Java and C Performance Comparison on Palm OS PDA device

- Basic numerical computation
  example: Multidimensional matrix computation
- Memory management
  example: Java primitive type vs. Java Object
- SpecJVM98, SciMark and JkernelMark official benchmarks
- Measure in execution time and memory usage

Zhi-Kai Xin
zxin@columbia.edu
Motivation

- Java is widely adopted in embedded system to develop various applications
  - PDA, wireless phone, and game console
- Java is easy to learn and powerful
  - Rich set of library
  - Platform independent
  - Easy integration
Motivation cont.

- Popular JVM for PDA or Wireless device
  - Sun Microsystems J2ME (KVM)
    - CLDC and MIDP for Palm OS+
  - IBM J9 VM
  - HP ChaiVM
  - iPAQ PersonalJava (Jeoda)

- C uses Cygwin with PRC-TOOL or CodeWarrior for PDA application development
Related works

- Sosnoski [1][2] Java and C/C++ performance comparison and analysis
  - Java object allocation wastes memory space and takes too long
  - But improvement can be made by using primitive types instead of Java object
  - Use array instead of java.util.Vector
  - Avoid creating new object in large software
Related works cont.

- Moreira et al. [3] numerical computation
  - Multidimensional matrix addition and multiplication
  - Java Array incurs overhead of runtime checking.
  - Java multidimensional array is array of arrays (slower indexing)
  - Java outperforms C if Java array runtime checking is disabled
Questions to be answered by end of this project

- Which language has better performance on Palm OS PDA device, Java or C?
- If C is better, how bad is Java’s performance on PDA? Is it acceptable performance?
- What improvements can be made to Java to run better on PDA?
Development Environment

- Sony Clie (Palm OS Device)
  - 33MHz
  - 16MB RAM
- Java Development IDE
  - Java wireless toolkit
  - CodeWarrior
Results and Analysis
Looping

Double Loop Test

Number of Iteration (Figure 5)

Time in Millisecond

Java Un-optimized  Java Optimized  C

CS4995 Embedded System  zxin@columbia.edu
Array Copy

![Array Copy Test Graph](image)

- Java Loop Array Copy
- Java System Array Copy
- C Array Copy

Size of Array (Figure 6)

Time in Millisecond

Zhi-Kai Xin
zxin@columbia.edu

CS4995 Embedded System
Hashing

Hashing Test

Number of hashing (Figure 7)

- Java No Object Pooling
- Java Object Pooling
- C Hashing

Time in millisecond

CS4995 Embedded System

Zhi-Kai Xin
zxin@columbia.edu
String concatenation

String Concatenation Test

Time in millisecond

Number of concatenation (Figure 8)

- Java
- C

Zhi-Kai Xin
zxin@columbia.edu
Factorial

Java Factorial Test

Time in millisecond

Factorial Input (Figure 10)

Recursion While Loop
Conclusion

Java performance can be improved by fine tune Java program such as:

- Avoid recursion
- Avoid having array access within nested loops
- Use object pooling, avoid create new object, thus avoid garbage collect, especially within loops
- Try use Java’s build in methods, avoid re-writing your own routines
Conclusion cont.

- Java has a rich set of APIs for fast development
- Definitely worth to use on PDA device software development
Future works

- Look into Java’s IO and network performance on PDA devices
Source code

The project source code can be found at:

http://www.cs.columbia.edu/~zxin/cs4995-2/final/project
References

