Savina – An Actor Benchmark Suite

AGERE! 2014
Monday, October 20, 2014

Shams Imam, Vivek Sarkar
shams@rice.edu, vsarkar@rice.edu
Rice University
Introduction

• Multicore processors are now ubiquitous
• Parallelism is the future of computing
• Actor Model regained popularity
  • Erlang – flagship language
• Many actor libraries out there for various languages
Motivation

• Benchmarks help motivate language implementers to
  • Improve their implementations
  • Calibrate competitive advantages of their approach

• Currently rely on micro-benchmarks
  • Rarely reflect behavior of real world applications

• Need a benchmark suite that goes beyond micro-benchmarks
Goals

• Savina, a benchmark suite for actor-oriented programs
• Cover a wide range of diverse and realistic use-cases
  • Enable apple-to-apples comparisons
• Implementations available as open source
  • Provide implementations of benchmarks in various actor libraries
  • Encourage researchers to contribute their implementations
Outline

• Benchmarks breakdown
• Micro-benchmarks
• Concurrency Benchmarks
• Parallelism Benchmarks
• Experimental Results
• Availability and Summary
Benchmarks Diversity

- Focuses on computationally intensive applications
- Display commonly used parallel patterns
- Covers wide range of domains
  - Common concurrency problems
  - Graph and Tree Traversal
  - Linear Algebra
Benchmarks Breakdown

- 7 Micro-benchmarks
- 8 Classical Concurrency benchmarks
- 14 Parallelism benchmarks
Micro-benchmarks (I)

- Ping Pong
  - Message delivery overhead
- Counting Actor
  - Message passing overhead
- Fork Join (throughput)
  - Messaging throughput
- Fork Join (actor creation)
  - Actor creation and destruction
Micro-benchmarks (II)

- Thread Ring
  - Message sending; Context switching between actors
- Chameneos
  - Contention on mailbox; Many-to-one message passing
- Big
  - Contention on mailbox; Many-to-Many message passing
Concurrency benchmarks (I)

- Concurrent Dictionary
  - Reader-Writer concurrency; Constant-time data structure
- Concurrent Sorted Linked-List
  - Reader-Writer concurrency; Linear-time data structure
- Producer-Consumer with Bounded Buffer
  - Multiple message patterns based on Join calculus
- Dining Philosophers
  - Inter-process communication; Resource allocation
Concurrency benchmarks (II)

- Sleeping Barber
  - Inter-process communication; State synchronization
- Cigarette Smokers
  - Inter-process communication; Deadlock prevention
- Logistic Map Series
  - Synchronous Request-Response with non-interfering transactions
- Bank Transaction
  - Synchronous Request-Response with interfering transactions
Parallelism benchmarks (I)

- All-Pairs Shortest Path
  - Graph exploration; Phased computation
- A-Star Search
  - Graph exploration; Message priority
- NQueens first K solutions
  - Divide-and-conquer style parallelism; Message priority
- Recursive Matrix Multiplication
  - Divide-and-conquer style parallelism; Uniform load
- Quicksort
  - Divide-and-conquer style parallelism; Non-uniform load
Parallelism benchmarks (II)

• Radix Sort
  • Static Pipeline; Message batching

• Filter Bank
  • Static Pipeline; Split-Join Pattern

• Bitonic Sort
  • Static Pipeline; Round-robin message forwarding and reception

• Sieve of Eratosthenes
  • Dynamic Pipeline; Non-uniform load
Parallelism benchmarks (III)

- Unbalanced Cobwebbed Tree
  - Tree exploration; Non-uniform load
- Online Facility Location
  - Dynamic Tree generation and navigation
- Trapezoidal Approximation
  - Master-Worker; Static load-balancing
- Precise Pi Computation
  - Master-Worker; Dynamic load-balancing
- Successive Over-Relaxation
  - 4-point stencil computation
Experimental Results

- 12-core (two hex-cores) 2.8 GHz Intel Westmere SMP node
- Java Hotspot JDK 1.8.0
- Nine Actor libraries:
  - Akka 2.3.2
  - Functional-Java 4.1
  - GPars 1.2.1
  - Habanero-Java library 0.1.3
  - Jetlang 0.2.12
  - Jumi 0.1.196
  - Lift 2.6-M4
  - Scala 2.11.0
  - Scalaz 7.1.0-M6
Counting Micro-benchmark

![Graph showing the average execution time in log scale against the number of increment messages (in millions). The graph compares different algorithms represented by different markers and line styles. The y-axis is on a log scale ranging from $10^{-1}$ to $10^2$. The x-axis represents the number of increment messages ranging from 2 to 20 in millions.]
ForkJoin Creation Micro-benchmark

Average Execution Time (in secs) in log scale

Number of actors created (in millions)

AK, FJ, GP, HA, JL, JU, LI, SC, SZ
Producer-Consumer with Bounded Buffer benchmark

- Buffer size of 6000
- 5000 producer actors each producing up to 1000 messages
- 2000 consumer actors
Filter Bank benchmark

- 8-way join branches
- 300,000 data items and 131,072 columns
Bitonic Sort benchmark

- 32,768 data items
Sieve of Eratosthenes benchmark

- Find primes smaller than 100,000
Related Work

- Cardoso et al at AGERE last year
  - Compare actor and agent languages
  - Focus on micro-benchmarks (Thread Ring, Chameneos, Fibonacci)
- bencherl: Scalability benchmark suite for Erlang applications
- Theron C++ concurrency library: Five actor micro-benchmarks
- nofib suite: Haskell programs
- Computer Language Benchmarks Game:
  - compares over 20 programming languages on a set of 13 micro-benchmarks
Future Work

• Bug fixes and improved implementations
• Java versions of benchmarks
  • Save on pattern matching overheads
• Discover and add diverse benchmarks
• Other runtime implementations
  • Perform inter-language comparisons
• Compare solutions for elegance
Availability

• Implementation available in github
  
  https://github.com/shamsmahmood/savina

• Open source release allows
  
  • Verifying what is actually being tested
  • Porting the benchmarks to other actor languages and runtimes
  • Comparison of solutions for syntax and elegance
  • Analysis of benchmarks to further study impact of different features

• Encourage community to submit solutions
  
  • Improve existing ones
  • Add new libraries or runtimes
Summary

- Introduced Savina, Actor Benchmark Suite
- Described benchmark breakdown
- Open source release
  - Nine actor libraries compared
  - Expect contributions for other libraries
Comments

- Introduced Savina, Actor Benchmark Suite
- Described benchmark breakdown
- Open source release
- Nine actor libraries compared
- Expect contributions for other libraries

import agere.audience.Feedback
Backup-Slides