Using Docker Container for Apache Spark Scale-up Server Scalability

ABSTRACT

This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings. It is an *alternate* style which produces a *tighter-looking* paper and was designed in response to concerns expressed, by authors, over page-budgets. It complements the document *Author's (Alternate) Guide to Preparing ACM SIG Proceedings Using LaTeX2$_\epsilon$ and BibTeX*. This source file has been written with the intention of being compiled under LaTeX2$_\epsilon$ and BibTeX.

The developers have tried to include every imaginable sort of "bells and whistles", such as a subtitle, footnotes on title, subtitle and authors, as well as in the text, and every optional component (e.g. Acknowledgments, Additional Authors, Appendices), not to mention examples of equations, theorems, tables and figures.

To make best use of this sample document, run it through LaTeX and BibTeX, and compare this source code with the printed output produced by the dvi file. A compiled PDF version is available on the web page to help you with the 'look and feel'.

CCS Concepts

•Computer systems organization → **Embedded systems;** *Redundancy;* Robotics; **•Networks** → Network reliability;

Keywords

ACM proceedings; LaTeX; text tagging

1. INTRODUCTION

2. BACKGROUND AND PROBLEM

3. DOCKER

4. CONCURRENT UPDATES FOR LINUX KERNEL

4.1 Case study:reverse mapping

5. IMPLEMENTATION

6. EVALUATION

This section answers the following questions experimentally:

- Does LDU's design matter for applications?
- Why does LDU's scheme scale well?
- What about LDU's read-write ratio?

7. DISCUSSION

8. RELATED WORK

Operating system scalability. [?]In order to improve Linux scalability, researchers have been optimized memory management in Linux by finding and fixing scalability bottlenecks [?] [?]. Shared address spaces in multithreaded applications easily become scalability bottlenecks since kernel operations including `mmap` and `munmap` system calls and `page faults` handling require per-process locks for synchronization. Multithreaded application, for example, can become bottleneck by kernel operations on their shared address space, whose operations are the `mmap` and `munmap` system calls and `page faults`. These operations are synchronized by a single per-process lock. BonsaiVM [?] solved this address space problem by using the RCU; RadixVM [?] created a new VM using refcache and radix tree, which enable `munmap`, `mmap`, and `page fault` on non-overlapping memory regions to scale perfectly. Alternatively, to avoid contention caused by shared address space locking, system programmers change their multithreaded applications to use processes [?].

9. CONCLUSION