CS6843 Program Analysis at IIT Madras

EndSem May 5, 2015

Duration: 120 minutes Number of questions: 6 **compulsory** questions

1. Guess the English word for "fear of Friday the 13th". It ends in ...*phobia*. [1]

2. The final points-to information computed by **Andersen's analysis** for a program P is as below.

   a → {x}; x → {y, p}; b → {y}; y → {a, y, p}; p → {a, y, p}; q → {b}; m → {p}

   Program P contains the following address-of statements: a = &x; b = &y; p = &a; q = &b; m = &p; All the remaining statements in P are either load or store type. Find and write those. Do not write how you got the missing statements. [4]

3.
   a. How would you improve **shape analysis** if each entry in the D and I matrices is an integer rather than a boolean? [2]
   b. Explain the proposal above with a non-trivial example. [2]
   c. Discuss whether your approach proposed above has any trade-offs. [1]

4. An analysis is allowed to use a single bit to check if files are opened before they are read. Assume that the programs contain only *fopen()* and *fscanf()* functions relevant to the analysis.
   a. What kind of programs / scenarios can such an analysis handle? [1]
   b. Give an example program / scenario where the analysis fails to precisely answer if a file is opened before being read. [2]
   c. How would you solve the problem in (b) above by modifying the analysis? Make your answer as precise and complete as possible. [2]

5.
   a. Design an analysis to check that each *lock()* has a corresponding *unlock()* in the program, and each *unlock()* has a corresponding *lock()*. The functions do not have any arguments. [2]
   b. Give an example program where a single *lock()* may have multiple corresponding *unlock()*s. [1]
   c. Modify your analysis to support *lock(x)* and *unlock(x)* where *x* is a lock variable. [1]
   d. Modify your analysis in (c) to disallow *barrier()* within any lock-unlock region. [1]
   e. Analyze the complexity of your analysis in (d). If it already isn't, improve your analysis in (d) such that its complexity is *O(1)* per barrier? [1]

6.
   a. For the following program, find out if there exists any array accesss beyond allowed limits, using polyhedral model. That is, formulate it as a **system of linear inequations**. [2]
   b. You don't need to solve the equations, but mention how would you interpret the solution provided by an LP solver. Make your answer as complete as you can without writing a blog. [2]

```
void main(int argc, char *argv[]) {
        int size = atoi(argv[1]);          // convert to integer.
        int N = atoi(argv[2]);
        int *a = malloc(size);
        ...                                // some code that need not be analyzed.
        for (int i = 0; i < N; ++i) {
                a[3 * i – 6] = a[5];
        }
}
```