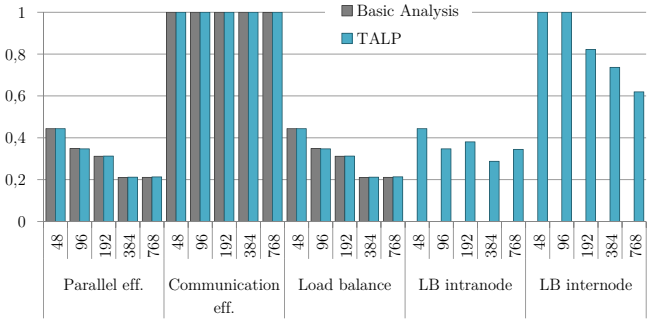

Figure 15: Alya complex efficiencies with TALP and Basic analysis

would not benefit so much from running with a round robin distribution of MPI processes among nodes as the main problem is within the most loaded node.

5 CONCLUSIONS AND FUTURE WORK

In this paper we present TALP, a transparent to the application, lightweight, and scalable tool to measure the parallel efficiency of MPI applications. It can be utilized by inexperienced users just by adding a couple of lines to the submission script and it will report the POP metrics about parallel efficiency. These metrics capture fundamental behaviour of the execution, indicating if the run has an acceptable parallel performance, or if not which are the main factors limiting it.

Additionally it offers an API for advanced users, that allow measuring specifics part of the code in case the user is interested in having a more detailed report. The API can also be used by resource managers or auto-tuning applications that want to adapt the execution dynamically at runtime. The API gives information that can be used to decide on requesting more resources or on the contrary releasing them to achieve a target parallel efficiency.

We show that the overhead added is not relevant while the frequency of MPI calls is below 50 MPI calls per millisecond, this corresponds to bursts of useful computation of 20 μseconds. We demonstrate the use of TALP with and without the API with two widely used HPC scientific applications.

Finally, we introduce two new metrics that can give relevant information on the Load balance efficiency and how to address it. The two new metrics differentiate between the load balance inside the nodes and across nodes, telling us if it is worth to use the DLB library to load balance, and to launch he application with a round robin distribution of processes across nodes.

As future work we plan to add the option of collecting hardware counters during the execution, and also to measure the frequency of MPI calls done by the application, in case that it is above the threshold of acceptable overhead, we can either deactivate TALP or emit a warning to the users advising them to take the overhead into account.

ACKNOWLEDGMENTS

This work is partially supported by the Spanish Government through Programa Severo Ochoa (SEV-2015-0493), by the Spanish Ministry of Science and Technology (TIN2015-65316-P), by the Generalitat de Catalunya (2017-SGR-1414), and by the European POP CoE (GA n. 824080).

REFERENCES

[1] Fabio Banchelli, Kilian Peiro, Andrea Querol, Guillem Ramirez-Gargallo, et al. 2020. Performance study of HPC applications on an Arm-based cluster using a generic efficiency model. In *2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*. IEEE, 167–174.

[2] Holger Brunst, Daniel Hackenberg, Guido Juckeland, and Heide Rohling. 2010. Comprehensive performance tracking with vampir 7. In *Tools for High Performance Computing 2009*. Springer, 17–29.

[3] BSC tools for performance analysis. [n.d.]. https://tools.bsc.es.

[4] Marc Casas, Rosa Badia, and Jesús Labarta. 2008. Automatic analysis of speedup of MPI applications. In *Proceedings of the 22nd annual international conference on Supercomputing*. 349–358.

[5] CoE Performance Optimization and Productivity (POP). [n.d.]. https://pop-coe.eu/.

[6] Marco D'Amico, Marta Garcia-Gasulla, Víctor López, Ana Jokanovic, Raül Sirvent, and Julita Corbalan. 2018. DROM: Enabling Efficient and Effortless Malleability for Resource Managers. In *Proceedings of the 47th International Conference on Parallel Processing Companion*. ACM, 41.

[7] Alexandre E Eichenberger, John Mellor-Crummey, Martin Schulz, Michael Wong, Nawal Copty, Robert Dietrich, Xu Liu, Eugene Loh, and Daniel Lorenz. 2013. OMPT: An OpenMP tools application programming interface for performance analysis. In *International Workshop on OpenMP*. Springer, 171–185.

[8] Marta Garcia, Jesus Labarta, and Julita Corbalan. 2014. Hints to improve automatic load balancing with LeWI for hybrid applications. *J. Parallel and Distrib. Comput.* 74, 9 (2014), 2781–2794.

[9] Marta Garcia-Gasulla, Guillaume Houzeaux, Roger Ferrer, Antoni Artigues, Victor López, Jesús Labarta, and Mariano Vázquez. 2019. MPI+ X: task-based parallelisation and dynamic load balance of finite element assembly. *International Journal of Computational Fluid Dynamics* 33, 3 (2019), 115–136.

[10] Marta Garcia-Gasulla, Filippo Mantovani, Marc Josep-Fabrego, Beatriz Eguzkitza, and Guillaume Houzeaux. 2020. Runtime mechanisms to survive new HPC architectures: a use case in human respiratory simulations. *The International Journal of High Performance Computing Applications* 34, 1 (2020), 42–56.

[11] Markus Geimer, Felix Wolf, Brian JN Wylie, Erika Ábrahám, Daniel Becker, and Bernd Mohr. 2010. The Scalasca performance toolset architecture. *Concurrency and Computation: Practice and Experience* 22, 6 (2010), 702–719.

[12] Sergi Girona, Jesús Labarta, and Rosa M Badia. 2000. Validation of Dimemas communication model for MPI collective operations. In *European Parallel Virtual Machine/Message Passing Interface UsersâĂŹ Group Meeting*. Springer, 39–46.

[13] Jürg Hutter, Marcella Iannuzzi, Florian Schiffmann, and Joost VandeVondele. 2014. cp2k: atomistic simulations of condensed matter systems. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 4, 1 (2014), 15–25.

[14] Susan L Graham Peter B Kessler and Marshall K McKusick. 1982. gprof: a Call Graph Execution Profiler1. In *Proceedings of the Symposium on Compiler Construction, pp.–. Press, New York„. Cited on p.. Greco, Gianluigi.* Citeseer.

[15] Andreas Knüpfer, Christian Rössel, Dieter an Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen Malony, et al. 2012. Score-p: A joint performance measurement run-time infrastructure for periscope, scalasca, tau, and vampir. In *Tools for High Performance Computing 2011*. Springer, 79–91.

[16] Thomas D Kühne, Marcella Iannuzzi, Mauro Del Ben, Vladimir V Rybkin, Patrick Seewald, Frederick Stein, Teodoro Laino, Rustam Z Khaliullin, Ole Schütt, Florian Schiffmann, et al. 2020. CP2K: An electronic structure and molecular dynamics software package-Quickstep: Efficient and accurate electronic structure calculations. *The Journal of Chemical Physics* 152, 19 (2020), 194103.

[17] Vincent Pillet, Jesús Labarta, Toni Cortes, and Sergi Girona. 1995. Paraver: A tool to visualize and analyze parallel code. In *Proceedings of WoTUG-18: transputer and occam developments*, Vol. 44. Citeseer, 17–31.

[18] Prace, UEABS. [n.d.]. https://repository.prace-ri.eu/git/UEABS/ueabs/.

[19] Harald Servat, Germán Llort, Kevin Huck, Judit Giménez, and Jesús Labarta. 2013. Framework for a productive performance optimization. *Parallel Comput.* 39, 8 (2013), 336–353.

[20] Sameer S Shende and Allen D Malony. 2006. The TAU parallel performance system. *The International Journal of High Performance Computing Applications* 20, 2 (2006), 287–311.

[21] Mariano Vázquez, Guillaume Houzeaux, Seid Koric, Antoni Artigues, Jazmin Aguado-Sierra, Ruth Arís, Daniel Mira, Hadrien Calmet, Fernando Cucchietti, Herbert Owen, et al. 2016. Alya: Multiphysics engineering simulation toward exascale. *Journal of computational science* 14 (2016), 15–27.

[22] Michael Wagner, Stephan Mohr, Judit Giménez, and Jesús Labarta. 2017. A Structured Approach to Performance Analysis. In *International Workshop on Parallel Tools for High Performance Computing*. Springer, 1–15.