Answers to

COP5621 Compiler Construction Exam 4 - Spring 2007

Name: ________________________________________ (Please print)

Put the answers on these sheets. Use additional sheets when necessary. You can collect 100 points in total for this exam.

1. Which of the following optimizations is considered a peephole optimization? (mark one)(4 points)
   (a) Common-subexpression elimination
   (b) Code motion
   (c) Branch chaining
   (d) Register assignment

2. Subroutine frames (activation records) manage a procedure’s local data. For a typical programming language implementation, who deallocates the frame? (mark one)(4 points)
   (a) Caller
   (b) Callee
   (c) Caller and callee
   (d) None of the above

3. List three local optimizations of your choice and describe what they optimize. (9 points)

   constant folding
   constant combining
   strength reduction
   constant propagation
   common subexpression elimination
   backward copy propagation
4. Consider the following program:

```plaintext
program P()
    var p : integer;
    procedure Q(k : integer)
        begin
            R(k, p)
        end
    procedure R(i : integer, j : integer);
        var n : integer
        procedure T(i : integer)
            begin
                ... (* body of T *)
            end;
        procedure S()
            var m : integer
            begin
                T(n)
            end;
        begin
            S()
        end;
    begin
        Q(p)
    end
```

(a) Program P calls Q, Q in turn calls R, R in turn calls S, and S in turn calls T. Draw the resulting stack layout with activation records. Show the arguments and local variables in each record and draw the access links. (10 points)
(b) Which variables are visible (in scope) in the body of T and how many access links must be traversed to reach the nonlocal data? (6 points)

<table>
<thead>
<tr>
<th>Var</th>
<th>visible (Y/N)</th>
<th>#links</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>Y</td>
<td>0</td>
</tr>
<tr>
<td>j</td>
<td>Y</td>
<td>1</td>
</tr>
<tr>
<td>k</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>m</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>Y</td>
<td>1</td>
</tr>
<tr>
<td>p</td>
<td>Y</td>
<td>2</td>
</tr>
</tbody>
</table>

5. Partition the following fragment of three-address code into basic blocks and construct the CFG. (9 points)

```
if n>0 goto L1
goto L3
L1:  i := 0
     f := 1
L2:  i := i + 1
     f := f * i
     if i<n goto L2
goto L4
L3:  f := 0
L4:  halt
```
6. Apply global constant propagation to the following CFG. (8 points)

Diagram:

- \( z := 0 \)
- \( y := 5 \)

- \( x := 0 \)
- \( z := 3 \)

- \( y := 0 \)
- \( y := y + x \)
- \( y := y - z \)

- \( z := y + z \)
7. Apply register allocation and assignment using graph coloring to the following CFG:

```
\[
\begin{array}{c}
x := \text{read}() \\
t := \text{read}()
\end{array}
\]

\[
\begin{array}{c}
t > 0 \ ?
\end{array}
\]

\[
\begin{array}{c}
x := x + 1 \\
y := 0
\end{array}
\]

\[
\begin{array}{c}
y := t \\
t := 0
\end{array}
\]

\[
\begin{array}{c}
p := t + y \\
p := p + x
\end{array}
\]

x, y, t are not live

Hint: show the live ranges of the variables in the CFG, assuming that x, y, and t are no longer live at the exit from the CFG. Draw the register-interference graph (conflict graph) for the variables and determine the minimum number of colors to color the graph. (12 points)

```
\[
\begin{array}{c}
p \xrightarrow{} x \xrightarrow{} t \\
\end{array}
\]

```

Then registers

For example: 

\[
\begin{array}{c}
x \rightarrow R_0 \\
y \rightarrow R_1 \\
t \rightarrow R_2 \\
p \rightarrow R_1
\end{array}
\]

```
8. Consider the following CFG:

(a) Draw the dominator tree of the CFG. (8 points)
(b) Identify the natural loops. (5 points)
(c) Is the CFG reducible? Explain why or why not. (5 points)

a)

\[ \begin{align*}
1 & \quad 2 \\
\quad & \quad 3 \\
\quad & \quad 4 \\
\quad & \quad 5 \\
& \quad 6 \\
\end{align*} \]

b) Only one natural loop defined by nodes \( 1 \) and \( 6 \)

c) No, nodes \( 6 \) and \( 5 \) cannot form a cycle that can be reduced.
9. Give the data-flow equations for reaching definitions (gen, kill, in and out sets) as described in the book and illustrated in class for the four programming constructs (assignment, statement composition, if-then-else, and do-while). (10 points)

See Chapter 9 (old 10) lecture notes 11 to 14
10. Consider the following program:

\[
\begin{align*}
&\text{if } n > 0 \text{ then} \\
&\quad d_1: \quad i := 0; \\
&\quad d_2: \quad f := 1; \\
&\quad \quad \text{do} \\
&\quad \quad d_3: \quad i := i + 1; \\
&\quad \quad d_4: \quad f := f \cdot i \\
&\quad \quad \quad \text{while } i < n; \\
&\quad \text{else} \\
&\quad d_5: \quad f := 0; \\
&\end{align*}
\]

end if

Annotate the syntax tree of the program with \textit{in}, \textit{gen}, \textit{kill}, and \textit{out} bit-vectors:

Hint: compute the \textit{gen} and \textit{kill} vectors bottom-up first, i.e. start at the leaves. Then compute the \textit{in} and \textit{out} vectors in a top-down, left-to-right traversal. (\textit{in} is inherited, while \textit{out} is synthesized.) (10 points)