Program Analysis

Data Flow Analysis (Part 2)
Outline

- First example: Available expressions
- Basic principles
- More examples
- Solving data flow problems
- Inter-procedural analysis
- Sensitivities
Defining a Data Flow Analysis

Any data flow analysis: Defined by six properties

- Domain
- Direction
- Transfer function
- Meet operator
- Boundary condition
- Initial values
Domain

- Analysis associates some information with every program point
  - “Information” means elements of a set
- Domain of the analysis: All possible elements the set may have
  - E.g., for available expressions analysis:
    Domain is set of non-trivial expressions
Direction

- Analysis propagates information along the control flow graph
  - Forward analysis: Normal flow of control
  - Backward analysis: Invert all edges
    - Reasons about executions in reverse
- E.g., available expression analysis: Forward
Transfer Function

- Defines how a \textit{statement affects the propagated information}

\[ DF_{\text{exit}}(s) = \text{some function of } DF_{\text{entry}}(s) \]

- E.g., for available expression analysis:

\[ AE_{\text{exit}}(s) = (AE_{\text{entry}}(s) \setminus \text{kill}(s)) \cup \text{gen}(s) \]
Meet Operator

- What if two statements $s_1, s_2$ flow to a statement $s$?
  - Forward analysis: Execution branches merge
  - Backward analysis: Branching point

- Meet operator defines how to combine the incoming information
  - Union: $DF_{entry}(s) = DF_{exit}(s_1) \cup DF_{exit}(s_2)$
  - Intersection: $DF_{entry}(s) = DF_{exit}(s_1) \cap DF_{exit}(s_2)$
Meet Operator

- What if two statements $s_1, s_2$ flow to a statement $s$?
  - Forward analysis: Execution branches merge
  - Backward analysis: Branching point

- Meet operator defines how to combine the incoming information
  - Union: $DF_{entry}(s) = DF_{exit}(s_1) \cup DF_{exit}(s_2)$
  - Intersection: $DF_{entry}(s) = DF_{exit}(s_1) \cap DF_{exit}(s_2)$

E.g., available expressions analysis
Boundary Condition

What information to start with at the first CFG node?

- Forward analysis: First node is entry node
- Backward analysis: First node is exit node

Common choices

- Empty set
- Entire domain
Boundary Condition

- **What information to start with at the first CFG node?**
  - Forward analysis: First node is entry node
  - Backward analysis: First node is exit node

- **Common choices**
  - Empty set
  - Entire domain

E.g., available expressions analysis
Initial Values

- What is the information to start with at intermediate nodes?

- Common choices
  - Empty set
  - Entire domain
Initial Values

- What is the information to start with at intermediate nodes?

- Common choices
  - Empty set
  - Entire domain

E.g., available expressions analysis
Defining a Data Flow Analysis

Any data flow analysis:
Defined by six properties

- Domain
- Direction
- Transfer function
- Meet operator
- Boundary condition
- Initial values
Defining a Data Flow Analysis

Any data flow analysis:
Defined by six properties

- Domain
- Direction
- Transfer function
- Meet operator
- Boundary condition
- Initial values

- Non-trivial expressions
- Forward
- $AE_{exit}(s) = (AE_{entry} \setminus \text{kill}(s)) \cup \text{gen}(s)$
- Intersection ($\cap$)
- $AE_{entry}(\text{entryNode}) = \emptyset$
- $\emptyset$

Example: Available expressions