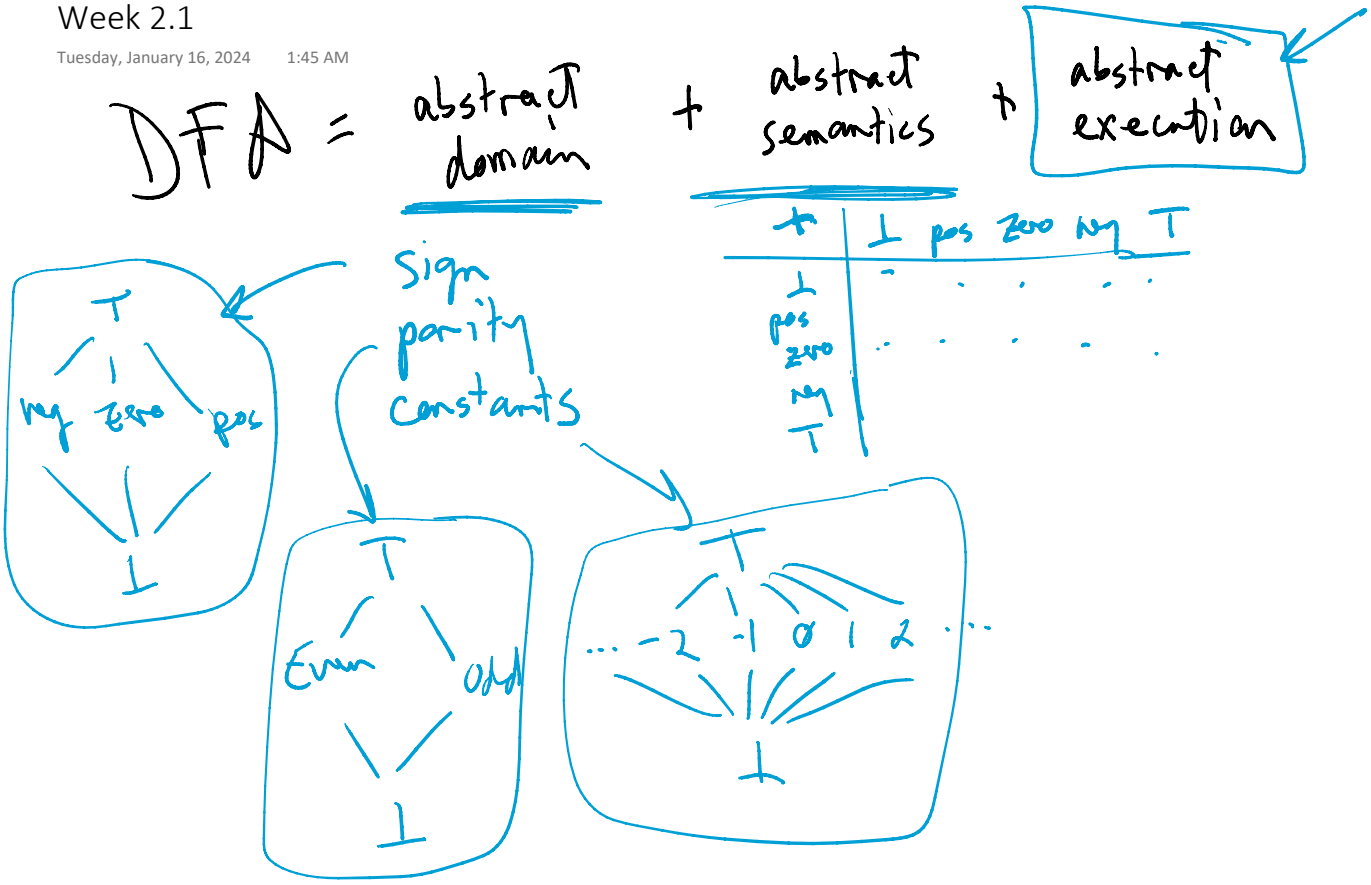
Week 2.1

Tuesday, January 16, 2024 1:45 AM

DFA = abstract domain + abstract semantics + abstract execution



---

abstract execution

↳ flow-sensitive analysis: computes a result for each program point

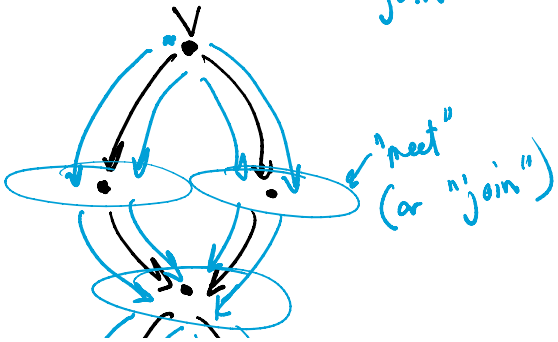
↳ in contrast to flow-insensitive analysis

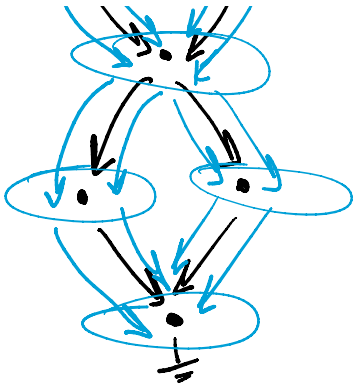
x = -2 [x↦neg]

x = 2 [x↦pos]

MOP ("meet" over all paths)

↳ "join"





MFP (maximal/minimal fixpoint)

- treat the abstract semantics as a system of equations & compute their fixpoint (specifically, the "most precise" fixpoint)

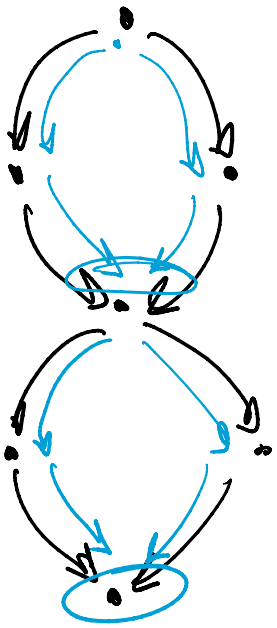
fixpoint

- $f(x) = x$
- there can be $\emptyset$, 1, finite, or infinite fixpoints
- $f(x) = x^2$ fixpoints $= \{\emptyset, 1\}$, least fixpoint $= \emptyset$

MFP: assign abstract values to each program point s.t. if we execute the abstract semantics then we get the same result

the abstract semantics
the same result

↳ want the most precise values



What is relation between MFP soln & the MOP soln?

↳ an analysis is distributive iff
for abstract semantics F, $join(F(x), F(y)) = F(join(x,y))$

↳ for a distributive analysis, MOP = MFP
otherwise MFP ⊑ MOP
"less precise"

---

d. fixing an intraprocedural, pointer-concurrent

defining an intraprocedural, pointer-conservative integer-based analysis

need
- abstract domain
- abstract semantics
- MFP worklist alg

MFP worklist alg

setup
- map "bb2store" : BasicBlock ⟼ entry abstract store
- bb2store['entry'] = initial abstract store
  - ↳ parameters & globals set to ⊤ ('int'-typed)
- worklist of basic blocks
  - ↳ queue, stack, priority queue, ...
- initially, worklist contains 'entry'

analysis

```
while worklist is not empty:
    bb = worklist.pop()
    let σ = bb2store[bb]          <- this is a copy
    execute bb on σ, updating σ as appropriate
```

execute bb on σ, updating σ as appropriate

at terminal:

for each target basic block bb':

bb2store[bb'] = bb2store[bb'] $\sqcup$ σ

if bb2store[bb'] changed:

put bb' on worklist

We end w/ bb2store mapping to entry abstract stores

we might want exit abstract store

↳ just take bb2store, execute each basic block once, remember the result