

Write your answers on another sheet of paper. Homework assignments are to be completed individually. Hand-written submissions are fine, but they must be readable. Due at the beginning of class. Total points: 100, 5% of course grade

1. [10 Points] Exercise 2.2.1 from book. “Consider the context-free grammar

$$S \rightarrow SS + |SS * |a$$

- a) Show how the string “aa+a*” can be generated by this grammar.
b) Construct a parse tree for this string.
c) What language does this grammar generate? Justify your answer.”
2. [10 Points] Garbage Collection. Please write a one paragraph answer to the following question: How does type safety or lack thereof affect GC?
3. [20 Points] Instruction Scheduling. The loop shown below

```
for (i=1000; i>0; i--) {  
    x[i] = x[i] + s;  
}
```

can be converted to assembly as follows:

```
// body of the loop  
Loop:  
    L.D      F0, 0(R1)      // F0 = &R1  
    ADD.D    F4, F0, F2     // F4 = F0 + F2  
    S.D      F4, 0(R1)  
    DADDI    R3, R3, #-1    // R3 = R3-1  
    DADDI    R1, R1, #-8    // R1 = R1-8  
    BNE     R3, R2, Loop
```

ADD.D is the instruction for adding doubles and DADDI is the instruction for adding integers. Use the following set of latencies between different instructions when the second instruction has a flow dependence on the first instruction. If no latency is given for a pair of instructions, then assume it is one. For example, if the result of one ADD.D instruction is used by a following ADD.D instruction, the following ADD.D instruction will have to wait 3 cycles to begin execution. If only one instruction is scheduled in between the two, there will still be two cycles of interlock.

first instruction	second instruction	
	ADD.D	S.D
ADD.D	4	3
L.D	2	1

The body of the loop given above requires 9 cycles to execute (plus any cycles it takes to finish the BNE instruction). Use list scheduling with the following priorities to find a schedule that only requires 8 cycles:

- (a) Avoid stalls with previously scheduled instructions.
- (b) Does the instruction interlock with any immediate successors?
- (c) How many successors does the instruction have?
- (d) Is the instruction on the critical path?

Show the DAG that you use to perform the scheduling. Assume that the BNE instruction must occur last.

Extra credit: What small edit would enable scheduling the store after the add of -8 to R1 and result in a schedule that is only 7 cycles?

4. [20 Points] Do Exercise 9.3.5 in book. “Suppose the set F of functions for a framework are all of gen-kill form. That is, the domain V is the power set of some set, and $f(x) = G \cup (x - K)$ for some sets G and K . Prove that if the meet operator is either (a) union or (b) intersection, then the framework is distributive.”
5. [20 Points] Activity analysis is a data-flow analysis needed in the context of Automatic Differentiation. One piece of activity analysis is a forward data-flow analysis called vary analysis. The goal of vary analysis is to determine the set of variables in a procedure that depend on a specified subset of independent variables at various points in the program. For example, in the below program, if x is the independent variable of interest, then $OUT(s1) = \{a, x\}$, $OUT(s2) = \{a, b, x\}$, $OUT(s3) = \{b, x\}$, and $OUT(s4) = \{b, x, y\}$.

```

// independent = {x}
s1:  a = x + 3;
s2:  b = a *2;
s3:  a = c;
s4:  y = a + b;

```

Vary analysis has the following specification:

- must or may: may
- direction: forward
- meet: union
- data-flow values: sets of variables
- initial value: empty set
- $OUT[\text{entry}] =$ the set of independent variables
- transfer function: $f(X) = GEN \cup (X - KILL)$

GEN is defined as the set of variables being defined in the statement, if a variable in X is being used in the statement. $KILL$ is the set of variables being defined in the statement.

Perform vary analysis on the following procedure using iterative data-flow analysis. For each statement in the loop, show the OUT data-flow set. How many iterations of iterative data-flow analysis are required for convergence assuming that the statements are visited in the order they are listed in the program?

```
// independent = { a }
  for ( i=0; i<N; i++ ) {
    e = a + b + c + d;
    d = c + b;
    c = b * a;
    b = a - 3;
  }
```

6. [10 Points] For the following program, draw the control-flow graph and perform copy propagation followed by dead-code elimination.

```
S1:    j = 0
S2:    y = read()
S3:    x = y
S4:    z = x
S5:    L1:
S6:    if j>10 goto L2
S7:    z = x
S8:    j = j + 1
S9:    goto L1
S10:   L2:
S11:   print x, y, z
```

7. [10 Points] Show the following sets for each block in Figure 9.33 in the book: `anticipated_out`, `anticipated_in`, `available_in`, `available_out`, `earliest`, `postponable_in`, `postponable_out`, `postponable_in`, `latest`, `used_in`, and `used_out`. A grid format as was done in class would be preferable.