LLVM Inliner Enhancement

Jiangning Liu
Kevin Qin
Background

• Carefully tuned for a large scope of real applications.

• Some missing opportunities for typical computation intensive benchmarks.
Performance Gain and Code Size Bloat for SPEC2000 After Using -inline-threshold=1000

sweet spots
Heuristic Rules

• Performance
  (A) 2X threshold for callee inside loop.
  (B) 2X threshold for callee with const argument.
  (C) 4X threshold for callee inside loop with <= 3BB.

• Code Size
  (D) Reduce threshold to 225 for *cold* callees.
SPEC2000 Performance Gain and Code Size Bloat after applying heuristic rule A+B+C+D (r226173)

* Show trend only! The x axis positions don’t 1:1 match with the names in this chart!
Compile Time

• Loop Info Analysis is expensive
• Fix A->B->C issue
  – Early exit
  – Choose A->B->C rather than B->D

![Diagram of nodes A, B, C, D with connections]

<table>
<thead>
<tr>
<th></th>
<th>Minutes on single x86 core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>155</td>
</tr>
<tr>
<td>Patch without ABC fix</td>
<td>160</td>
</tr>
<tr>
<td>Patch with ABC fix</td>
<td>155</td>
</tr>
</tbody>
</table>

Ilvm bootstrap time (r226173)
Trade-off between performance and codesize for SPEC2000 (r226173)
Current Status

• Patch is under review in community.
Thank you!
Challenges

• Trade-off
  – Performance gain
  – Code size bloat
  – Compile-time slowdown
SPEC2006 Performance Gain and Code Size Bloat after applying heuristic rule A+B+C+D (r226173)