Data Flow Analysis

Soonho Kong
soonhok@cs.cmu.edu

2011/1/26
Computer Science Department
Carnegie Mellon University
Motivating Example: **Reaching Definitions**

A definition $d$ reaches a point $p$ if there is a path from the point immediately following $d$ to $p$.
Motivating Example: **Reaching Definitions**

A definition $d$ *reaches* a point $p$ if there is a path from the point immediately following $d$ to $p$. 

\[
\begin{align*}
\text{B0} & : \begin{array}{l}
d0: y = 3 \\
d1: x = 10 \\
d2: y = 11 \\
\text{if e}
\end{array} \\
\text{B1} & : \begin{array}{l}
d3: x = 1 \\
d4: y = 2
\end{array} \\
\text{B2} & : \begin{array}{l}
d5: z = x \\
d6: x = 4
\end{array} \\
\text{In[B0]} & : \{ \} \\
\text{Out[B0]} & \\
\text{In[B1]} & \\
\text{Out[B1]} & \\
\text{In[B2]} & \\
\text{Out[B2]} & \\
\text{In[B3]} & \\
\text{Out[B3]} & \\
\end{align*}
\]
Motivating Example: **Reaching Definitions**

A definition \(d\) reaches a point \(p\) if there is a path from the point immediately following \(d\) to \(p\).
Motivating Example: Reaching Definitions

A definition $d$ reaches a point $p$ if there is a path from the point immediately following $d$ to $p$. 
Motivating Example: **Reaching Definitions**

A definition d **reaches** a point p if there is a path from the point immediately following d to p.
Motivating Example: Reaching Definitions

A definition \( d \) reaches a point \( p \) if there is a path from the point immediately following \( d \) to \( p \).
Motivating Example: Reaching Definitions

A definition \(d\) **reaches** a point \(p\) if there is a path from the point immediately following \(d\) to \(p\).
Motivating Example: **Reaching Definitions**

A definition $d$ **reaches** a point $p$ if there is a path from the point immediately following $d$ to $p$. 

---

**d₀**: $y = 3$
**d₁**: $x = 10$
**d₂**: $y = 11$
if $e$

---

**B₁**

**d₃**: $x = 1$
**d₄**: $y = 2$

---

**B₂**

**d₅**: $z = x$
**d₆**: $x = 4$

---

**B₃**

**d₂, d₃, d₄, d₅, d₆**
Transfer Function (statement-level)

What is the relation between $\text{In}[s_0]$ and $\text{Out}[s_0]$?

$$\text{Out}[s_0] = f_{d_0}(\text{In}[s_0])$$

$$f_d(x) = \text{gen}_d \cup (x - \text{kill}_d)$$

where

- $\text{gen}_d$ is definition generated: $\text{gen}_d = \{d\}$
- $\text{kill}_d$ is set of all otherdefs to $x$ in the rest of program
Transfer Function (statement-level)

$\text{d0: } y = 3$
$\text{d1: } x = 10$
$\text{d2: } y = 11$
$\text{if e}
$\text{d5: } z = x$
$\text{d6: } x = 4$

$\text{gen}_{d_0} = \{d_0\}$
$\text{kill}_{d_0} = \{d_2, d_4\}$
$\text{f}_{d_0}(x) = \text{gen}_{d_0} \cup (x - \text{kill}_{d_0})$
$= \{d_0\} \cup (x - \{d_2, d_4\})$
Transfer Function (block-level)

What is the relation between In\[B0\] and Out\[B0\]?

\[
Out[s_0] = f_B(In[s_0])
\]

\[
f_B(x) = f_{d_n} \circ \cdots \circ f_{d_0}(x)
\]
Transfer Function (block-level)

\[
Out[B_0] = \{d_1, d_2\} \cup (In[B_0] - \{d_3, d_4, d_6, d_0\})
\]
\[
Out[B_1] = \{d_3, d_4\} \cup (In[B_1] - \{d_0, d_1, d_2, d_6\})
\]
\[
Out[B_2] = \{d_5, d_6\} \cup (In[B_2] - \{d_1, d_3\})
\]
Meet Operator

What is the relation between $\text{Out}[B_1]$, $\text{Out}[B_2]$, and $\text{Out}[B_0]$?

$$\text{In}[B] = \bigcup_{B' : \text{pred}(B)} \text{Out}[B']$$
Meet Operator

What is the relation between $\text{Out[B1]}$, $\text{Out[B2]}$, and $\text{Out[B0]}$?

$$\text{In[B]} = \bigcup_{B': \text{pred}(B)} \text{Out[B']}$$

Why $\bigcup$?

“A definition $d$ reaches a point $p$ if there is a path from the point immediately following $d$ to $p$”
Meet Operator

What is the relation between \(\text{Out}[B_1]\), \(\text{Out}[B_2]\), and \(\text{Out}[B_0]\)?

\[
\text{In}[B] = \bigcup_{B' : \text{pred}(B)} \text{Out}[B']
\]

What if “MUST reach definition”?

“A definition \(d\) reaches a point \(p\) if for all path from the point immediately following \(d\) to \(p\)”

\[
\text{In}[B] = \bigcap_{B' : \text{pred}(B)} \text{Out}[B']
\]
What is the relation between Out[B1], Out[B2], and Out[B0]?

\[
\text{In}[B] = \bigcup_{B' : \text{pred}(B)} \text{Out}[B']
\]

In general

\[
\text{In}[B] = \bigcap_{B' : \text{pred}(B)} \text{Out}[B']
\]

and \( \bigcap \) depends on the problem.
Meet Operator

\[
\begin{align*}
In[B_1] &= Out[B_0] \\
In[B_2] &= Out[B_0] \\
In[B_3] &= Out[B_1] \cup Out[B_2]
\end{align*}
\]
Boundary Condition

So far, we have...

\[ \text{Out}[B_0] = \{d_1, d_2\} \cup (\text{In}[B_0] - \{d_3, d_4, d_6, d_0\}) \quad \text{In}[B_1] = \text{Out}[B_0] \]
\[ \text{Out}[B_1] = \{d_3, d_4\} \cup (\text{In}[B_1] - \{d_0, d_1, d_2, d_6\}) \quad \text{In}[B_2] = \text{Out}[B_0] \]
\[ \text{Out}[B_2] = \{d_5, d_6\} \cup (\text{In}[B_2] - \{d_1, d_3\}) \quad \text{In}[B_3] = \text{Out}[B_1] \cup \text{Out}[B_2] \]
Boundary Condition

What $\text{In}[B0]$ should be?

$\text{In}[B0] = \{\}$

It depends on the problem.
Extracting Constraints

B0
- \( d_0: y = 3 \)
- \( d_1: x = 10 \)
- \( d_2: y = 11 \) if \( e \)

B1
- \( d_3: x = 1 \)
- \( d_4: y = 2 \)

B2
- \( d_5: z = x \)
- \( d_6: x = 4 \)

In\[B0\] = \{\}

Out\[B0\] = \( \{d_1, d_2\} \cup (In[B_0] - \{d_3, d_4, d_6, d_0\}) \)

Out\[B_1\] = \( \{d_3, d_4\} \cup (In[B_1] - \{d_0, d_1, d_2, d_6\}) \)

Out\[B_2\] = \( \{d_5, d_6\} \cup (In[B_2] - \{d_1, d_3\}) \)

In\[B_1\] = Out\[B_0\]

In\[B_2\] = Out\[B_0\]

In\[B_3\] = Out\[B_1\] \( \cup \) Out\[B_2\]

Meet Operator

Boundary Condition
Solving Constraints

Transfer Function

\[
\begin{align*}
    Out[B_0] &= \{d_1, d_2\} \cup (In[B_0] - \{d_3, d_4, d_6, d_0\}) \\
    Out[B_1] &= \{d_3, d_4\} \cup (In[B_1] - \{d_0, d_1, d_2, d_6\}) \\
    Out[B_2] &= \{d_5, d_6\} \cup (In[B_2] - \{d_1, d_3\})
\end{align*}
\]

Meet Operator

\[
\begin{align*}
    In[B_1] &= Out[B_0] \\
    In[B_2] &= Out[B_0] \\
    In[B_3] &= Out[B_1] \cup Out[B_2] \\
    In[B_0] &= \{\}
\end{align*}
\]

Boundary Condition

Goal: Find \((In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2])\) satisfying the above constraints.
Solving Constraints

Transfer Function

| $Out[B_0]$ | $= \{ d_1, d_2 \} \cup (In[B_0] - \{ d_3, d_4, d_6, d_0 \})$ |
| $Out[B_1]$ | $= \{ d_3, d_4 \} \cup (In[B_1] - \{ d_0, d_1, d_2, d_6 \})$ |
| $Out[B_2]$ | $= \{ d_5, d_6 \} \cup (In[B_2] - \{ d_1, d_3 \})$ |

Meet Operator

| $In[B_1]$ | $= Out[B_0]$ |
| $In[B_2]$ | $= Out[B_0]$ |
| $In[B_3]$ | $= Out[B_1] \cup Out[B_2]$ |
| $In[B_0]$ | $= \{ \}$ |

Boundary Condition

Goal: Find $(In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2])$ satisfying the above constraints.

THIS IS A FIXED POINT PROBLEM

$$\mathcal{F}(In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) = (In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2])$$
Solving Constraints

**Transfer Function**

\[
\begin{align*}
Out[B_0] &= \{d_1, d_2\} \cup (In[B_0] - \{d_3, d_4, d_6, d_0\}) \\
Out[B_1] &= \{d_3, d_4\} \cup (In[B_1] - \{d_0, d_1, d_2, d_6\}) \\
Out[B_2] &= \{d_5, d_6\} \cup (In[B_2] - \{d_1, d_3\})
\end{align*}
\]

\[
\begin{align*}
In[B_1] &= Out[B_0] \\
In[B_2] &= Out[B_0] \\
In[B_3] &= Out[B_1] \cup Out[B_2] & \text{Meet Operator} \\
In[B_0] &= \{\} & \text{Boundary Condition}
\end{align*}
\]

**Goal:** Find \((In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2])\) satisfying the above constraints.

**THIS IS A FIXED POINT PROBLEM**

\[
\mathcal{F}(In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) = (In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2])
\]

Do we have a solution for this problem? If it is, how can we compute it?...
Basic Fixpoint Theorems

If \( \tau \) is monotonic, then it has a least fixpoint, \( \text{lfp} Z [\tau(Z)] \), and a greatest fixpoint, \( \text{gfp} Z [\tau(Z)] \).

\[
\text{lfp} Z [\tau(Z)] = \bigcap \{Z \mid \tau(Z) = Z\} \text{ whenever } \tau \text{ is monotonic.}
\]

\[
\text{lfp} Z [\tau(Z)] = \bigcup_i \tau^i(\text{False}) \text{ whenever } \tau \text{ is also } \bigcup\text{-continuous;}
\]

\[
\text{gfp} Z [\tau(Z)] = \bigcup \{Z \mid \tau(Z) = Z\} \text{ whenever } \tau \text{ is monotonic.}
\]

\[
\text{gfp} Z [\tau(Z)] = \bigcap_i \tau^i(\text{True}) \text{ whenever } \tau \text{ is also } \bigcap\text{-continuous.}
\]
As a consequence of the preceding lemmas, if $\tau$ is monotonic, its least fixpoint can be computed by the following program.

```
function Lfp($\tau$ : PredicateTransformer)
begin
    $Q := False$;
    $Q' := \tau(Q)$;
    while ($Q \neq Q'$) do
    begin
        $Q := Q'$;
        $Q' := \tau(Q')$
    end;
    return $Q$
end
```
Solving Constraints

THIS IS A FIXED POINT PROBLEM
\[ F(In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) = (In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) \]

Do we have a solution for this problem? If it is, how can we compute it? Can we get a solution within finite time?
Solving Constraints

This is a fixed point problem

\[ F(\text{In}[B_0], \text{In}[B_1], \text{In}[B_2], \text{In}[B_3], \text{Out}[B_0], \text{Out}[B_1], \text{Out}[B_2]) = (\text{In}[B_0], \text{In}[B_1], \text{In}[B_2], \text{In}[B_3], \text{Out}[B_0], \text{Out}[B_1], \text{Out}[B_2]) \]

Do we have a solution for this problem? If it is, how can we compute it? Can we get a solution within finite time?

Yes, we have a solution only if \( F \) is monotone.

Yes, we can compute a solution only if \( F \) is continuous.

Yes, we can compute a solution in finite time only if the domain has descending chain condition (DCC).
Solving Constraints

THIS IS A FIXED POINT PROBLEM
\[ F(In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) = (In[B_0], In[B_1], In[B_2], In[B_3], Out[B_0], Out[B_1], Out[B_2]) \]

Do we have a solution for this problem? If it is, how can we compute it? Can we get a solution within finite time?

Yes, we have a solution only if \( F \) is monotone.
Yes, we can compute a solution only if \( F \) is continuous.
Yes, we can compute a solution in finite time only if we the domain has descending chain condition (DCC).

\[ F : V^7 \rightarrow V^7 \text{ is monotone, continuous, and the domain has DCC.} \]

Why?
Solving Constraints

**THIS IS A FIXED POINT PROBLEM**

\[ \mathcal{F}(\text{In}[B_0], \text{In}[B_1], \text{In}[B_2], \text{In}[B_3], \text{Out}[B_0], \text{Out}[B_1], \text{Out}[B_2]) = (\text{In}[B_0], \text{In}[B_1], \text{In}[B_2], \text{In}[B_3], \text{Out}[B_0], \text{Out}[B_1], \text{Out}[B_2]) \]

Do we have a solution for this problem? If it is, how can we compute it? Can we get a solution within finite time?

Yes, we have a solution only if \( \mathcal{F} \) is monotone.

Yes, we can compute a solution only if \( \mathcal{F} \) is continuous.

**Algorithm**

```plaintext
// Boundary Condition
In[B0] = {} // Initialization for iterative algorithm

For each basic block B
    Out[B] = {}

// iterate
while (Changes to any In[], Out[] occur) {
    For each basic block B {
        In[B] = meet(Out[p_0], ... Out[p_1])
        Out[B] = f_B(In[B])
    }
}
```

\[ \text{In}[B] = \bigcup_{B' : \text{pred}(B)} \text{Out}[B'] \]

\[ \text{Out}[s_0] = f_B(\text{In}[s_0]) \]
Solving Constraints

// Boundary Condition
In[B0] = { }

// Initialization for iterative algorithm
For each basic block B
  Out[B] = { }

// iterate
while(Changes to any In[] , Out[] occur) {
  For each basic block B {
    In[B] = meet(Out[p_0], ... Out[p_1])
    Out[B] = f_B(In[B])
  }
}

Ifp \( Z[\tau(Z)] = \bigcup \tau^i(False) \) whenever \( \tau \) is also \( \cup \)-continuous;

\[
In[B] = \bigcup_{B':\text{pred}(B)} Out[B'] \\
Out[s_0] = f_B(In[s_0])
\]

Does this algorithm terminate? If so, what is complexity?
Solving Constraints

```plaintext
// Boundary Condition
In[BO] = { }

// Initialization for iterative algorithm
For each basic block B
  Out[B] = { } [\textbf{Ifp} Z[\tau(Z)] = \bigcup_{i} \tau^i(False) \text{ whenever } \tau \text{ is also } \cup\text{-continuous;}

// iterate
while(Changes to any In[], Out[] occur) {
  For each basic block B {
    In[B] = \text{meet}(Out[p_0], \ldots, Out[p_1])
    Out[B] = f_B(In[B])
  }
}
```

Does this algorithm terminate? If so, what is complexity?

Yes, why? Since \( \mathcal{F} : V^7 \rightarrow V^7 \) is monotone, and the domain has DCC property.
Summary: Reaching Definition

Reaching definition problem is defined by

- **Domain of values:** $V = 2\{d_0, d_1, d_2, d_3, d_4, d_5, d_6\}$
- **Meet operator:** $\cup : V \rightarrow V$
- **Boundary Condition:** $In[B_0] = \{\}$
- **Set of Transfer Functions:**
  
  \[
  \{f_{B_0}, f_{B_1}, f_{B_2}, f_{B_3}, f_{B_4}, f_{B_5}, f_{B_6}\} : 2^V \rightarrow V
  \]
A Unified Framework

Data flow problems are defined by

- Domain of values: $V$ (meet-semilattice is enough).
- Meet operator: $\sqcap : V \rightarrow V$
- Boundary Condition: value of entry/exit node
- Set of Transfer Functions: $2^V \rightarrow V$
Meet-semilattice

A set $S$ **partially ordered** by the binary relation $\sqsubseteq$ if for all $x, y, z$:

- reflexive $x \sqsubseteq x$
- anti-symmetric $x \sqsubseteq y \land y \sqsubseteq x \implies x = y$
- transitive $x \sqsubseteq y \land y \sqsubseteq z \implies x \sqsubseteq z$

A poset $S$ is **meet-semilattice** if for all elements $x$ and $y$ of $S$, the greatest lower bound of the set $\{x, y\}$ exists.

$$\forall x, y \in S : \exists \sqcap \{x, y\} \in S$$

Semi-lattice has the top element.

$$\exists T \in S : \forall x \in S : x \sqsubseteq T$$
Meet-semilattice

- Example: \( \{d0, d1, d2, d3, d4, d5, d6\}, \supseteq \)  
  What is \( \bigwedge \) and \( \bigvee \) ?
General Iterative Data Flow Analysis Algorithm (Forward)

// Boundary Condition
Out[Entry] = V_Entry

// Initialization for iterative algorithm
For each basic block B
    Out[B] = Top

// iterate
while(Changes to any Out[], In[] occur) {
    For each basic block B {
        In[B] = meet(Out[p_0], ... Out[p_1])
        Out[B] = f_B(In[B])
    }
}
Thank you