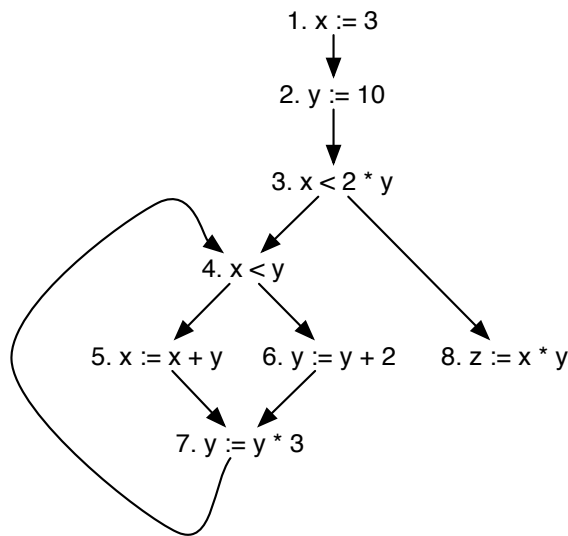


CMSC 631, Spring 2006
Homework 2
Due Tuesday, February 21, in class

1. For the following control flow graph
 - (a) Draw the dominator tree
 - (b) List the dominance frontiers of each node
 - (c) Put the control-flow graph in SSA form (you can eyeball this instead of running the algorithm by hand)



2. In class we gave a definition that α and γ form a Galois connection if

- α and γ are monotonic
- $S \leq \gamma(\alpha(S))$ for any concrete element S .
- $\alpha(\gamma(A)) \leq A$ for any abstract element A .

(Actually, in class the last inequality was an equality, which it usually is in practice; in that case, α and γ would form a Galois *insertion*.)

Similarly, we say that two (arbitrary) functions α and γ form an *adjunction* if

$$S \leq \gamma(A) \text{ iff } \alpha(S) \leq A \text{ for all concrete } S, \text{ abstract } A$$

Show that α and γ form a Galois connection if and only if they form an adjunction.

3. Consider the factorial function:

```
fun fact n = if (n == 0) then 1 else n*(fact (n-1))
```

We wish to use abstract interpretation to prove that, given a non-negative integer as input, this function always produces a positive integer as output.

- (a) Suppose we use as our abstract domain the rule of signs, so that numbers are represented as one of $A = \{+, -, 0, \perp, \top\}$. We want to compute a function $fact' : A \rightarrow A$ in our abstract domain that soundly models $fact$. Ultimately, we would like to show that $fact'(+) = fact'(0) = +$. Since $fact$ is a recursive function, we need to do this by computing a fixpoint. We will compute a series $fact_0, fact_1, fact_2, \dots$ of successively more accurate models of $fact$ until we reach a fixpoint.

Let $fact_0(x) = \perp$ for all $x \in A$. (**Note:** We are starting with \perp for this problem, and we will use join instead of meet.) Written out in a tabular form, we have

x	\perp	\top	$+$	$-$	0
$fact_0(x)$	\perp	\perp	\perp	\perp	\perp

This is our first (rather poor) approximation. To compute the next approximation, $fact_1$, for each possible input value of x , we will use abstract interpretation to evaluate $fact$. When we see the recursive call $fact(n-1)$, we use our previous approximation $fact_0$. Formally, we compute a series with

$$fact_{i+1}(x) = AEval(if (n == 0) then 1 else n * (fact_i (n - 1)))$$

For example, using this formula to compute $fact_1$, we get

x	\perp	\top	$+$	$-$	0
$fact_1(x)$	\perp	$+$	\perp	\perp	$+$

(Remember that all operations, including the test at the if , are strict in \perp , e.g., $x * \perp = \perp$.) Continue computing successive approximations of $fact$, writing each $fact_i$ in tabular form, until you reach a fixpoint $fact'$. Is it the case that $fact'(+) = +$? What went wrong?

- (b) Construct a new abstract domain A' for which we can show that $fact$ is positive given non-negative input. Draw a picture showing the lattice structure for A' , and describe (briefly) operations in your new domain (if they are obvious, you can just say that you use the obvious abstract operations). Finally, compute a fixpoint approximation (i.e., give a sequence of tables, as above) for $fact$ in your new abstract domain showing that $fact$ is positive given non-negative input. **Note:** To get an answer to this problem, $AEval()$ must treat if as precisely as possible, i.e., it must know that the test $0 == 0$ is always true, and it must know that n is zero on the true branch of the test $n == 0$ and non-zero on the false branch.