Hybrid Optimizations: Which Optimization to Use?

John Cavazos
Institute for Computing Systems Architecture
School of Informatics
University of Edinburgh, UK
Motivation

- Register Allocation: important
  - Effective use of registers
- Different Algorithms to choose from
  - Graph coloring: possibly expensive
  - Linear scan: not always effective
- Important for Dynamic Compilation
- Which algorithm to apply?
Allocation Cost
Graph Coloring vs Linear Scan

Graph Coloring

Linear Scan
Solution

- **Features** predict which algorithm to use
- Heuristic function controls allocator
  - Reduces **cost** significantly
  - Retains **benefit**
- Successful with simple features
- Applicable to other optimizations
Hybrid Register Allocation

- Source Code
- Compiler
- IR
- Features of code
- Heuristic Controller
- Graph Coloring Register Allocator
- Linear Scan Register Allocator
- Target Code
## Features of Methods

<table>
<thead>
<tr>
<th>Features</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out, In, and Exception Out Edges</td>
<td>Out, in, and exception out edges in CFG (total, avg)</td>
</tr>
<tr>
<td>Live on Entry</td>
<td>Number of edges live on entry and exit (total, min, max)</td>
</tr>
<tr>
<td>Live on Exit</td>
<td></td>
</tr>
<tr>
<td>Insts and Blocks</td>
<td>Number of instructions and blocks in method (total)</td>
</tr>
<tr>
<td>Block size</td>
<td>Size of blocks (max, min, avg)</td>
</tr>
<tr>
<td>Intervals</td>
<td>Number of live intervals (max, total, avg)</td>
</tr>
<tr>
<td>Symbolics</td>
<td>Number of symbolics (total, avg)</td>
</tr>
</tbody>
</table>
Hybrid Register Allocation

- Source Code
- Compiler
- IR
- Heuristic Controller
- Features of code
- Linear Scan Register Allocator
- Graph Coloring Register Allocator
- Target Code
Inducing Heuristic Controller

- For each method generate **raw** training data
  - Features of method
  - Dynamic spills incurred
  - Cost of allocation algorithms
- Process raw data to generate training set
- Leave-one-out cross-validation
- Rule-Induction
- Learning output **heuristic controller**
Labelling Training Instances

- Two factors:
  - Cost of register allocation
  - Spill benefit of different allocators
- Prefer graph coloring
  - If spill benefit above threshold
- Prefer linear scan
  - If graph coloring cost above threshold
  - No spill benefit
Thresholding

1 - (LS Cost / GC Cost) vs GC more expensive

- Linear Scan
- Graph Coloring
- No Instance
- Cost Threshold (0.5)
- Spill Threshold (8192)

LS Spills - GC Spills
GC less spills --->
Motivation for Threshold Technique

- Noise reduction technique
- Simplifies learning
- Removes cases of fine distinction
- Separation by a threshold gap
Experimental Setup

- Jikes RVM
  - JIT / Adaptive compilation
- PowerPC 533 MHz G4, model 7410
- 10 Registers (5 volatiles/5 non-volatiles)
- SPEC JVM benchmarks
- Running Time (NO compile time)
  - Min of 25 runs
- Total Time (WITH compile time)
Benchmark Running Times

Ratio to Linear Scan

GC
HYBRID
Benchmark Total Times

The graph compares the ratio of the total times for different benchmarks to the linear scan times, with two categories: GC and HYBRID. The benchmarks include jack, db, javac, mpegaudio, raytrace, compress, jess, and the geometric mean. The ratio values range from 0 to 1.3, with certain benchmarks and categories highlighted.
Sample Induced Heuristic

\[
\begin{align*}
\text{GC} & \leftarrow \text{avgLiveExitBB} \geq 3.8 \quad \wedge \quad \text{avgVirtRegBB} \geq 13 \\
\text{GC} & \leftarrow \text{avgLiveEntryBB} \geq 4 \quad \wedge \quad \text{avgCFGInEdgesBB} \geq 1.4 \quad \wedge \\
& \quad \text{avgLiveExitBB} \geq 5.5 \quad \wedge \quad \text{numberInsts} < 294 \\
\text{GC} & \leftarrow \text{avgLiveExitBB} \geq 4.3 \quad \wedge \quad \text{maxLiveEntry} \leq 13 \\
\text{GC} & \leftarrow \text{avgLiveExitBB} \geq 3.7 \quad \wedge \quad \text{maxLiveEntry} \geq 9 \quad \wedge \\
& \quad \text{numVirtReg} \geq 895 \quad \wedge \quad \text{maxLiveIntervals} \geq 38 \quad \wedge \\
& \quad \text{maxLiveIntervals} \geq 69 \\
\text{LS} & \leftarrow
\end{align*}
\]
Conclusion

- Hybrid Register Allocation successful
- Preserves Benefit Graph Colloring
- Reduces total (allocation) time
More Information

- COLO (COmpilersThat Learn to Optimise)
  http://www.anc.ed.ac.uk/machine-learning/colo/
- My Website:
  http://homepages.inf.ed.ac.uk/jcavazos/