Performance Optimisation and Productivity

Nick Dingle

nick.dingle@nag.co.uk

EU H2020 Centre of Excellence (CoE)

Grant Agreement No 676553

1 October 2015 – 31 March 2018
Motivation

Why?

• Complexity of machines and codes
  → Frequent lack of quantified understanding of actual behaviour
  → Not clear most productive direction of code refactoring

• Important to maximize efficiency (performance, power) of compute intensive applications and productivity of the development efforts

What?

• Parallel programs, mainly MPI/OpenMP
  • Although also CUDA, OpenCL, OpenACC, Python, ...
POP CoE

- A Centre of Excellence
  - On Performance Optimisation and Productivity
  - Promoting **best practices in parallel programming**
- Providing Services
  - Precise **understanding** of application and system behaviour
  - Suggestion/support on how to refactor code in the most productive way
- Horizontal
  - Transversal across application areas, platforms, scales
- For academic and industrial codes and users
- **FREE !**
Services provided by the CoE

Parallel Application Performance Audit
- Primary service
- Identify performance issues of customer code (at customer site)
- Small effort (< 1 month)

Parallel Application Performance Plan
- Follow-up on the audit service
- Identifies the root causes of the issues found and qualifies and quantifies approaches to address them
- Longer effort (1-3 months)

Proof-of-Concept
- Experiments and mock-up tests for customer codes
- Kernel extraction, parallelisation, mini-apps experiments to show effect of proposed optimisations
- 6 months effort
Outline of a typical audit report

• Application Structure
• (if appropriate) Region of Interest
• Scalability Information
• Application Efficiency
  • E.g. time spent outside MPI
• Load Balance
  • Whether due to internal or external factors
• Computational Performance
  • Identification of areas for improvement
• Communications
  • E.g. sensitivity to network performance
• Summary and Recommendation
The process ...

When?

October 2015 – March 2018

How?

• Apply
  • Fill in small questionnaire describing application and needs [https://pop-coe.eu/request-service-form](https://pop-coe.eu/request-service-form)
  • Questions? Ask pop@bsc.es
  • Selection/assignment process
  • Install tools @ your production machine
  • Interactively: Gather data → Analysis → Report
• How responsive have the POP experts been to your questions or concerns about the analysis and the report?

• What was the quality of their answers?
Partners

• **Who?**
  - BSC (coordinator), ES
  - HLRS, DE
  - JSC, DE
  - NAG, UK
  - RWTH Aachen, IT Center, DE
  - TERATEC, FR

A team with

• Excellence in performance tools and tuning
• Excellence in parallel programming models and practices
• Research and development background AND
  proven commitment in application to real academic and industrial use cases
Target customers

• **Code developers**
  • Assessment of detailed actual behaviour
  • Suggestion of most productive directions to refactor code

• **Users**
  • Assessment of achieved performance in specific production conditions
  • Possible improvements modifying environment setup
  • Evidence to interact with code provider

• **Infrastructure operators**
  • Assessment of achieved performance in production conditions
  • Possible improvements from modifying environment setup
  • Information for time computer time allocation processes
  • Training of support staff

• **Vendors**
  • Benchmarking
  • Customer support
  • System dimensioning/design
Tools

• Install and use already available monitoring and analysis technology

• Open-source toolsets
  • Extrae + Paraver
  • Score-P + Cube + Scalasca/TAU
  • Dimemas, Extra-P
  • SimGrid

• Commercial toolsets (if available at customer site)
  • Intel tools
  • Cray tools
  • Allinea tools
Code Audit Examples
GITM (Cefas)

• An offline particle tracking model code
• Written in Fortran with OpenMP
• Key audit results:
  • Current performance: 16 threads offer 5x speed-up vs 1 thread
  • Load imbalance amongst OpenMP threads
  • Maximise opportunities for vectorisation by aligning arrays and refactoring Fortan array operations to use DO loops
  • Large number of floating-point divisions

28/10/2016
dwarf-D-ellipticSolver-GCR (ECMWF)

- Extracted from ECMWF's Integrated Forecasting System (IFS) code
- Mixed Fortran/C++ code with hybrid MPI+OpenMP
- Key audit results:
  - Good scalability as communication pattern introduces little overhead from synchronisation or serialisation
  - Minor load imbalance due to variability in IPC across nodes
  - Opportunities to further increase performance as IPC is 0.8 on average
NEMO (Atos)

• Undertaken as part of the ESCAPE (Energy-efficient Scalable Algorithms for Weather Prediction at Exascale) project

• Written in Fortran with MPI

• Traces gathered by user and analysed by POP

• Key audit findings:
  • Good computational load balance
  • Observed superlinear speed-up of analysed region of the program, which was due to improved cache efficiency when strong scaling

28/10/2016
Other POP activities

• **Customer advocacy**
  - Gather customers feedback, ensure satisfaction, steer activities

• **Sustainability**
  - Explore business models

• **Training**
  - Best practices on the use of the tools and programming models
This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 676553.