CS 2210  Compiler Construction

Sample Final Exam

April 25, 2018

Student Name : 
_________________________
1. True or False questions (similar in flavor to midterm)

2. Short answer questions (similar in flavor to midterm). Important concepts:
   a. Comparisons between different typing systems
   b. Two aspects of semantic analysis (SDTS and SDD)
   c. Semantic analysis on top-down / bottom-up parsers, L-attribute grammars
   d. Different IR representations and how they work
   e. Runtime environment and runtime code
   f. Different types of memory management
   g. Organization of the stack and calling convention
   h. Buffer overflow attacks and their prevention
   i. Comparison of different garbage collection schemes
   j. Layout-related optimizations
   k. Local and global optimizations
   l. Control flow analysis and control flow graphs
   m. Data flow analysis framework
   n. Comparison of different register allocators

3. Tracing / Coding

   (1) Given the following procedure, please compute the outcome according to different parameter passing methods: (a) call-by-value (b) call-by-reference (c) call-by-name
   
   int x = 1, y = 2;
   function F1 (parameter a, b) {
     a = x + a;
     x = a + b;
     y = a - b;
     b = y + b;
   }
   main() {
     F1( x, y );
     print("x=",x,"y=",y); /* print the value of x and y */
   }

   (2) (6+6=12 points) Use both (1) **two-pass approach** and (2) **backpatching** to translate the following C-like statement

   S \rightarrow DOIF (E1, E2)
   S3; S4; S5
   UNTIL (E6)

   Its semantics are the same as follows

   
   do {
     if (E1) S3;
     else if (E2) S4;
     else S5;
   } until (E6);

   Assume we use LR parsing based syntax-directed translation.
4. Consider the following 3-address code.

\[
\begin{align*}
a &= a + 1; \\
\text{if } (b > 0) \text{ goto L2;}
\end{align*}
\]

L1:
\[
\begin{align*}
a &= a + b; \\
c &= b + 1; \\
d &= b + 1; \\
goto \text{L3};
\end{align*}
\]

L2:
\[
\begin{align*}
e &= e + b; \\
\text{if } (b < a) \text{ goto L1;}
\end{align*}
\]

L3:
\[
\begin{align*}
b &= b + 1; \\
b &= b \times c; \\
b &= b \times d;
\end{align*}
\]

(1) Break the code into basic blocks and draw the control flow graph;
(2) List what the four components of the dataflow analysis framework are for global variable liveness analysis.
(3) Perform global liveness analysis, and annotate the flow graph with the \textbf{in} and \textbf{out} sets at each point. \textbf{Perform the analysis at the statement level}.
(4) List what the four components of the dataflow analysis framework are for global constant propagation.
(5) Perform global constant propagation, and annotate the flow graph with the \textbf{in} and \textbf{out} sets at each point. \textbf{Perform the analysis at the statement level}.
(Assume that a = 1, b = 2 at the beginning of program and c, d, e are undefined)

5. Given the following inference rules:

\[
\frac{i \text{ is an integer}}{i : \text{int}} \quad \text{and} \quad \frac{e_0 : \text{int}}{e_1 : \text{int}} \quad \frac{e_0 + e_1 : \text{int}}
\]

Prove the following:

\[
\frac{3 \text{ is an integer}}{4 \text{ is an integer}} \quad \frac{5 \text{ is an integer}}{3 + 4 + 5 : \text{int}}
\]

* Be prepared to apply other inference rules involving type environments and variable definitions
6. Given the following live ranges:
   (1) Construct the register interference graph
   (2) Use Chaitin’s graph coloring algorithm to allocate with 4 registers
   (Show two things: a) Stack after all variables have been pushed and b) Final mapping to registers)