

Data Dependence

Definition

- Data dependences are constraints on the order in which statements may be executed

We say statement s_2 depends on s_1

- **Flow (true) dependence:** s_1 writes memory that s_2 later reads (RAW)
- **Anti-dependence:** s_1 reads memory that s_2 later writes (WAR)
- **Output dependences:** s_1 writes memory that s_2 later writes (WAW)
- **Input dependences:** s_1 reads memory that s_2 later reads (RAR)

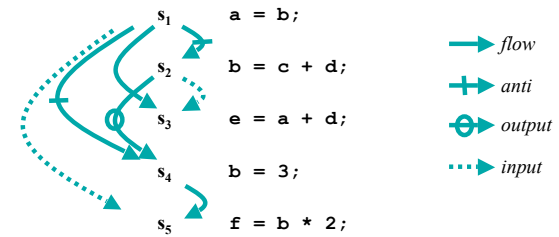
True dependences

- Flow dependences represent actual flow of data

False dependences

- Anti- and output dependences reflect reuse of memory, not actual data flow; can often be eliminated

Example



Representing Data Dependences

Implicitly

- Using variable defs and uses
- Pros: simple
- Cons: hides data dependence (analyses must find this info)

Def-use chains (du chains)

- Link each def to its uses
- Pros: explicit; therefore fast
- Cons: must be computed and updated, space consuming

Alternate representations

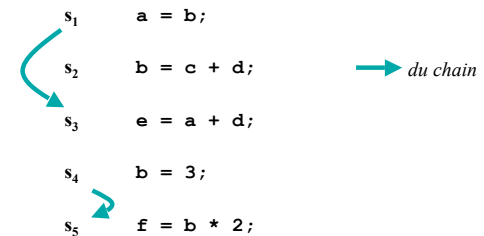
- e.g., Static single assignment form (SSA), dependence flow graphs (DFG), value dependence graphs (VDG)

DU Chains

Definition

- du chains link each def to its uses

Example

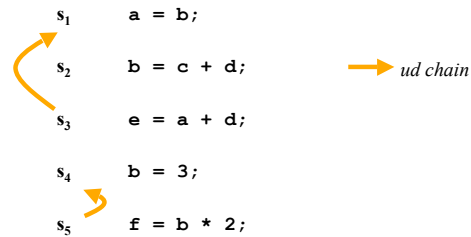


UD Chains

Definition

- ud chains link each use to its defs

Example



CS553 Lecture

Static Single Assignment Form

7

Role of Alternate Program Representations

Advantage

- Allow analyses and transformations to be simpler & more efficient/effective

Disadvantage

- May not be "executable" (requires extra translations to and from)
- May be expensive (in terms of time or space)

Process



CS553 Lecture

Static Single Assignment Form

8

Static Single Assignment (SSA) Form

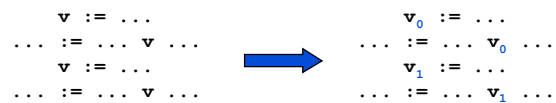
Idea

- Each variable has only one static definition
- Makes it easier to reason about values instead of variables
- Similar to the notion of functional programming

Transformation to SSA

- Rename each definition
- Rename all uses reached by that assignment

Example



What do we do when there's control flow?

CS553 Lecture

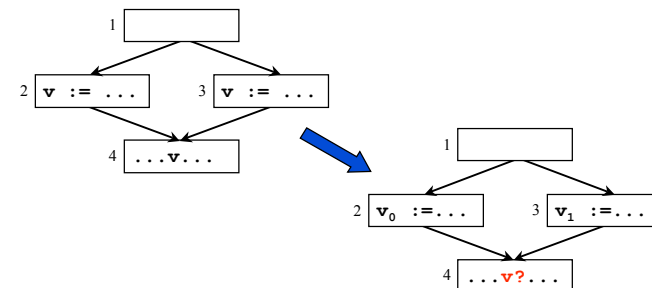
Static Single Assignment Form

9

SSA and Control Flow

Problem

- A use may be reached by several definitions



CS553 Lecture

Static Single Assignment Form

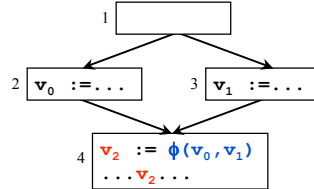
10

SSA and Control Flow (cont)

Merging Definitions

- ϕ -functions merge multiple reaching definitions

Example

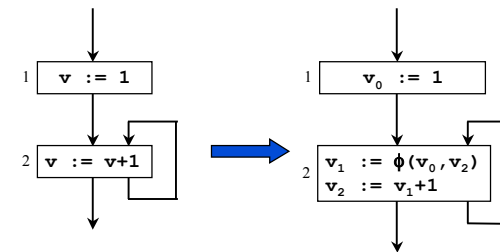


CS553 Lecture

Static Single Assignment Form

11

Another Example



CS553 Lecture

Static Single Assignment Form

12

SSA vs. ud/du Chains

SSA form is more constrained

Advantages of SSA

- More compact
- Some analyses become simpler when each use has only one def
- Value merging is explicit
- Easier to update and manipulate?

Furthermore

- Eliminates false dependences (simplifying context)

```

for (i=0; i<n; i++)
  A[i] = i;
for (i=0; i<n; i++)
  print(foo(i));
  
```

Unrelated uses of i are given different variable names

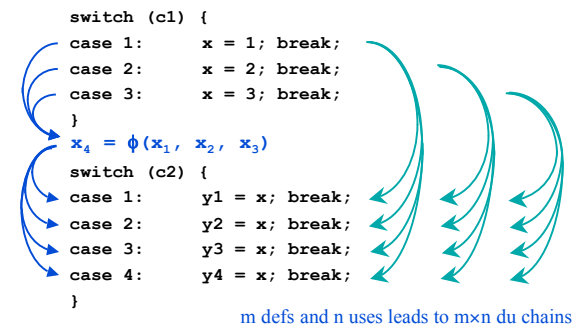
CS553 Lecture

Static Single Assignment Form

13

SSA vs. ud/du Chains (cont)

Worst case du-chains?



CS553 Lecture

Static Single Assignment Form

14

Transformation to SSA Form

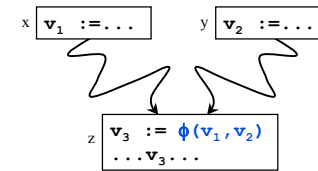
Two steps

- Insert ϕ -functions
- Rename variables

Where Do We Place ϕ -Functions?

Basic Rule

- If two distinct (non-null) paths $x \rightarrow z$ and $y \rightarrow z$ converge at node z , and nodes x and y contain definitions of variable v , then a ϕ -function for v is inserted at z



Approaches to Placing ϕ -Functions

Minimal

- As few as possible subject to the basic rule

Briggs-Minimal

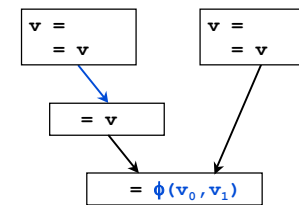
- Same as minimal, except v must be live across some edge of the CFG

Pruned

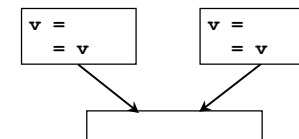
- Same as minimal, except dead ϕ -functions are not inserted

What's the difference between Briggs Minimal and Pruned SSA?

Briggs Minimal vs. Pruned



Briggs Minimal will add a ϕ function because v is live across the blue edge, but Pruned SSA will not because the ϕ function is dead



Neither Briggs Minimal nor Pruned SSA will place a ϕ function in this case because v is not live across any CFG edge

Why would we ever use Briggs Minimal instead of Pruned SSA?

Concepts

Data dependences

- Three kinds of data dependences
- du-chains

Alternate representations

SSA form

Conversion to SSA form

- ϕ -function placement

Next Time

Assignments

- HW1 due

Lecture

- SSA continued