The P3 Explorer: Exploring the Performance, Portability, and Productivity Wilderness

Matthew A. Smith, Steven A. Wright
Department of Computer Science
University of York
York, UK
{ms3408, steven.wright}@york.ac.uk

Zaman Lantra, Gihan R. Mudalige
Department of Computer Science
University of Warwick
Coventry, UK
{zaman.lantra, g.mudalige}@warwick.ac.uk

***Abstract*—This paper documents the development of a web-based tool designed to organise and present visual representations of performance, portability, and productivity (P3) data from previously published scientific studies. The P3 Explorer operates as both an open repository of scientific data and a data dashboard, providing visual heuristic analyses of performance portability and developer productivity, created using the Intel's P3 Analysis library. The aim of the project is to create a community-led database of P3 studies to better inform application developers of alternative approaches to developing new applications targeting high performance on diverse hardware, with consideration of developer productivity.**

***Index Terms*—High Performance, Performance Portability, Developer Productivity**

I. Introduction

The P3 Explorer is an open data repository and exploration tool, designed to showcase Performance, Portability, and Productivity (P3) data with plots and insights automatically generated by Intel's P3 Analysis Library [1].

This provides the following benefits to the wider HPC community:

- A repository of user-submitted experimental data focused on the performance, portability, and productivity of HPC applications.
- A dashboard for exploring user-submitted data from P3 studies, using automatically-generated plots for visualising performance portability, and productivity.
- An improvement to the reproducibility of P3 studies with an explorable archive of configuration data, performance data, and P3 analysis.

The P3 Explorer provides an interface for application developers to assess a variety of approaches to developing new parallel software with the goal of maximising performance, portability, and productivity.

II. Context and Related Work

HPC hardware is diversifying, and there is a proliferation of parallel programming models that target heterogeneous systems, containing GPU accelerators from vendors such as NVIDIA, AMD, and Intel.

Application developers typically seek to maximise *performance*, *portability*, and *productivity* on modern heterogeneous HPC platforms [2].

A. Evaluating the Three Ps

Our data dashboard displays visual data relating to the performance, performance portability, and developer productivity of approaches to developing parallel software. Performance data typically takes the form of runtime, while performance portability is measured using the Pennycook metric (Eq. (1)) [3].

$$\mathbf{\Phi}(a,p,H) = \begin{cases} \dfrac{|H|}{\displaystyle\sum_{i \in H} \dfrac{1}{e_i(a,p)}} & \text{if } i \text{ supported } \forall i \in H \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where, the performance portability $\mathbf{\Phi}$ of an application a, solving problem p, on a given set of platforms H, is calculated by finding the harmonic mean of an application's performance efficiency ($e_i(a,p)$).

Developer productivity is evaluated using code divergence (Eq. (2)), where code divergence is a measure of the average "distance" between the source code required to compile an application a, and execute problem p for each pair of platforms in H [4]. The P3 Analysis Library uses the Jaccard distance between codebases (Eq. (3)).

$$\text{CD}(a,p,H) = \binom{|H|}{2}^{-1} \sum_{\{i,j\} \in H \times H} d_{i,j}(a,p) \tag{2}$$

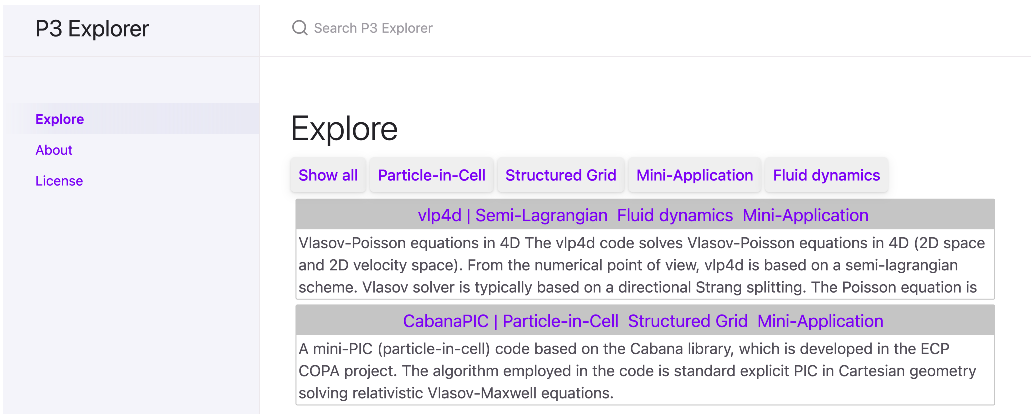
$$d_{i,j}(a,p) = 1 - \frac{|c_i(a,p) \cap c_j(a,p)|}{|c_i(a,p) \cup c_j(a,p)|} \tag{3}$$

To provide a more holistic view of performance portability and developer productivity, our dashboard visualises data using *performance portability cascade plots* and *performance portability code convergence* ($\mathbf{\Phi}$*-CC*) *charts* [5]. These allow users to better evaluate the P3 properties of an application visually.

III. The P3 Explorer

The P3 Explorer primarily serves as a data dashboard, allowing application developers to explore P3 data from previously published performance studies.

The dashboard is updated using CI/CD actions on GitHub, generating appropriate plots that demonstrate the performance, performance portability, and development productivity of an application, using relevant P3 visualisations.



P3 Explorer

Search P3 Explorer

Explore
About
License

Explore

Show all | Particle-in-Cell | Structured Grid | Mini-Application | Fluid dynamics

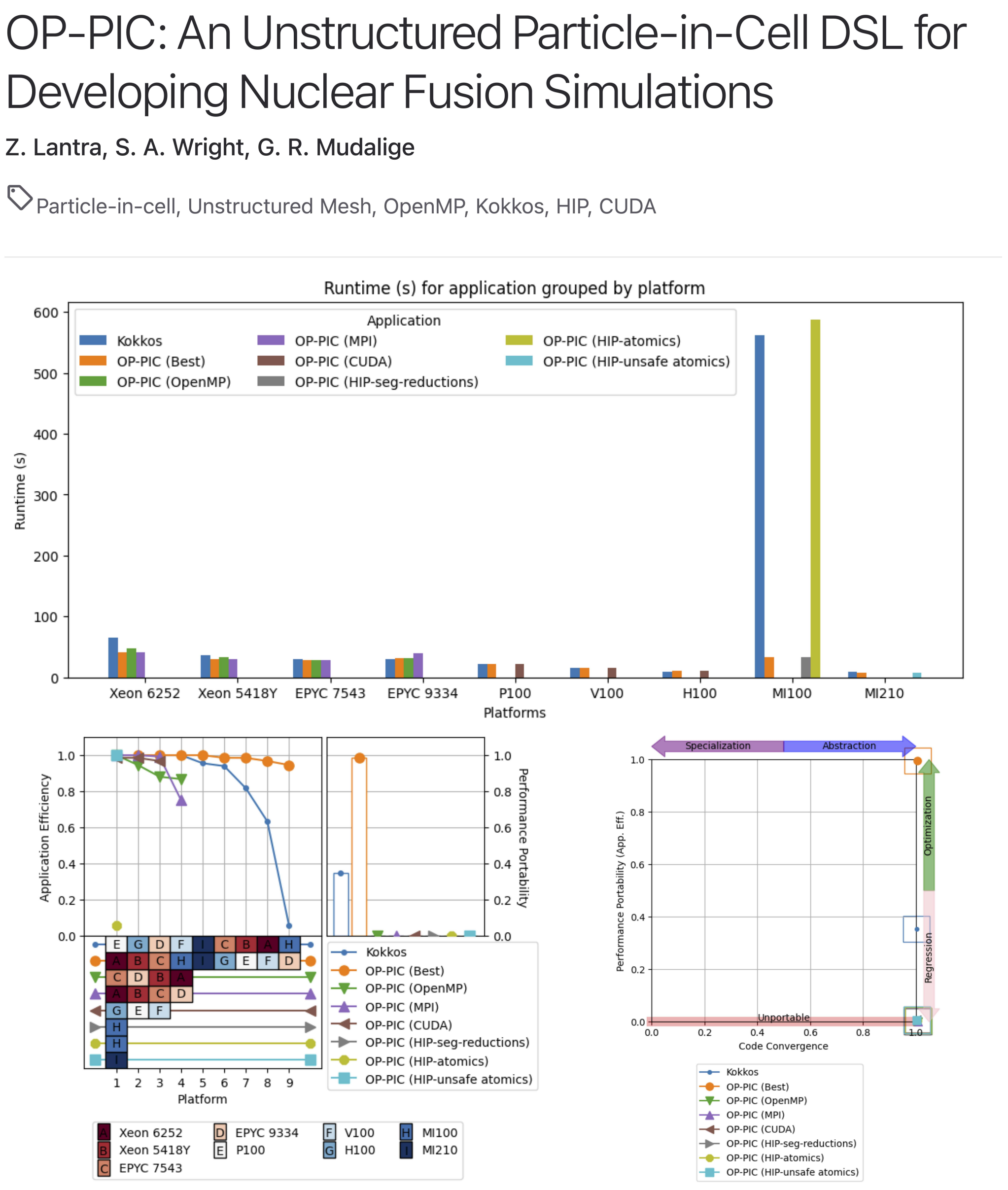
vlp4d | Semi-Lagrangian Fluid dynamics Mini-Application

Vlasov-Poisson equations in 4D The vlp4d code solves Vlasov-Poisson equations in 4D (2D space and 2D velocity space). From the numerical point of view, vlp4d is based on a semi-lagrangian scheme. Vlasov solver is typically based on a directional Strang splitting. The Poisson equation is

CabanaPIC | Particle-in-Cell Structured Grid Mini-Application

A mini-PIC (particle-in-cell) code based on the Cabana library, which is developed in the ECP COPA project. The algorithm employed in the code is standard explicit PIC in Cartesian geometry solving relativistic Vlasov-Maxwell equations.

Fig. 1: An example of the database exploration page



OP-PIC: An Unstructured Particle-in-Cell DSL for Developing Nuclear Fusion Simulations

Z. Lantra, S. A. Wright, G. R. Mudalige

Particle-in-cell, Unstructured Mesh, OpenMP, Kokkos, HIP, CUDA

Fig. 2: An example experimental data page for Lantra et al. [6]

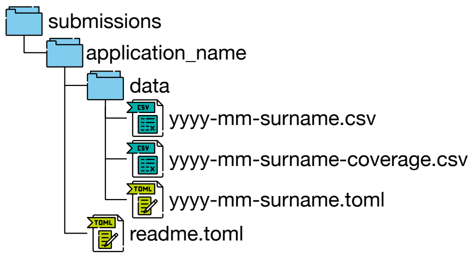


Fig. 3: An illustration of the submissions directory structure

```
title       = "CabanaPIC"
appDomain   = ["Particle-in-Cell", "Structured Grid", ...]
sources     = [["GitHub", ...]]
description = """A mini-PIC (particle-in-cell) code ..."""
```

Fig. 4: An example TOML file for a new application

Performance studies are archived on the dashboard organised by application, and categorised by scientific domain(s). Figure 1 shows an example of the dashboard's application exploration page.

Each study contains performance data, and a submission information file that provides artefact metadata (e.g., paper DOI, links to source code). Each application may contain multiple studies, and no attempt is made to combine datasets.

The dashboard itself is built using the static site generator, Jekyll, and is deployed using GitHub Pages. The P3 Explorer is available at: `https://p3-explorer.github.io`

A. An example

Each data exploration page typically contains:

- Metadata related to the study, including a description of how the data was gathered.
- A bar chart of submitted performance data, plotted using Matplotlib.
- Appropriate P3 plots, generated using the P3 Analysis Library [1].

Figure 2 shows an example study using the CabanaPIC application [6]. In this study, the OP-PIC domain specific language is compared to Kokkos. The generated visualisations show no code divergence in either implementation, since both are single-source approaches. OP-PIC is shown to offer considerably better performance portability, though Kokkos has a similar level of performance portability across most platforms (falling away on the Xeon CPUs and the MI100 GPU where hardware atomics are poorly supported).

```
title       = "OP-PIC: An Unstructured Particle-in-..."
authors     = ["Z. Lantra" , ...]
sources     = [
                 ["OP-PIC Repository", ...],
                 ["OP-PIC Documentation", ...],
              ]
doi         = "10.1145/3673038.3673130"
fom         = "Runtime (s)"
tags        = ["Particle-in-cell", ...]
description = """In this work we introduce OP-PIC ..."""
```

Fig. 5: An example TOML file for a new data submission

IV. Contributing

Contributing to the project is simple, and all of the data processing is done automatically on merging a pull request.

A pull request should contain a new application and associated data and/or data for an existing application within the submissions directory. Figure 3 shows the directory structure. All submissions must adhere to our submission guidelines and format; and as a minimum, data submissions should include performance data in CSV format (with column labels matching the P3 Analysis Library [1]), and a TOML file with information about the submission.

For a new application, an information TOML file is required in the application's root directory; Figure 4 shows an example for the CabanaPIC application. For a scientific performance study submission, Figure 5 shows a sample information file, where the description explains how the data was collected.

Optionally, code coverage data can be provided in CSV format to generate a Φ-CC chart. Datasets can be updated at any time, and the dashboard updates on a commit or merge.

References

[1] S. J. Pennycook, J. Sewall *et al.*, "Performance, Portability and Productivity Analysis Library," Mar. 2023.

[2] S. A. Wright, C. P. Ridgers *et al.*, "Developing performance portable plasma edge simulations: A survey," *Computer Physics Communications*, vol. 298, 2024.

[3] S. J. Pennycook, J. D. Sewall, and V. W. Lee, "Implications of a metric for performance portability," *Future Generation Computer Systems*, vol. 92, 2019.

[4] S. L. Harrell, J. Kitson *et al.*, "Effective performance portability," in *Proceedings of the IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)*, 2018, pp. 24–36.

[5] S. J. Pennycook, J. D. Sewall, D. Jacobsen, T. Deakin, and S. McIntosh-Smith, "Navigating Performance, Portability, and Productivity," *Computing in Science & Engineering*, vol. 23, no. 5, 2021.

[6] Z. Lantra, S. A. Wright, and G. R. Mudalige, "OP-PIC – an Unstructured-Mesh Particle-in-Cell DSL for Developing Nuclear Fusion Simulations," in *Proceedings of the International Conference on Parallel Processing*, ser. ICPP'24, 2024, pp. 294—304.