Computer Science 320: Final Examination
May 17, 2017

You have as much time as you like before the Monday May 22nd 3:00PM ET deadline to answer the following questions. For partial credit, show all work. You may only use your notes, the recommended texts, and the course material available on the Spring 2017 COS 320 website. You may not discuss test content with anyone other than the course instructor/TA. Other than for visiting the Spring 2017 COS 320 website and for making direct contact with the instructor/TA, you may not use electronic devices for any purposes related to this test.

Write out and sign the Honor Code pledge just prior to turning in the test. "I pledge my honor that I have not violated the Honor Code during this examination."

This test is not formatted for your answers. Submit your answers via email to: august@princeton.edu

1) Name a dataflow analysis discussed in class that is *forward*

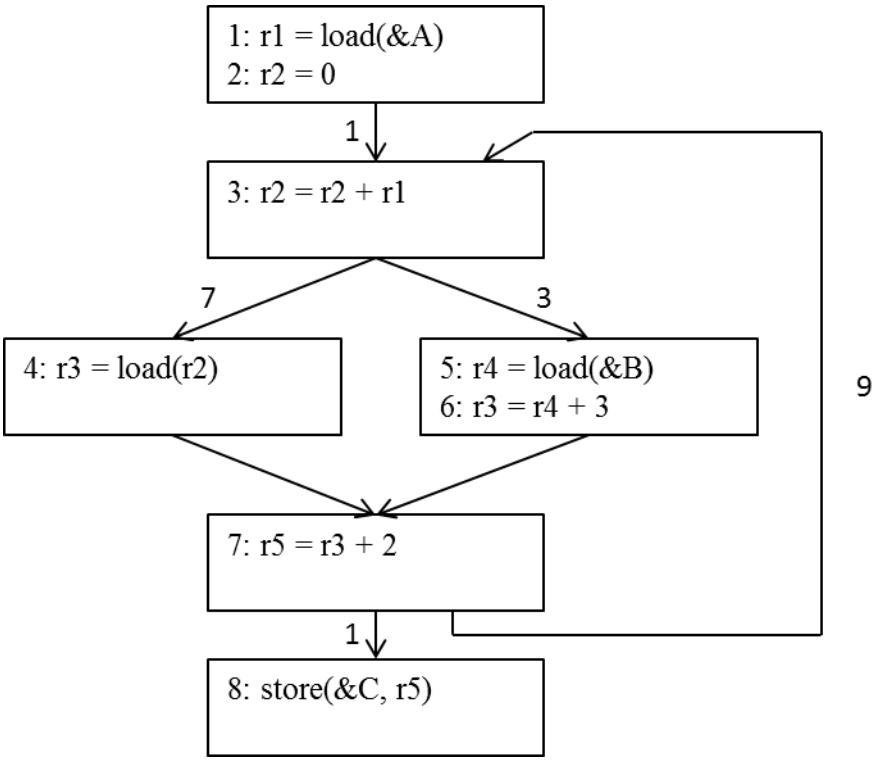
2) Name one advantage that superblocks provide a compiler over traditional basic blocks

3) Name two optimizations that exhibit *constructive interference*, i.e., the application of one optimization enables additional opportunities for the other to be applied.

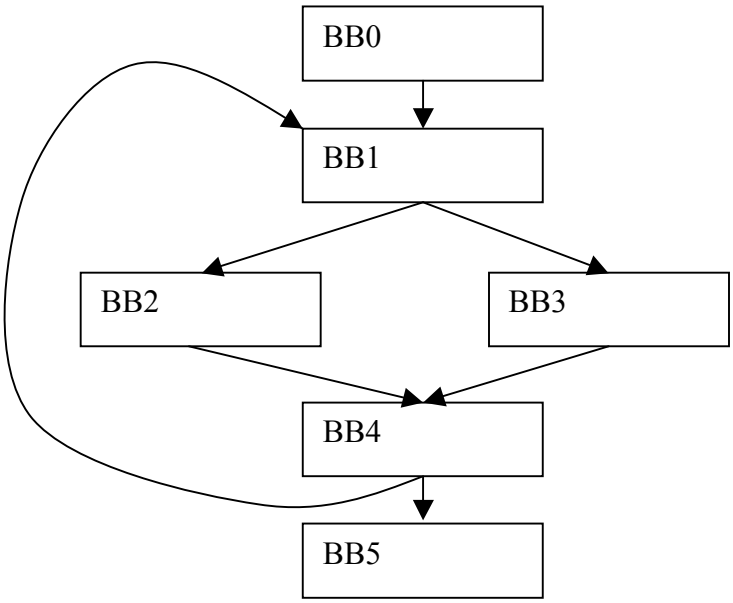
4) Give a reason why it may be *undesirable* to apply common subexpression elimination (CSE) to an instruction that can be *legally* optimized?

5) Compilers often provide warnings if local variables are used without first being defined in a function. Name the iterative dataflow analysis described in class that best provides this information.

6) Consider the following code segment. If *2 physical registers* are available, what is the minimum number of dynamic spills that must occur when virtual registers r1, r2, r3, r4, and r5 are allocated? Assume that &A, &B, and &C are compile time constants and do not require registers. The edges of the control flow graph have been annotated with the profile execution counts. Justify your answer.



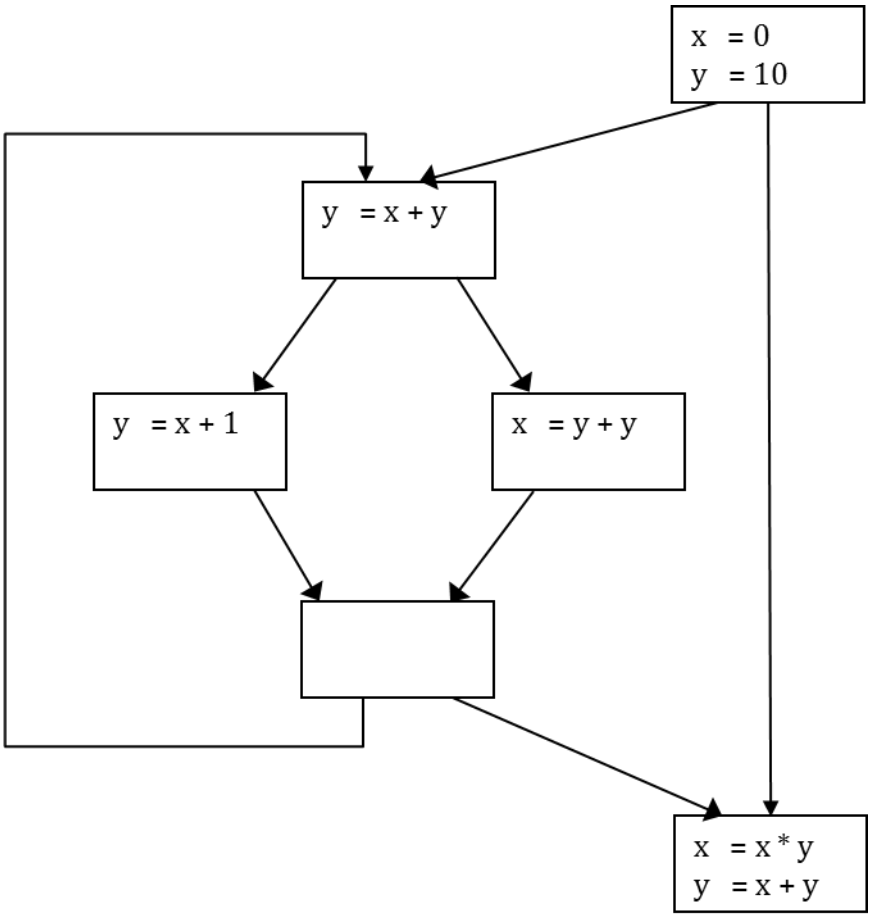
7) In the following control flow graph (CFG), add 4 instructions such that the following conditions are met: each instruction is in a different basic block, each instruction sources the destination register of one of the instructions (possibly itself) but all destination registers must be sourced at least once, at least 3 of the instructions can be removed from the loop by LICM. You may assume any relevant registers are properly initialized.



8) Convert this code to assembly, draw the control flow graph (CFG), draw the dominator tree, and compute the dominance frontier for each node.

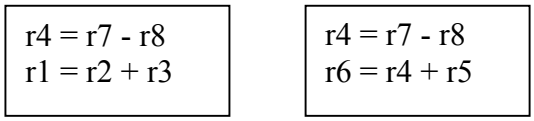
```
do {
  a = load(x)
  if ((a > 0) || (y > 0)) {
        if (a > 10)
           y--;
        else
           x++;
  }
} while (a < 50)
```

9) Convert the following program segment into static single assignment (SSA) form. You should perform the necessary renames and show the Phi nodes. Solving by inspection is fine.



10) Given the following definition of "anticipated": *An expression E is anticipated at a point p if every path from p to Exit contains an instruction that evaluates E and is not preceded on that path by an instruction that might kill E*. The idea is to determine how early one could compute an expression in the program before it actually needs to be used.

So, for example, at the top of the left block, the expressions r2+r3 and r7-r8 are anticipated, but at the top of the right block only r7-r8 is anticipated.



Define the set of dataflow equations to solve for anticipated expressions. You should define GEN, KILL, IN, and OUT.

11) Your boss has tasked you with reverse engineering a competitor's VLIW processor. You have been provided an application and its resulting compiler schedule. Through some preliminary analysis, some of the machine characteristics have been determined, as provided in the table on the left. But they have not yet determined the complete resource usage of each instruction, which is where you come in. (Note: "?" can be between 0 and the total resources the associated type.)

Opcode	Latency	Resources
Add	1	1X, 0Y, 0Z
Mpy	2	?X, ?Y, 2Z
Load	3	1X, ?Y, 1Z
Store	1	?X, 2Y, ?Z

Total resources: 2X, 3Y, 3Z

Application
1: r1 = load (A)
2: r2 = load (B)
3: r3 = load (C)
4: r4 = r1 + r2
5: r5 = load (r4)
6: r6 = r4 + 1
7: r7 = r6 × r5
8: r8 = r3 + r5
9: r9 = r7 × r6
10: store(r4, r8)

Best schedule	
time	instructions
0	1
1	2
2	3
3	
4	4
5	5,6
6	
7	
8	7
9	8
10	10
11	9

a) Draw the data dependence graph for the application labeling each edge with the latency. You can assume that each source operand is read at time 0 and each destination operand is written at the latency. Note, instructions 1, 2 and 3 do not alias with instruction 10.

b) What operations are on the critical path? How many cycles does the application require if there were infinite resources?

c) Suppose the schedule on the right is the best that can be achieved for the application on the processor. What can you conclude about the resource utilization of each operation type? Explain your answer.

d) Suppose you can add 1X resource to the processor or reduce the multiply latency to 1 cycle. Which change will increase the performance of the application the most? Explain your answer.

12) Write a grammar that is in LR(0) but not in LL(1). Prove that your answer is correct.