Register Allocation via Coloring of Chordal Graphs

Fernando Magno Quintao Pereira
Jens Palsberg
ASPLAS 2005

Presented by Daniel Lee
15-745, Spring 2006

Register Allocation

- Compilers traditionally generate code for a specific architecture assuming there is an infinite number of temporaries (registers).
- However, real chips have a finite number of registers.
- Register allocation is the problem of safely mapping temporaries to a finite supply of registers.

Register Allocation is NP Complete

- Shown by Chaitin via reduction from graph coloring.
- Most algorithms for register allocation reduce register allocation to the coloring of interference graphs.
- Despite being NP-complete, there are many reasonably fast approximation algorithms for register allocation, Iterative Register Coloring or whatever voodoo gcc does.

Register Allocation is not always NP Complete!

- For certain classes of graphs, graph coloring is solvable optimally in polynomial time.
- Many interference graphs generated by real programs falls into one or more of these classes (perfect graphs, 1-perfect graphs, chordal graphs).
- Polynomial time graph coloring for perfect graphs is complicated.

Chordal Graphs

- However, graph coloring for chordal graphs is beautifully simple and O(|v| + |e|).
- A chordal graph has the property that every cycle with 4 or more edges has a chord.
- A chord is an edge that connects two vertices in the cycle but is not part of the cycle.
- Programs in strict SSA form have chordal interference graphs, result due to Sebastian Hack.

Greedy Graph Coloring

- Give an arbitrary ordering to your colors.
- While there is a node that has not been colored, assign it the lowest color that has not already been assigned to one of its neighbors.
- That's it. This procedure is O(|e|).

Simplicial Elimination Ordering

- A clique is a fully connected graph.
- A vertex v in graph G is simplicial if its neighborhood in G is a clique.
- A simplicial elimination ordering of G is a bijection s : V (G) -> {1,...,|V|}, such that every vertex vi is a simplicial vertex in the subgraph induced by v1,...,vi.
- Theorem: a graph is chordal iff it possesses a simplicial elimination ordering.

Optimal Chordal Graph Coloring

- Greedy coloring is optimal for chordal graphs if vertices are colored in a simplicial elimination ordering.
- A simplicial elimination ordering can be determined by Maximal Cardinality Search, which runs in O(|e|+|v|).
- The algorithm for MCS is in the paper.

Spilling

- A snazzy graph coloring algorithm does not a register allocator make.
- Even when optimally allocated, some interference graphs will not fit into however many registers your machine has.
- These values must be spilled onto the stack.
- Optimal spilling is NP-complete.

Pereira-Palsberg Register Allocation

- Observation: 95% of interference graphs generated by a large sample of code have chordal interference graphs.
- Even if the graph is not chordal, the greedy algorithm will give a reasonable coloring.
- Result: a non-iterative register allocator based on greedy coloring for chordal graphs and powerful heuristics for spilling and coalescing.

P-P Register Allocation: Phases

- Build interference graph
- Optional: MCS and then pre-spilling.
- MCS
- Greedy Coloring (with an unbounded number of colors)
- Post-spilling (removes extra colors by spilling)
- Coalescing

Pre-spilling

- Calculate maximal cliques.
- Spill vertices at their intersections until the result is k-colorable (so post spilling will never be necessary).
- If the graph is not chordal, this may produce unnecessary spills.

Spilling Heuristics

- Easy to implement: spill the highest colors in the color ordering.
- Better, but harder to implement: spill the colors that are used the least.
- Not addressed in paper: inner-loop considerations.

Coalescing Heuristic

- When there is a move instruction r1 := r2, it is desirable to try to coalesce these into the same register.
- The heuristic used by P-P is to coalesce r1 and r2 if there is a register r3 that is not used by a neighbor of r1 or a neighbor of r2.
- This is unique in that it happens last (in IRC, coalescing happens before spilling).
- Coalescing would cause some chordal graphs to become non-chordal.

Comparisons with IRC

- Iterated register coloring is... iterative. If spilling occurs, the interference graph must be reconstructed and the process repeated.
- (From personal experience) iterative register coloring is a pain to implement.
- P-P requires only one pass.

Experimental Results

- Paper, page 12.
- Over 90% of the interference graphs seen were chordal.
- On average uses fewer registers, spills fewer registers, and finds more opportunities for coalescing.
- Better worst case spilling behavior.
- Variations depending on which spilling heuristic is used.
- Overall, the least-used color heuristic is best.