A Quantitative Argument Against Java-specific Processors

John Novitsky
19 August 96
408-864-6182
novitsky@mms.com

Speaker’s Perspective

• Following argument is *weakly suggestive*
  – Actual Java data required to make strong assertion

• Worked for over 11 years at Intel, contributing architectural ideas to 386, 486, Pentium™ processor.
  – Led a group analysing AI languages & applications (LISP, Prolog, Smalltalk) for impact on general purpose computer architecture (1985-1988).
  – Researched prior attempts at interpreted environments, and impact to computer architecture:
    • Database ~ mid 1970s
    • P-Code ~ early 1980s
    • SOAR, SPUR ~ mid-1980s
    • LISP on RISC ~ mid 1980s
Architectural Argument Against Java CPUs

1) Interpreted Languages spend ~80%-90% of time and instruction counts:
   • Touching memory;
   • COMPARing operands;
   • BRANCHing.

2) There have been no dramatic changes in interpreter technology in the last 10 years.

3) Java is an interpreted language.

4) Compared to Java running on conventional architectures, any Java-specific instruction improvements will have a negligible impact on performance.

Summary of 386/Unix Profiling

- Programs Studied:
  - Common LISP ~ 280M instructions
  - Prolog ~ 80M instructions
  - Smalltalk ~ 25M instructions

- Machine Used:
  - 80386-based, Unix System V.x

- C-LISP Results:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>% of Inst. Count</th>
<th>% of Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVE</td>
<td>54-61%</td>
<td>40-42%</td>
</tr>
<tr>
<td>BRANCH</td>
<td>20-22%</td>
<td>39-42%</td>
</tr>
<tr>
<td>COMPARE</td>
<td>7-10%</td>
<td>8-11%</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>3-8%</td>
<td>1-6%</td>
</tr>
<tr>
<td>ARITHMETIC</td>
<td>2-4%</td>
<td>2-4%</td>
</tr>
</tbody>
</table>

...
Summary

- Java CPUs are unlikely to run Java programs appreciably faster than general-purpose CPUs
- Java CPUs are unlikely to be cheaper to make than high-volume CPUs
- Recommended Strategy:
  - Focus Java compiler and interpreter development, targeted at existing high volume platforms, i.e., Windows or Macintosh PCs, Unix-based workstations
  - Develop embedded Java CPUs only for embedded applications where high-volume end-user pull (OEM funding) is demonstrated

References

- "Profiling Machine Instructions on 80386/Unix-based LISP Systems"
  - Novitsky, Yamada, Lenehan, published ~ 1987, Intel Corp, contact K. Sridharan, Intel Corp, 408-765-5694, sri@gomez.sc.intel.com
- Dissertation: Profiling LISP on MIPS
  - Peter Steenkiste, John Hennessy, Stanford University, ~1987
- SOAR, SPUR: Smalltalk, Common-LISP on RISC
  - Patterson, et al, UC-Berkeley, mid 1980's
- P-Code: Pascal interpreter results
  - UCSD, early 1980s