

### Exercise Set 5

19 April 2005\*

**Exercise 5.1** (Appel Ex. 10.1) Perform flow analysis on the following program:

```
1  $m \leftarrow 0$ 
2  $v \leftarrow 0$ 
3 if  $v \geq n$  goto 15
4  $r \leftarrow v$ 
5  $s \leftarrow 0$ 
6 if  $r < n$  goto 9
7  $v \leftarrow v + 1$ 
8 goto 3
9  $x \leftarrow M[r]$ 
10  $s \leftarrow s + x$ 
11 if  $s \leq m$  goto 13
12  $m \leftarrow s$ 
13  $r \leftarrow r + 1$ 
14 goto 6
15 return  $m$ 
```

- Draw the control-flow graph
- Calculate live-in and live-out at each statement.
- Construct the register interference graph.
- What is the smallest  $k$  such that this code can be run on  $k$  registers without spilling? Do not prove it formally, but try to give an example of efficient coloring.

**Exercise 5.2** Compiler (cl6x) produces a warning message while compiling the following program.

```
int uninitialized(void) {
    int c;
    while (c<100)
        c += 7;
    return c;
}
```

Can you explain what part of the compiler is producing this warning? Can you explain the mechanism that leads to its detection?

**Exercise 5.3** Implement saturating subtraction in ANSI C. Then identify the corresponding intrinsic and use it in place of your implementation. Generate assembly from both files using `cl6x`. Compare the output.

**Exercise 5.4** (subproject) Analyze the functions in your project with respect to applicability of inlining. Begin with reading conditions for inlining functions on p. 2-42 [SPRU187, *Optimizing Compiler User Guide*]. Then take the C code of your project and classify each function, whether it can be inlined or not. Focus on computation intensive functions, called many times (you may use a profiler to identify these).

Find functions that you think are suitable for inlining and will benefit the program if inlined (would bring a speed up, without excessive growth of code size). Use the keywords (`inline`) to control inlining. Analyse the generated code to see whether inlining was applied in the places you expect it to be.

If you find a function that should be inlined, but it cannot be inlined due to inlining restrictions (p. 2-42), consider rewriting the code so that it can meet the requirements for inlining.