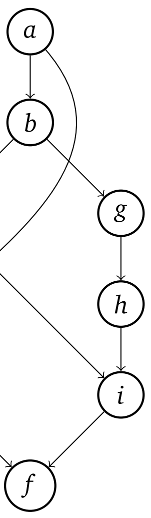
Exam in Optimising Compilers EDA230

January 8, 2009, 8.00 — 13.00

Examinator: Jonas Skeppstedt, tel 0767 888 124



$$\begin{tikzcd}
 & a \arrow[d] \arrow[ddl, bend left] & \\
 & b \arrow[dl] \arrow[dr] & \\
c \arrow[d] \arrow[ddrr] & & g \arrow[d] \\
d \arrow[d] & & h \arrow[d] \\
e \arrow[dr] & & i \arrow[dl] \\
 & f &
\end{tikzcd}$$

Figure 1: Control flow graph.

1. (10p) Explain how the Lengauer-Tarjan algorithm (the $O(N^2)$-version) finds the dominator tree in the control flow graph in Figure 1. For each vertex, your solution should explain:

   - when is the vertex put in a bucket?
   - in which bucket?
   - when is it deleted from the bucket?
   - when does the algorithm find the immediate dominator for the vertex?

   Answer: see book.

2. (10p) Consider again the control flow graph in Figure 1. Suppose there is a use of variable x in each vertex and an assignment to x in vertices a, c, and h. **In vertex a the definition is before the use and in vertices c and h the definition is after the use.**

Translate the program to SSA form. Show the contents of the rename stack and when the stack is pushed and popped. *You do not have to show how you compute the dominance frontiers.*

Answer: see book.

3. (5p) What is the definition of **dominance frontiers**?

   Answer: see book.

4. (5p) What is a **critical edge** in a control flow graph, what is the purpose of removing critical edges, and how is that done?

   Answer: a critical edge is for example (a, d) in Figure 1.fig where a has multiple successors and d has multiple predecessors. Critical edges can be removed by inserting a new and empty vertex between the tail and the head of the edge, and the reason why we would like to do so is that if we we want e.g. insert an expression at the end of a during SSAPRE at a Φ-function in d. That expression would be evaluated also on the execution path (a, b) which makes another expression redundant.

5. (10p) Explain with a small example (either source code or a control flow graph or both) what the fundamental ideas of **global value numbering** are.

   Answer: see book.

6. (10p) Which problem does **SSAPRE** have with optimizing the following loop and how can it be solved? Show the result as a C program.

```
double f(double a, double b, double c)
{
        double  d;

        d = 0.0;

        while (d < c)
                d += a * b;

        return d;
}
```

   Answer: The Φ-function inserted in the loop header is not downsafe (since the while-loop might not be executed at all). This can be overcome by the compiler through rewriting the while-loop as an if-statement and a do-while loop.

```
double f(double a, double b, double c)
{
        double  d;

        d = 0.0;

        if (d < c) {
                t = a * b;
                do
                        d += t;
                while (d < c);
        }

        return d;
}
```

7. (5p) When a node is removed from the interference graph in graph colouring based register allocation, under which circumstance is it certain that there will be an available colour for it when the graph is rebuilt? Why?

   Answer: If the node u was removed during simplify because it had fewer than K neighbours it will find a colour when it is later reinserted into the interference graph, since even if all neighbours are assigned different colours there will be at least one unassigned colour left for u.

8. (5p) In unimodular loop transformations it is desirable that the loop nest is **perfect**. Why?

   Answer: Since otherwise the outer statements must be protected by if-statements in the new loop.