Project no. 248828

ADVANCE

Strategic Research Partnership (STREP)
ASYNCHRONOUS AND **D**YNAMIC **V**IRTUALISATION THROUGH PERFORMANCE
ANALYSIS TO SUPPORT **C**ONCURRENCY **E**NGINEERING

2^{nd} User Community Workshop

D23

Due date of deliverable: July 31^{st}, 2012
Actual submission date: July 31^{st}, 2012

Start date of project: February 1^{st}, 2010

Type: Deliverable
WP number: WP8

Responsible institution: HERTS
Editor & and editor's address: Raimund Kirner
University of Hertfordshire, College Lane
Hatfield, AL10 9AB, United Kingdom

Version 1.0 / Last edited by Raimund Kirner / July $31^{s}t$, 2012

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	$\surd$
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history:

Version	Date	Authors	Institution	Section affected, comments
0.1	18/02/2012	Raimund Kirner	HERTS	Initial version
1.0	30/07/2012	Raimund Kirner	HERTS	Feedback included

Reviewers:
Frank Penczek, Alex Shafarenko, Clemens Grelck

Tasks related to this deliverable:

Task No.	Task description	Partners involved°
WP8	Community Building	HERTS*, UvA

°This task list may not be equivalent to the list of partners contributing as authors to the deliverable
*Task leader

Executive Summary

Within the **Advance** project we extent key technologies for the development and execution of concurrent programs to deploy the optimisation potential of runtime optimisations based on statistical program information.

In this document we describe the second user community building activity for the key technology used within the **Advance** project, which was held from 27. September 2010 till 1. October 2010 at the Siberian Branch of the Russian Academy of Sciences in Novosibirsk. Attendees of the workshop learned the fundamental concepts of programming parallel applications with S-Net and/or SAC. The workshop was open to the public, with most participants having been graduate students at the Siberian Branch of the Russian Academy of Sciences.

Contents

Chapter 1

Introduction

From the programmer's perspective, the two main technologies used in the **Advance** project are S-Net and SAC. S-Net [4, 5, 1] is a coordination language that is targeted for course-grain parallelism. S-Net can be also used to reuse legacy code for parallel execution with some code modifications in order to split a program up into different components, called boxes in S-Net terminology. SAC [3, 2] on the other side is a functional programming language that kind of mimics the syntax of ISO C, to give programmers a feeling of familiarity on their first contact. The main source of parallelism in SAC programs is provided by array operations.

Within the **Advance** project the goal is to extend the S-Net and SAC technologies with statistical feedback in order to trigger performance optimisations. The approach is meant to be validated by industrial case studies provided by the industrial partners. Other academic partners will adapt their tools or frameworks to be used within the project.

To disseminate the **Advance** project at a wider community and to learn feedback about the usability of the programming approach with S-Net and SAC, we organised a second user community workshop at the Siberian Branch of the Russian Academy of Sciences; 17 Lavrentjev Street, 630090 Novosibirsk. Participants were mainly graduate students of the Siberian Branch of the Russian Academy of Sciences. The workshop was organised in the programming laboratory facilities of the institute.

In this document we give a brief summary of the second user community workshop, organised from 27. September till 1. October 2010 at the Siberian Branch of the Russian Academy of Sciences; 17 Lavrentjev Street, 630090 Novosibirsk.

Chapter 2

Overview of the Workshop

The second user community workshop was held on three days, starting with 27.09.2010 and ending on 31.10.2010. The user community workshop program is shown in Figure 2.1. The day of Wednesday, September 29^{th}, 2010 was left out of the user community workshop, as on this day we organised at the Ershov Institute of Informatics Systems a special project dissemination workshop focusing on an in-depth presentation of the design and technical aspects of S-Net and SAC.

	Mo, 27.09	Tu, 28.09	We, 29.09	Th, 30.09	Fr, 01.10
09:00-9:50	Foreward (**Fedoruk**):10 min. *Array functional programming: principles, techniques and tools* **(Sven-Bodo)**: 40min	*Principles of stream processing* (**Alex**)	**Workshop**	Masters and Doctorate students' session	Individual projects
Coffee-break					
10:20-11:10	*Programming in SaC* **(Sven-Bodo)**	*. The idea of coordinaton and principles of S-Net* **(Alex Shafarenko)**	**Workshop**		
Coffee-break					
11:40-12:30	*SaC programming patterns (case studies and comparative code analysis)* **(Clemens & Kudryavtsev)**	*5. Programming with SaC/S-Net* **(Clemens Grelck)**	**Workshop**		
Lunch					
14:30-16:00	Practical session	Practical session	**Workshop**	Visiting local attractions	Progress meeting
16:30-17:30	Practical session	Practical session	Presentation of projects for the forthcoming practical sessions		Progress meeting and closing talk

Figure 2.1: Program of the 2^{nd} User Community Workshop

The user community workshop itself effectively lasts three full working days.

- The first day started with an introduction to SAC. The rest of the day was spent on programming some simple SAC programs.
- On the second day of the workshop an introduction session on the design of S-Nethas been given. The afternoon was dedicated to practical programming sessions with S-Net.
- On the third day students were working on their own projects to be implemented with S-Net and/or SAC.
- On the fourth day the students were presenting the results of implementing their own projects and gave feedback on their user experience with the tools.

Chapter 3

Summary of the Programming Examples

The practical programming experience of the user community workshop consisted of two phases. In the first phase prepared small programming exercises for S-Net and SAC have been given. In the second phase the participants were given the freedom to chose a programming challenge that is related to their current study, e.g., an algorithm for their Master's thesis. In the following we summarise the prepared exercises of the first phase.

3.1 Programming in SAC

The SAC programming sessions have been centred around several subtasks of a Mandelbrot programming example. The display part of the Mandelbrot example has been provided, so the participants could focus on the algorithmic part of the Mandelbrot algorithm, expressed in the SAC language.

3.2 Programming in S-Net

The S-Net programming examples then took over the SAC solution to be converted into an S-Net-based program by decomposing the program into several boxes, with the individual boxes still implemented in SAC. This task of decomposition is actually a good introduction for the programming work of the industry partners within the **Advance** project, where legacy applications are also transformed into S-Net-based programs, or the algorithms are re-implemented in SAC.

An overview of the programming tasks for the S-Net part of the workshop is given in Figure 3.1.

Task 1
As a warm-up exercise, take a look at `mandelbrot_simple.snet`. This tiny program uses the mandelbrot implemetation of the SaC exercises as one monolithic box. Open `boxes/mandelbrot.sac` and scroll to the bottom to get an idea of how to "S-Netify" existing source code. Take a look at the provided Makefile to find out how the S-Net application is built.

Task 2
Develop an S-Net that models the data-flow of your SaC mandelbrot implementation. Convert your SaC functions into a set of boxes for use with your developed S-Net.

Task 3
Try to exploit concurrency as much possible by identifying independent tasks of your program and arranging these tasks in parallel.

Task 4
Exploit even more concurrency by decomposing the plane into multiple parts and apply the mandelbrot algorithms to each of the parts independently. After processing, merge the results back into one big plane.

Task 5
Explore the MPI capabilities of S-Net. Make use of the placement combinator and try your program of Task 4 on our cluster.

Task 6
Can you come up with a dynamic scheduling of processing tasks? Investigate what impact different scheduling approaches have on the runtime of your program.

Figure 3.1: Summary of the S-Net Programming Exercises

Chapter 4

Conclusion

In this document we provided a summary of the second user community workshop within the **Advance** project, held from September 27^{th} till October 1^{st}, 2010 at the Siberian Branch of the Russian Academy of Sciences in Novosibirsk.

The purpose of the workshop was to broaden the user community for S-Net and SAC the two key technologies for parallel programming used in the **Advance** project. The workshop was characterised by an attendance of mostly graduate students of the Siberian Branch of the Russian Academy of Sciences.

As a start, programming examples were specifically prepared to allow the participants a fast success experience in programming with S-Net and SAC. The participants gained a first impression of the merit of programming with S-Net and/or SAC. In the second part of the practical work the participants were allowed to choose their own projects to be implemented with S-Net and/or SAC. At the end of the week the students gave project presentations of their own examples they have been implementing.

A remarkable highlight of the project presentations were the cases where students achieved better performance with the S-Net/SAC tools than with native PThreads programming.

Bibliography

[1] C. Grelck and A. Shafarenko. Report on S-Net: A Typed Stream Processing Language, Part I: Foundations, Record Types and Networks. Technical report, University of Hertfordshire, Department of Computer Science, Compiler Technology and Computer Architecture Group, Hatfield, England, United Kingdom, 2006.

[2] Clemens Grelck. Shared memory multiprocessor support for functional array processing in SAC. *Journal of Functional Programming*, 15(3):353–401, 2005.

[3] Clemens Grelck and Sven-Bodo Scholz. SAC: A functional array language for efficient multithreaded execution. *International Journal of Parallel Programming*, 34(4):383–427, 2006.

[4] Clemens Grelck, Sven-Bodo Scholz, and Alex Shafarenko. A Gentle Introduction to S-Net: Typed Stream Processing and Declarative Coordination of Asynchronous Components. *Parallel Processing Letters*, 18(2):221–237, 2008.

[5] A. Shafarenko, S.B. Scholz, and C. Grelck. Streaming networks for coordinating data-parallel programs. In I. Virbitskaite and A. Voronkov, editors, *Perspectives of System Informatics, 6th International Andrei Ershov Memorial Conference (PSI'06), Novosibirsk, Russia*, volume 4378 of *Lecture Notes in Computer Science*, pages 441–445. Springer-Verlag, Berlin, Heidelberg, New York, 2007.