D[6.2]– Performance Tuning Workflow developed and published Version [1.0]

Document Information

Contract Number	676553
Project Website	www.pop-coe.eu
Contractual Deadline	M24
Dissemination Level	PU
Nature	DEC
Authors	Dirk Schmidl (RWTH Aachen)
Contributors	
Reviewers	
Keywords	Training, Learning Material

Notices: *The research leading to these results has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No n°676553.*

Change Log

Version	Author	Description of Change
v0.1		Initial version of the POP LaTeXtemplate
v0.2	Dirk Schmidl (RWTH)	First draft for internal review.
v0.3	Christoph Niethammer (USTUTT (HLRS))	Internal Review.
v1.0	Dirk Schmidl (RWTH)	Incorporated changes from internal review.

Contents

Executive Summary

This deliverable is part of work package 6 "Training and Documentation". The goal of this deliverable is to describe and publish the general performance tuning workflow, developed as part of POP, as best practice guide. This will enable users to start creating a performance analysis report as it is provided by POP themselves. Of course this report might not have the same level of detail in some parts, where individual expert knowledge is necessary, but it will show a structured way to generate a report. The described methodology will allow code developers to integrate regular performance analysis and optimization steps into their own software development cycle. The goal is to allow them to do a regular performance review of their code and to store them in a comparable way, as it is done in a POP audit.

1 Introduction

In the POP project several partners joint forces to provide services in the area of performance analysis and optimization. All partners have already had a long standing background in performance optimization. However, the methodologies applied by the partners differed a lot before the POP project and in many cases it was based more on expert knowledge than on a well-defined methodology. Still, the expert knowledge let to valuable insights in very short time. However, one of the goals in the POP project was to come up with a mostly standardized way of applying performance analysis for the following two reasons:

1. If a service is offered by POP, the output of the service should be specified and the output should be mostly independent of the person performing the request. This makes results comparable between studies, e.g. if a code is evaluated regularly after code changes. This is possible with the developed workflow, but it would not be achievable with an undefined expert knowledge based approach.

2. The described way of doing a performance study, developed in POP, can be performed also by non performance analysis experts to a certain extend. This allows developers to integrate performance analysis in their software development cycle by applying our workflow themselves. In many cases they should be able to verify the effect of an optimization with the given instructions in the workflow. Once a point is reached where further expert knowledge is needed, of course they can still apply for a POP service as usual.

The partners in POP use or even develop a variety of different performance analysis tools. Hence they know that there is not one tool best suitable for all scenarios. With this in mind, the workflow was developed to be suitable independent of the used tools. Although in practice the focus was mostly on the tools developed by POP partners, the methodology, which came out of POP, is now independent of the used performance tool.

The core of our standardized methodology is the POP performance audit, which is provided as the introductory service in POP. Here all the POP partners merged their experiences to come up with a well-structured document to give a good performance overview for parallel applications. The way such a performance audit is created is documented under "Learning Material" on our website[1] in "How to create a POP performance audit". The workflow is documented and published in two parts: (1) General workflow description, and (2) tool specific HowTos and examples.

2 Workflow Description

The POP workflow describes the general methodology how to create a performance audit. The workflow description document How to create a POP performance audit targets to describe the data that should be gathered within this workflow and helps to interpret this to find the cause of an inefficiency. It does not describe how this is done with an individual tool. In the following a short overview of the content in the workflow description document is given This HowTo describes the workflow to obtain a POP audit. The steps therein follow the structure of the POP audit report as it is defined in the deliverable D4.1. Beside the instructions additional information to increase the general understanding for each part are provided.

The HowTo includes the following information:

[1] https://pop-coe.eu/sites/default/files/pop_files/whitepaperperformanceaudits.pdf

1. An overview and a short description of the sections in a POP audit, `Background`, `Application Structure`, `Region(s) of Interest`, `Scalability Information`, `Application Efficiency`, `Load Balance`, `Serial Performance`, `Communication`, `Summary and Recommendations`, is provided to the reader to make him familiar with the general structure of an audit.

2. A description is given how to make a first analysis and identify regions of interest in the code. This can be a temporal overview by using a timeline analysis or a structural analysis based on call-path information. This advises the user how to give an overview of the code and identify where to focus on for the rest of the audit.

3. A scalability analysis of the application and the regions of interest is presented next in a report. Here first information is gathered on the scaling of the application. This can give a general overview about the overall efficiency of the application. This might help to identify which regions scale linearly and which regions need to be further investigated.

4. Then the efficiency metrics and their use in a POP report are explained in detail. These metrics are an essential part to exploit the characteristics of the application or individual regions with non-optimal scalability. Here, the How To does not focus on the generation of these metrics, but helps users with interpreting them. The user is instructed about the meaning of the metrics and in which cases a deeper analysis of one or the other kind is needed.

5. The next three sections of a POP audit are about the deeper analysis of three fields of performance issues: load imbalance, serial performance and communication. The described metrics advice the user to identify the region of the code, for which deeper analysis is needed. Here our workflow document gives hints what might be a good next analysis step. Furthermore, it provides general advises on how to fix found inefficiencies in the given application. These advises are based on the experiences gathered in POP. However, at this level problems might show up which are not handled by our document, in such cases users should contact a POP expert.

3 Tool specific HowTos/Examples

The POP workflow is not limited to a specific tool. However, the user will have to pick a tool to obtain the necessary data and perform the analysis. If the user is interested in more details how to use the tools of the POP partners to obtain and analyse the specific performance data he is referred to the user guides for the tools on the POP website. POP provides there user guides for BSC's tools (Paraver, Dimemas, and Extrae) as well as JSC's tools (Score-P, Cube, and Scalasca). With this information a user can generate a trace or profile and navigate through the performance data to do do the first steps of a performance analysis.

The creation of the mentioned efficiency metrics can be more complicated. To help with this task a set of exercises coming with solutions were developed. A user is guided step by step through the exercises generating the efficiency metrics. With this material an inexperienced user can start generating a POP performance audit with one of the partners' tools.

4 Summary

During the POP projects, a workflow has been developed to generate a performance audit. An audit provides a structured way to analyse and present data about the parallel efficiency of an application. If possible it will identify performance issues and provide ideas how they can be addressed.

This workflow is useful on the one hand to define the outcome of the service offered in POP and on the other hand it allows a step by step description of how performance analysis can be performed. Following this standardized approach enables non-experts in the field to perform useful performance analysis. After agreeing on a standardized way for such a workflow based on the experience of the POP partners and based on results gathered within the POP project, the document "How to generate a POP performance audit" was written for the non-experts. This was published on the POP website. The general workflow can be performed with any tool that can measure and analyse the required data. However,POP provides tool specific step-by-step guides as a set of examples and solutions for the POP partners' tools Score-P/Scalasca/Cube and Extrae/Paraver/Dimemas on its website, which can be used by non-experts.

Overall, the provided material will allow users to do a performance analysis of their code on their own, up to a certain point, where individual expert knowledge is needed. This hopefully allows code developers to integrate such a performance check into their software development cycle ensuring that performance is investigated regularly for their parallel high-performance application.

Acronyms and Abbreviations

- POP: Performance Optimization and Productivity
- RWTH Aachen: Rheinisch-Westfaelische Technische Hochschule Aachen
- BSC: Barcelona Supercomputing Center
- JSC: Jlich Supercomputing Center
- USTUTT (HLRS): University of Stuttgart