COMP 621 Analyses and Transformations - Assignment 3
Interprocedural Analysis, Points-to Analyses and Special Topics
Due Date: December 5th, 2016

1 Q1: Intra- and Inter-procedural Analysis (25 pts)

Consider the following program, with the usual C semantics (pass-by-value function calls, etc.):

```
01: int x,y;
02: main()                  16: void foo(int u)
03: { int i,j;              17: { u = u * 2;            28: void bar(int b)
04:   y = 1;                18:   x = u + 1;            29: { int z;
05:   x = y + 2;            19:   /*-- Point D --*/     30:   y = b + 1;
06:   i = x + 3;            20:   leaf(u);              31:   if (x > 0)
07:   /*-- Point A --*/     21: }                       32:     foo(b);
08:   foo(i);                                           33:   else
09:   j = y + 4;            22: void leaf(int c)        34:     { z = b+2;
10:   /*-- Point B --*/     23: { int z;                35:       foo(z);
11:   x = 2;                24:   z = 2;                36:     }
12:   bar(i);               25:   write(c+z);           37:   /*--Point F --*/
13:   j = 2*x;              26:   /* -- Point E --*/    38: }
14:   /*-- Point C --*/     27: }
15: }
```

For the above program, give the constant propagation information at the labelled program points (A through F), knowing the analysis both propagates constant values when it is safe to do so and evaluates constant expressions. Use the following different strategies to estimate the effects of procedure calls. You only need to give the results for the program points mentioned in each section below. Give your answer for constant values as a set of variable-value pairs (ex: {(x,1),(y,2)}) for each program point.

a) Conservative Approach: Assume nothing is known about the called procedures and provide the information at program points A, B and C. Clearly state what assumptions you are making for your conservative approach.

b) Summary Approach: Prepare a summary of each procedure as to which variables can be potentially modified by any call to the procedure (including transitive calls) as a side-effect. For each function, provide the set of variables that can potentially be modified (ex: {x,y}). Use this summary information to get sharper constant information at program points A, B and C.

c) Full-blown Approach: (1) Precisely estimate the effect of a procedure call, by visiting and analyzing the body of the called function for each call. Provide the constant information at program points A, B, C, D and E. You should tag each piece of dataflow information with the calling path. For example, supposing 'x' has value '1' in 'foo' at point D when called from main at line 8, the answer would be {('main-08/foo', x, 1)}.

2 Q2: Points-to Analysis (25 pts)

Consider the following program:

```
void foo(int x, int y)
{ int z, *p, *q, **pp, **qq;
  z = x + y;
  if (z > 10)
    { p = &x;
      q = &z;
      pp = &p;
      /* --- POINT A --- */
    }
  else
    { p = &y;
      q = &z;
      pp = &p;
      /* --- POINT B ---*/
     }
  qq = pp;
  /* --- POINT C --- */
  *qq = q;
  /* --- POINT D --- */
  S: *q = 8;
  write(x);
  write(y);
  write(z);
}
```

(a) What does this program write for the call foo(1,2)?

(b) Give the flow-sensitive points-to sets that would be computed at the named points (Points A through D). For example, if p definitely points-to the memory location of x, you would write $p \rightarrow \&x$. Also, if q may points-to the memory location of y, you would write $q \rightarrow ?\&y$.

(c) Using the points-to sets from (b), what program transformation, if any, could be performed at program point S?

(d) Show the points-to graph that would be computed by a flow-insensitive points-to analysis for function foo.

(e) Using the points-to graph from (d), what program transformation, if any, could be performed at program point S?

(f) Is the flow-sensitivity necessary to determine whether a program transformation is possible (or not) at program point S? If not, suggest a different statement for program point S that shows an example in which the flow-sensitivity is actually necessary.

3 Q3: Special Topics (20 pts)

Answer any long question from a special top presentation (other than your presentation):

(Remi) How can a dynamic optimizer like Dynamo handle synchronous and asynchronous signals?

(Hongji) The benchmark results of On-Stack-Replacement (OSR) for promotion and deferred compilation are not very interesting, but the technique is still implemented in modern VMs (e.g. Google V8 Javascript engine). Propose another application of OSR (except from deferred compilation and promotion).

(Xiru) What are Virtual Registers?

(Jake) Strictness analysis attempts to determine which paramaters a function will always evaluate. How can strictness analysis be used to optimize a function?

(Steven) How can we improve kind analysis? What are the implications of the differences in kind analyses between functions and scripts?

(Shruti) Part 1: Sometimes synchronized() blocks cannot represent all monitorenter and monitorexit instructions. Which other solutions are possible?

Part 2: How are loops with multiple entry points in control flow subgraphs resolved in Dava?

(Mathieu) What other decisions can be based on the information provided by clone analysis?

(Stefan) How would you go about garbage-collection without any tags?

(Adrian) Discuss how you would implement the different evolutionary changes (sexual recombination, mutation gene duplication/deletion) on the vector of parameters (Slides p9) for a given population.

4 Q4: Your Special Topic (30 pts)

(a) State your special topic question.

(b) Briefly justify why you chose that question.

(c) Give the answer that you are expecting, along with explanations to help the Teaching Assistant. Give the complete answer if you gave one option, or an outline for each option if you gave several options. Your own grade for this question will be influenced by how easy it is to grade other student's answers given yours. A complete, precise answer with highlighted important pedagogical points make it easier to evaluate other student's answers.