A Proposal for a Distributed Performance Analysis Tool for a Task-based Parallelism Framework

Christian Asch[1,2], **Lucas Mello Schnorr**[3], **Esteban Meneses**[1,4]

[1]Centro Nacional de Alta Tecnología

[2]Universidad de Costa Rica

[3]Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil

[4]Instituto Tecnológico de Costa Rica

{casch,emeneses}@cenat.ac.cr,schnorr@inf.ufrgs.br

Abstract. *We propose CharmVZ, a distributed Python-based performance analysis tool for applications developed with Charm++. Inspired by StarVZ, CharmVZ addresses the limitations of current tools by offering a distributed option to analyze large-scale traces of widely used applications such as ChaNGa and NAMD. Additionally, by integrating with the Python ecosystem, CharmVZ will enable new ways of analyzing data, for example, through machine learning, making it a novel tool for optimizing HPC applications.*

1. Introduction

Task-based parallelism is a programming paradigm in which the problem to be solved is divided into different steps, called tasks, that are related and ordered by their data dependencies. In this way, the problem can be represented as a direct acyclic graph (DAG), and the tasks can be scheduled so that those that are independent can be executed simultaneously in different processing elements, reducing the time to solution of the whole program. Since libraries that implement this framework often have a runtime system with knowledge of available hardware, they can implement steps to mitigate the computational load imbalance by dynamically allocating work to underutilized resources. Task-based parallelism libraries often support different profiling methods that help to optimize software and diagnose performance bottlenecks. StarVZ [Nesi et al. 2020] is an R-based performance analysis tool for the StarPU task-based parallelism framework [Augonnet et al. 2009]. It allows for powerful statistical analysis of the data and the creation of rich visualizations. The Charm++ task-based parallel programming model [Kale and Krishnan 1993] has a similar need to visualize traces, as is used in large-scale parallel applications such as ChaNGa, a cosmological simulation framework [Jetley et al. 2008], and NAMD, a widely used nanoscale molecular dynamics application [Phillips et al. 2020]. Both programs require high performance, scalability, and load balancing in massive distributed systems, which makes it essential to profile their behavior effectively; additionally, their traces can be large enough that there would be a benefit in using distributed computing for their analysis, similar to how it is done with MPI+OpenMP software using the Scalasca performance analysis tool [Geimer et al. 2010]. This paper proposes CharmVZ, a Python-based tool inspired by StarVZ and powered by the distributed computing library Dask to analyze Charm++ traces, facilitating the performance visualization of applications such as ChaNGa and

NAMD. Tapping into the Python ecosystem will allow programmers to analyze their data with a vast array of techniques, from statistical analysis to machine learning.

2. Proposal

To develop CharmVZ, it will be necessary to follow a two-stage approach similar to that of StarVZ. The first stage will be involved with the preprocessing and transformation of the tracing data generated by Charm++, while the second stage is mainly involved in the actual analysis and visualization of the data. CharmVZ will address limitations in the Projections' data presentation using Python data visualization libraries, to create flexible visualizations. These visualizations will help users explore execution timelines, communication patterns, and load balancing behavior more effectively, it will also allow users to filter data, compare different execution runs, and export data for further analysis. The tool will be designed with key Charm++ features in mind, for example, CharmVZ will provide visualizations that highlight message-passing patterns and communication overhead to help developers identify potential bottlenecks, it will visualize task over-decomposition and dependencies using a DAG, making it easier to understand task parallelism and its impact on performance, and it will show how Charm++ dynamically redistributes tasks across processors, offering insights into task migrations, resource usage, and the effectiveness of different load-balancing strategies. To analyze the huge traces of complex programs, CharmVZ will make use of the Dask, which allows for the easy and transparent scaling up of computational resources. By integrating with Python data manipulation libraries, CharmVZ will allow users to perform an in-depth analysis of their performance data. This will enable users to apply statistical analysis, create customized visualizations, and train parameter optimization algorithms, as well as improve the ability to diagnose and resolve performance issues in Charm++ applications like NAMD and ChaNGa.

References

Augonnet, C., Thibault, S., Namyst, R., and Wacrenier, P.-A. (2009). Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. In *Euro-Par 2009 Parallel Processing: 15th International Euro-Par Conference, Delft, The Netherlands, August 25-28, 2009. Proceedings 15*, pages 863–874. Springer.

Geimer, M., Wolf, F., Wylie, B. J., Ábrahám, E., Becker, D., and Mohr, B. (2010). The scalasca performance toolset architecture. *Concurrency and computation: Practice and experience*, 22(6):702–719.

Jetley, P., Gioachin, F., Mendes, C., Kale, L. V., and Quinn, T. (2008). Massively parallel cosmological simulations with changa. In *2008 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–12. IEEE.

Kale, L. V. and Krishnan, S. (1993). Charm++ a portable concurrent object oriented system based on c++. In *Proceedings of the eighth annual conference on Object-oriented programming systems, languages, and applications*, pages 91–108.

Nesi, L. L., Pinto, V. G., Miletto, M. C., and Schnorr, L. M. (2020). Starvz: Performance analysis of task-based parallel applications.

Phillips, J. C., Hardy, D. J., Maia, J. D., Stone, J. E., Ribeiro, J. V., Bernardi, R. C., Buch, R., Fiorin, G., Hénin, J., Jiang, W., et al. (2020). Scalable molecular dynamics on cpu and gpu architectures with namd. *The Journal of chemical physics*, 153(4).