CMSC 631 – Program Analysis and Understanding
Fall 2003

About this Class

- Topic: Analyzing and understanding software
- Three main focus areas:
  - Static analysis
    - Automatic reasoning about source code
  - Formal systems and notations
    - Vocabulary for talking about programs
  - Programming language features
    - Affects programs and how we reason about them

About Me

- Office: 4129 AVW
- E-mail: jfoster at cs.umd.edu
- Office hours: Tu 10am-11am, Fr 11am-12pm

Prerequisite

- CMSC 430 or equivalent compiler class
  - Ideas we will use in this class:
    - Parse trees/abstract syntax trees
    - BNF notation for grammars
    - Type checking (usually not much covered in compilers class)
    - Data flow analysis (sometimes not covered in compilers class)
    - Tools like yacc and lex may be useful for your project
  - We won't use most of the other material
    - So even if you haven't taken compilers class, you may be OK
    - Talk to me if you're not sure

Expectations: Readings

- Will read 1-2 papers for each class
  - Typically, papers available on the web
    - Otherwise will hand out photocopies in class
- *Must* participate in brief discussion on class wiki
  - http://corundum.cs.umd.edu:8000/CMSC631
  - Post a few sentences to a paragraph or two on
    - Main contributions/ideas in paper
    - Ideas that were unclear – what do I need to cover in class?
    - Relationship to other papers we've read
    - etc...

Expectations: Readings (cont'd)

- Must post comments by *noon* on day of class
  - First post! can just put up a summary
  - Later posts need to take earlier posts into account
  - Posting earlier is less work
- 20% of grade will be on class participation, including comments on wiki
  - Includes talk on selected paper in 2nd half of course

Expectations: Project

- Class goal: Teach you how to do research
  - So you have to do research as part of the class
- Substantial research project (40% of grade)
  - Any topic vaguely related to the class acceptable
    - Will post some suggestions for projects later on
    - May also be able to share project with other class
  - Completed in groups of size 1 or 2
  - Turn in project write-up at end of semester
  - Give short (15-20 minute) presentation in class at end of semester

Expectations: Exam

- Final exam and/or midterm (40% of grade)
  - Will be some written assignments early on to prepare
  - List of material covered on exams later on

Academic Dishonesty

- Don't do it

Software Chat

http://savoir.cs.umd.edu:8668

- Weekly meeting about programming languages, software engineering, and software systems
- Fridays at 3pm in CSIC 3120 this fall
  - Starting September 12th
- Topics include
  - Current research in the department
  - Practice talks
  - Interesting recent papers

Administrivia

- No class on Thursday, September 4

CMSC 631 – Program Analysis and Understanding Fall 2003

20 Ideas and Applications in Program Analysis in 40 Minutes

Abstract Interpretation

- Rice's Theorem: Any non-trivial property of programs is undecidable
  - Uh-oh! We can't do anything. So much for this course...
- Need to make some kind of approximation
  - Abstract the behavior of the program
  - ...and then analyze the abstraction
- Seminal papers: Cousot and Cousot, 1977, 1979

Example

e ::= n | e + e

$$\alpha(n) = \begin{cases} - & n < 0 \\ 0 & n = 0 \\ + & n > 0 \end{cases}$$

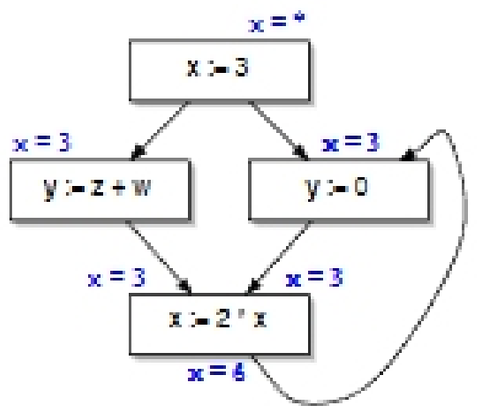
+	-	0	+
-	-	-	?
0	-	0	+
+	?	+	+

- Notice the need for ? value
  - Arises because of the abstraction

Dataflow Analysis

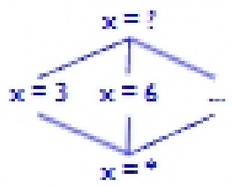
- Classic style of program analysis
- Used in optimizing compilers
  - Constant propagation
  - Common sub-expression eliminating
  - Loop unrolling and code motion
  - etc.
- Efficiently implementable
  - At least, interprocedurally (within a single proc.)
  - Use bit-vectors, fixpoint computation

Control-Flow Graph



Lattices and Termination

- Dataflow facts form a lattice



- Each statement has a transformation function
  - Out(S) = Gen(S) U (In(S) - Kill(S))
- Terminates because
  - Finite height lattice
  - Monotone transformation functions

Static Single Assignment Form

- Transform CFG so each use has a single defn