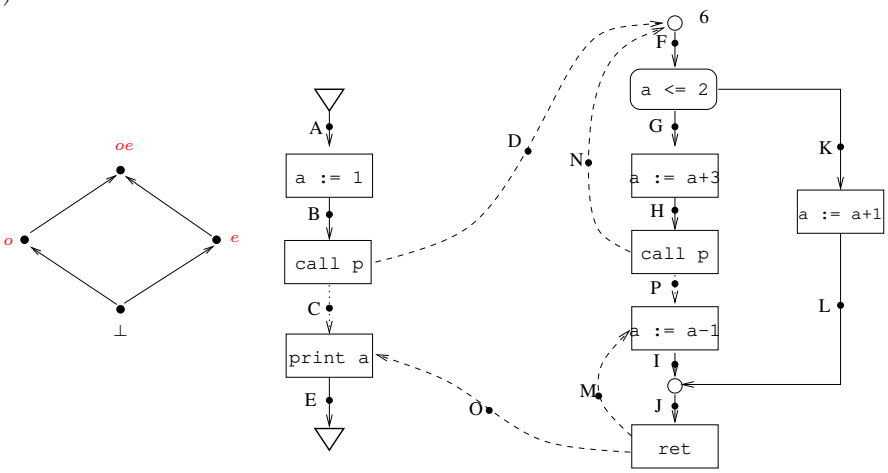
Program Analysis and Verification
Assignment 3
Inter-procedural analysis
Due date: Tue 19 Oct 2021.

Problem 1. Consider the data-flow equations Eq (1) for the join over valid and complete paths (the $\phi_{r_p,N}$'s), for an underlying analysis $\mathcal{A} = (D, \leq, f_{MN}, d_0)$ and program P. Show that if we consider the lattice of *all* functions (i.e. $D \to D$), ordered under the usual pointwise extension of $\leq$, the function $\bar{F}$ induced by these equations *may not* be monotonic. (*Hint*: you can consider the available expression analysis example done in class.) (0.5 pages,10 *marks*)

Problem 2. Consider the program below. Consider a parity analysis for the value of the variable a, using the lattice with elements $\{\bot, o, e, oe\}$ with the usual ordering as shown below. (2 pages, 20 *marks*)

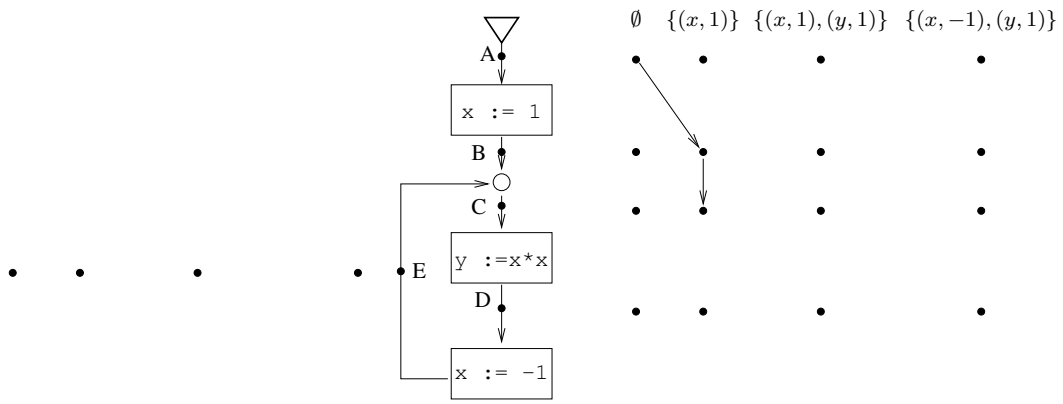


Perform the functional-approach based interprocedural version of this analysis on the given program, using a Kildall-style algorithm. Show the steps of the algorithm in tabular form, separately for the two stages (solving Eq (1) and Eq (2)).

Problem 3. We have seen in class that Kildall's algorithm computes only an overapproximation of the JOP for a given instance of an abstract interpretation. However, if the underlying lattice is *finite* we can give an algorithm to compute the *exact* JOP. This question asks you to suggest such an algorithm. (1.5pages, 15 *marks*)

(a) Consider the program below and the Constant Propogation abstract interpretation done in class, with initial value $\emptyset$. Consider only the 4 abstract data values shown in the figure. Construct an "exploded" control-flow graph as shown below, where each program point is duplicated 4 times (one for each of the abstract values considered). Beginning with the initial value $\emptyset$ at the initial

point A (we call this node in the graph "$(A, \emptyset)$"), add edges to the sucessor points by applying the transfer function for the concerned nodes. The first two steps are shown in the figure.

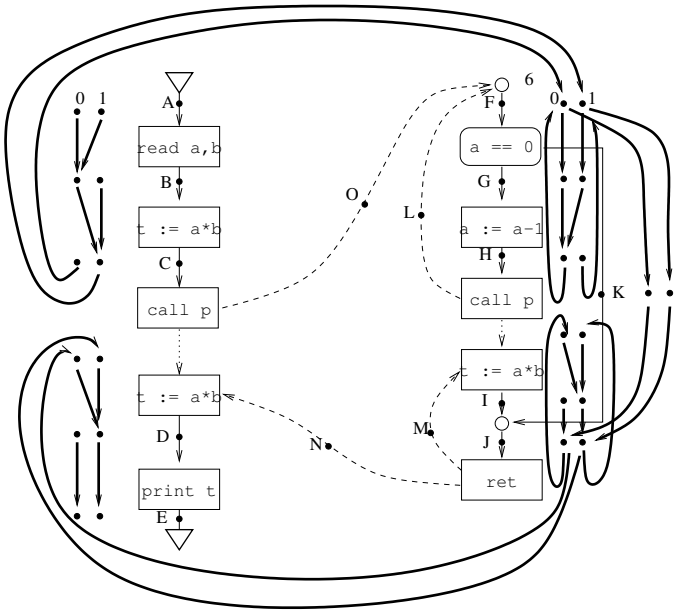


(i) Complete the procedure described above till no more edges can be added. Show the final set of edges added.

(ii) Deduce the JOP values at each point based on this exploded graph.

(b) Suggest an algorithm to compute the exact JOP for a given program P and an abstract interpretation $\mathcal{A} = (D, \leq, f_{MN}, d_0)$.

Problem 4. In a similar way to the previous problem, this problem is about a way to compute the *exact* JVP for a program with procedures. Consider the program below for which we want to do an analysis for the availability of `a*b`. We build an exploded graph representing the program and the abstract values at each program point. The edges connect an abstract data point d at program point M, to an abstract value d' at program point N, if there is a control flow edge from M to N, and $f_{MN}(d) = d'$.



Answer the following questions: (1.5pages, 15 *marks*)

(a) Draw the complete exploded graph for this program and analysis (the given graph may not be accurate, so do it on your own). Take the initial data value as 0 (unavailable).

(b) By inspection, say whether the point $(D, 0)$ and $(D, 1)$ are reachable (respectively) from $(A, 0)$. Thereby infer the JVP at D.

(c) Give an algorithm to compute the JVP at each point for such a program and analysis. *Hint:* Give a way to algorithmically check reachability from a point (S, d) to another point (N, e) in this exploded graph. This is also called the "CFL Reacability" problem, where one is given a finite graph G whose edges are labelled by letters from a finite alphabet A, and a context-free language L over A (via a context-free grammar or a pushdown automaton over A), and we need to tell whether a given target vertex t in G can be reached from a source vertex s in G, via a path in G *whose labels form a string* in L.