Introduction to PyPOP

Phil Tooley (phil.tooley@nag.co.uk) — NAG

EU H2020 Centre of Excellence

Grant Agreement No 824080

What is PyPOP?

What PyPOP is:

- Simple tool to automate common tasks in performance profiling
- Rapidly analyse traces and compute POP Metrics
- Easily generate high quality plots and reports
- User friendly and tool agnostic[1]
- Backend framework to build custom analyses using Python

[1] Extrae and manual input currently supported, other formats in development

What is PyPOP?

What PyPOP is:

- Simple tool to automate common tasks in performance profiling
- Rapidly analyse traces and compute POP Metrics
- Easily generate high quality plots and reports
- User friendly and tool agnostic[1]
- Backend framework to build custom analyses using Python

But...

- Only one part of the performance analysis workflow
- Not a substitute for manually inspecting traces
- Still under heavy development

[1] Extrae and manual input currently supported, other formats in development

What is PyPOP

PyPOP design choices and philosophy

- Python $\geq$3.6 with Numpy, Pandas, Bokeh
  - Widely used in science/industry — minimise barrier to entry
  - Plugin-based architecture for extensibility

What is PyPOP

PyPOP design choices and philosophy

- Python $\geq$3.6 with Numpy, Pandas, Bokeh
  - Widely used in science/industry — minimise barrier to entry
  - Plugin-based architecture for extensibility
- Jupyter notebook based interface
  - “Literate programming” encourages self-documentation and reproducibility
  - Easily mix text/code/plotting/GUI elements
  - Generate reports directly from an analysis notebook

What is PyPOP

PyPOP design choices and philosophy

- Python $\geq$3.6 with Numpy, Pandas, Bokeh
  - Widely used in science/industry — minimise barrier to entry
  - Plugin-based architecture for extensibility
- Jupyter notebook based interface
  - “Literate programming” encourages self-documentation and reproducibility
  - Easily mix text/code/plotting/GUI elements
  - Generate reports directly from an analysis notebook
- “Wizard”-like GUI for non Python-programmers
  - Generate POP metrics and reports without writing Python code

Typical PyPOP Workflow

POP Metrics from Extrae traces

1. Collect traces with Extrae — required data:
   - MPI Events
   - OpenMP Events
   - Hardware counters: Total Instructions, Total Cycles

Typical PyPOP Workflow

POP Metrics from Extrae traces

1. Collect traces with Extrae — required data:
   - MPI Events
   - OpenMP Events
   - Hardware counters: Total Instructions, Total Cycles
2. Optionally pre-process traces
   - Pre-process large (GBs) traces into small (KBs) summary files
   - Speeds notebook loading and minimizes data downloads

Typical PyPOP Workflow

POP Metrics from Extrae traces

3. Open Jupyter Notebook with “Wizard” GUI
   - Select and process traces or summary files using the GUI
   - Quick initial analysis to generate POP metrics

Typical PyPOP Workflow

POP Metrics from Extrae traces

3. Open Jupyter Notebook with “Wizard” GUI
   - Select and process traces or summary files using the GUI
   - Quick initial analysis to generate POP metrics
4. Generate report notebook from GUI
   - Report template notebook containing metric table and scaling plot
   - Add description/discussion text to notebook
   - Customise analysis using Python code
   - Convert notebook to PDF to create shareable report

Conclusion

A tool for efficient performance analysis workflows

- Python based with Jupyter notebook interface
- Quickly analyse traces and compute POP metrics
- Plot metric tables and scaling graphs
- Output fully annotated PDF reports

PyPOP Github

- https://github.com/numericalalgorithmsgroup/pypop

Performance Optimisation and Productivity

A Centre of Excellence in HPC

Contact:

- https://www.pop-coe.eu
- pop@bsc.es
- @POP_HPC
- youtube.com/POPHPC

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 676553 and 824080.