Regression analysis, guide on how to add new programming models	
	Impact of algorithm		
	Analysis of effort-intensive code patterns in proxy apps [4]; questionnaires	Instrument code patterns for development effort tracking	

In our work, we aim to create a model for development effort estimation in HPC and incorporate this into the productivity metric (Eq. 1). Development effort is dependent on numerous impact factors including pre-knowledge and capability of the developer, performance, programming model, tool landscape or kind and size of numerical algorithm. To tackle this challenge, we restrict our model to certain use cases in the first approach.

III. Methodology

Our methodology focuses on development effort and productivity parameters by covering a qualitative analysis first, followed by a quantification of the model parameters. We start by investigating numerous impact factors on development effort in HPC and focus on the ones with highest impact. Our first focus is on the impact of the used programming model and the used numerical algorithm and their correlations. An overview of the planned activities can be found in Tab. I.

IV. Preliminary Results

We started by designing a questionnaire that (also) captured numerous impact factors on development effort. Participants were asked to find their most and least productive setups. For the latter (e.g. no pre-knowledge on the programming

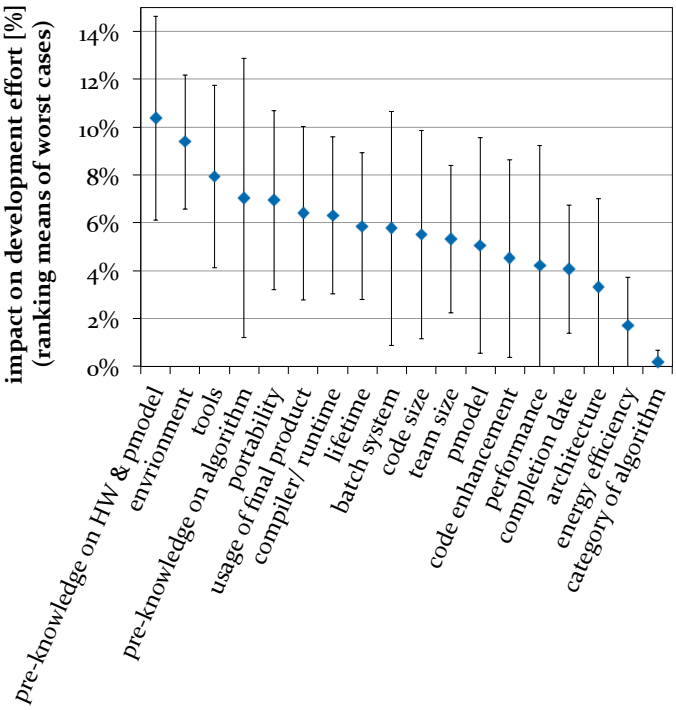

Fig. 1. Parameter ranking with respect to their impact level on development effort. Results gained by questionnaire [5]. Sample size is 8. Error bars indicate $\pm\sigma$.

model), they ranked parameters according to their impact levels. Sample data came from employees and students doing a 24-hour Hackathon in different computer science areas at Lawrence Livermore National Laboratory [5] (see Fig. 1). While having *pre-knowledge* on rank one is expected, it is surprising that influences of the *environment* and *tools* are ranked second and third. Additionally, impact factors are ranked differently to listed impact factors in company setups (COCOMO II) [2]. However, final conclusions cannot be derived because the sample size is small, the error bars are not negligible, not all participants focused on HPC problems and few questions were not understood well (e.g., *category of algorithm*). Nevertheless, results indicate that our methodology is a good first approach, while it needs to be extended to a bigger sample size and HPC emphasis.

In a second study [6], we analyzed the impact of parallel programming models on development effort and productivity. Students had to implement parallel CG solver versions using OpenMP on CPUs, OpenACC and CUDA on GPUs. Application performance was measured at the end of the semester and development diaries documented the effort spent. These results were processed by the one-sided signed Wilcoxon rank sum test. This statistical test returns the probability p that differences between samples are only due to chance. Figure 2 shows that both OpenMP and OpenACC need less development effort than an implementation with CUDA. Looking at the simplified productivity ratio *performance per effort*, this method illustrates that OpenMP and OpenACC both tend to more productivity than CUDA. Thus, the programming model

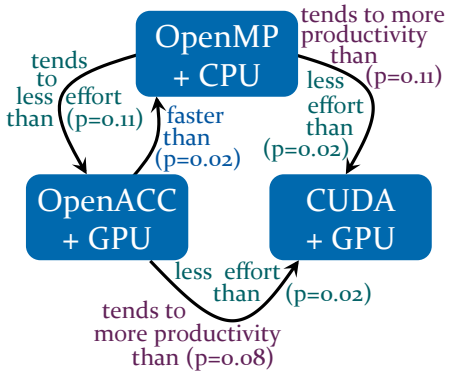

Fig. 2. Comparison of three programming models in development effort, performance and productivity using the one-sided signed Wilcoxon rank sum test. Sample size is 6 student groups. If $p < 0.05$, we say "less than". If $0.05 < p < 0.15$, we say "tends to less".

plays a role in productivity analysis and can be investigated by statistical methods.

V. Conclusion

Our work presents a first methodology to qualify and quantify development effort and productivity in HPC. Various empirical studies must be conducted to gain an appropriate database to rely on. Statistical methods and regression analysis help to analyze this data and to estimate development effort.

First results show some applicability of our methodology. However, future analysis must rely on bigger sample sizes in the HPC domain. We continue to gather data from software labs at RWTH Aachen University and Lawrence Livermore National Lab and use Hackathons as further data source. Also, help us to gain data sets: take part at our online survey [7]!

Acknowledgements

Part of this work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344 (LLNL-ABS-676032).

References

[1] S. Wienke, H. Iliev, D. an Mey, and M. S. Müller, "Modeling the Productivity of HPC Systems on a Computing Center Scale," in *High Performance Computing*, ser. LNCS, J. M. Kunkel and T. Ludwig, Eds. Springer International Publishing, 2015, vol. 9137, pp. 358–375.

[2] S. McConnell, *Software Estimation: Demystifying the Black Art*. Redmond, Wa.: Microsoft Press, 2006.

[3] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost models for future software life cycle processes: Cocomo 2.0," *Annals of Software Engineering*, vol. 1, no. 1, pp. 57–94, 1995.

[4] Lawrence Livermore National Laboratory, "LLNL ASC Proxy Apps," https://codesign.llnl.gov/proxy-apps.php, 2015.

[5] ——, "Hackathon in Computation," http://computation.llnl.gov/newsroom/cross-pollination-people-and-ideas-summer-hackathon, 2015.

[6] RWTH Aachen University, "Software Lab of Parallel Programming Models for Applications of High-Performance Computing (HPC)," http://www.hpc.rwth-aachen.de/teaching/lab, 2014.

[7] S. Wienke, "Productivity Questionnaire in HPC," http://www.hpc.rwth-aachen.de/research/tco/productivity/survey, 2015.