

Data-Flow Analysis

Paul Gazzillo

Data-Flow Analysis

- How do values flow through a program?
- Control-flow analysis useful first step
 - Order that instructions happen

Why Data-Flow Analysis? Optimization

- E.g., use registers instead of load/store
- Start with systematic translation

```
a = x * 2
...
b = a + 1
```



```
load a -> r1
load x -> r2
mult r2, 2 -> r1
store r1 -> a
...
load a -> r1
add r1, 1 -> r3
store r3 -> b
```

Why Data-Flow Analysis? Optimization

- E.g., use registers instead of load/store
- (at least) two improvements
- Stop using r2 for "x" (liveness analysis)
 - Use 2 registers instead of 3
- Keep "a" in a register (reaching definitions)
 - No more store/load

```
load a -> r1
load x -> r2
mult r2, 2 -> r1
store r1 -> a
...
load a -> r1
add r1, 1 -> r3
store r3 -> b
```



```
load a -> r1
load x -> r2
mult r2, 2 -> r1
store r1 -> a
...
load a -> r1
add r1, 1 -> r2
store r2 -> b
```

Reaching Definitions

- Tracks
 - values created at assignments (definitions)
 - to basic blocks
- For each block
 - GEN – new definitions
 - KILL – overwritten definitions
- Setup flow equations: REACHin and REACHout
- Solve by iterating to fixpoint
 - Update every block until no more updates