Cooperative Scheduling Implementation in an Object-Oriented Paradigm

Vlad Serbanescu *, Frank de Boer, and Mahdi Jaghouri

Leiden Institute of Advanced Computer Science
Centrum Wiskunde and Informatica
{vlad.serbanescu,frank.s.de.boer,jaghouri}@cwi.nl
http://www.cwi.nl

Abstract. The abstract should summarize the contents of the paper and should contain at least 70 and at most 150 words. It should be written using the *abstract* environment.

Keywords: We would like to encourage you to list your keywords within the abstract section

1 Introduction

-high-level description of cooperative scheduling -position with scala and Akka, state of the art - problem statement: how stacks need to be saved in OO, how expensive a thread is

2 Cooperative Scheduling Implementation Schemes

-sequence diagram of what happens in an actor during synchronous and asynchronous method calls.

2.1 Thread Pool Scheme

Each async. call is modeled as a thread competing for a lock

2.2 Optimization through lambda expressions

-particular uses of an executor service.

2.3 Synchronous calls context

- fully asynchronous environment - modeling continuations as labeled functions and converting them to lambda expressions. - allocating more memory in the heap and saving memory on the stack.

* Please note that the LNCS Editorial assumes that all authors have used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

3 Benchmarking the Implementation Schemes

-multiple stack frame problem. - Old Java, vs Erlang vs New Java benchmark, vs possibly Optimized suspended threads.

4 Proof of Concept: Actor-based implementation of Agent-based modelling

-expressive generic agent model -software engineering context - support for functional data types, FLI, type-checking -sequence of events that maximize agent throughput

5 Related Work

- here we want to discuss a bit about the solution in Jan Schaeffer's thesis.

References

1. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)
2. May, P., Ehrlich, H.C., Steinke, T.: ZIB Structure Prediction Pipeline: Composing a Complex Biological Workflow through Web Services. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1148–1158. Springer, Heidelberg (2006)
3. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
4. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: 10th IEEE International Symposium on High Performance Distributed Computing, pp. 181–184. IEEE Press, New York (2001)
5. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. Technical report, Global Grid Forum (2002)
6. National Center for Biotechnology Information, `http://www.ncbi.nlm.nih.gov`

Appendix: Springer-Author Discount

LNCS authors are entitled to a 33.3% discount off all Springer publications. Before placing an order, the author should send an email, giving full details of his or her Springer publication, to `orders-HD-individuals@springer.com` to obtain a so-called token. This token is a number, which must be entered when placing an order via the Internet, in order to obtain the discount.

6 Checklist of Items to be Sent to Volume Editors

Here is a checklist of everything the volume editor requires from you:

- ☐ The final LaTeX source files
- ☐ A final PDF file
- ☐ A copyright form, signed by one author on behalf of all of the authors of the paper.
- ☐ A readme giving the name and email address of the corresponding author.