Points-to Analysis in Almost Linear Time

Bjarne Steensgaard

Martin Velez
Class Presentation
ECS289C @ UC Davis
Winter 2015

Motivation

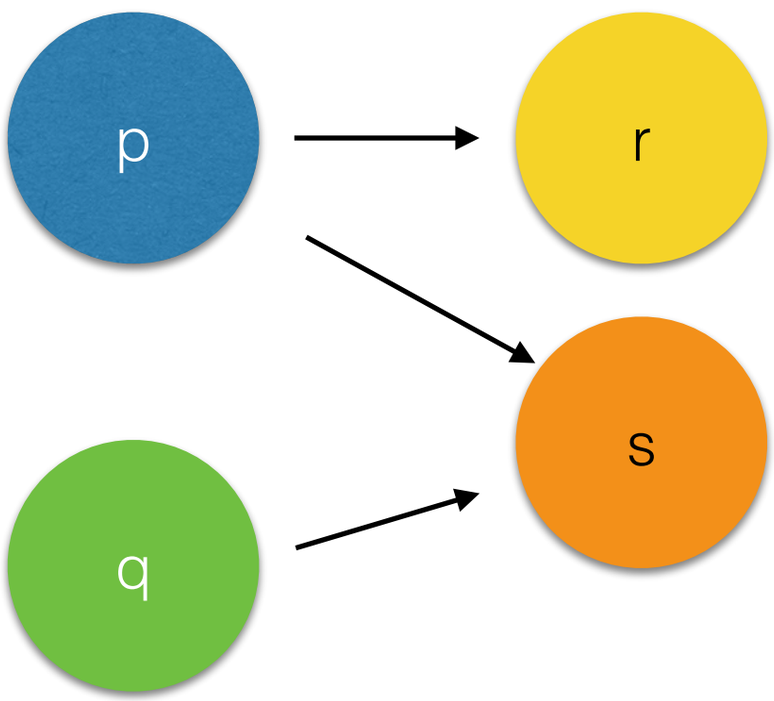
- Applications
  - Optimizing compilers
  - Program Understanding
  - Code browsing tools
  - Escape analysis
- Then state-of-the-art
  - Interprocedural analysis was slow but desired.
    - Whole-program optimization.

Flow-insensitive

- Order of statements is ignored
  - No control flow
  - Cannot do “strong updates” (overwrites)
    - `x = &a`
      - `a` is added to pts(x)

Points-to Graph

- |V| = n
- |E| = n^2

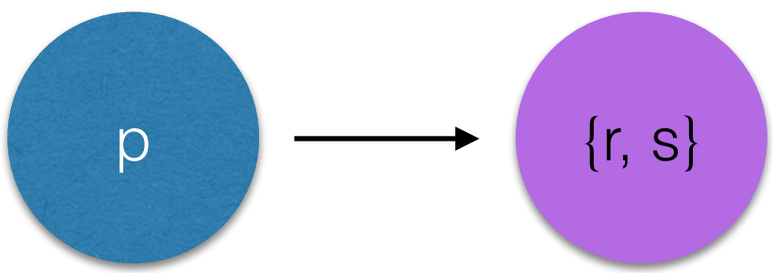


Vars	x_1	…	x_n
x_1	0		0
…		0	
x_n	1		0

n^2 entries

Smaller Points-to Graph

- |V| = n
- |E| = n

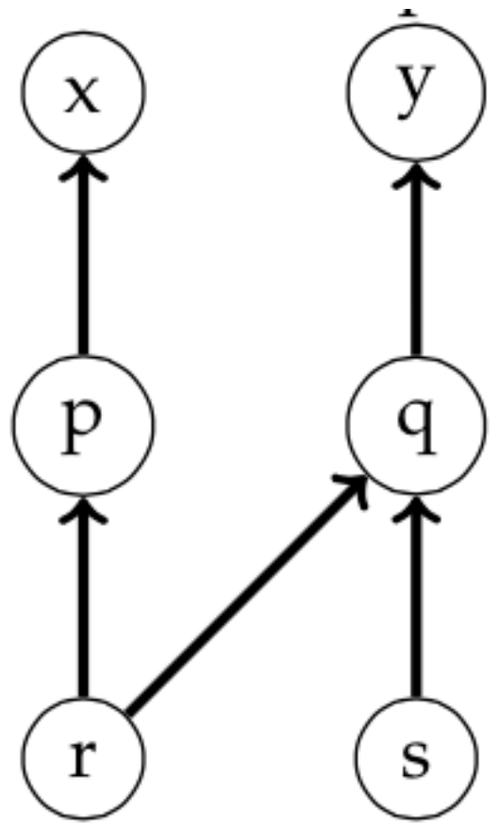


n entries

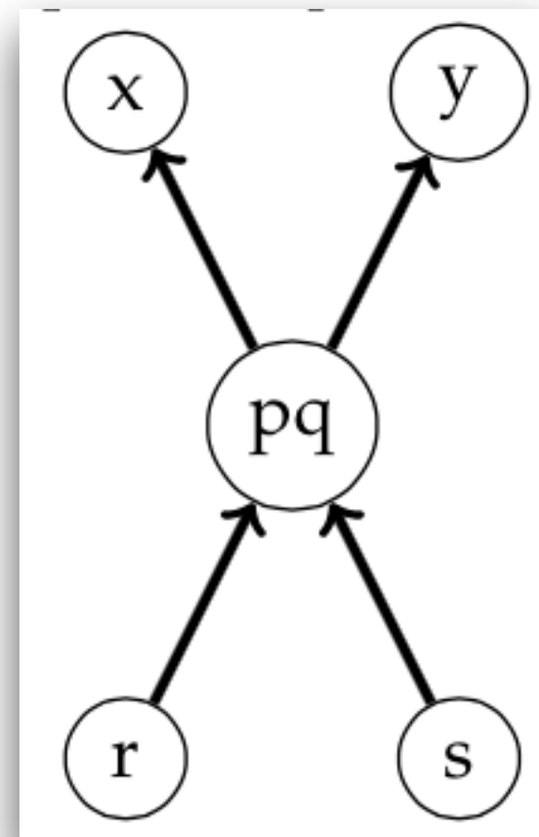
Variable	Points-to
x_1	x_j
…	…
x_i	x_k
…	…
x_n	x_l

Example 1

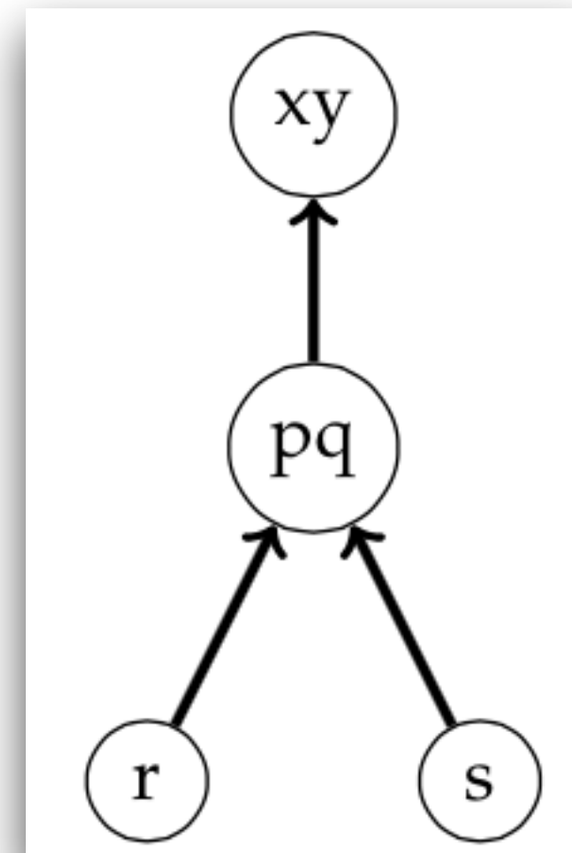
```
1. p = &x

2. r = &p

3. q = &y

4. s = &q

5. r = s
```



Andersen's
(More precise)



Join p and q

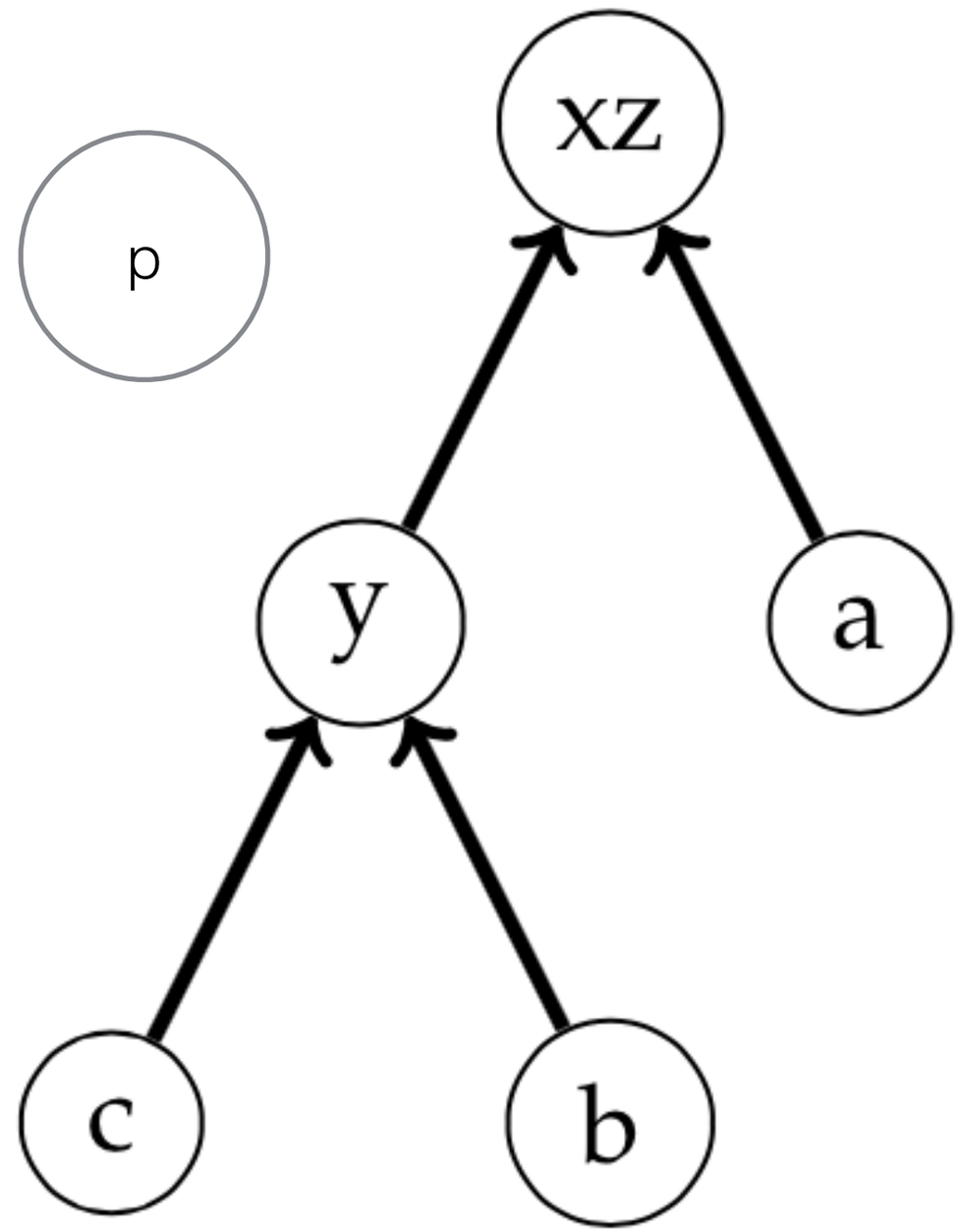


Join x and y

Steengaard's
(Faster)

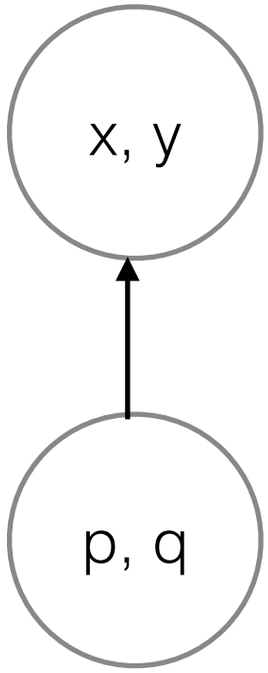
Example 2

```
a = &x
b = &y
if p then
  y = &z
else
  y = &x
c = &y
```

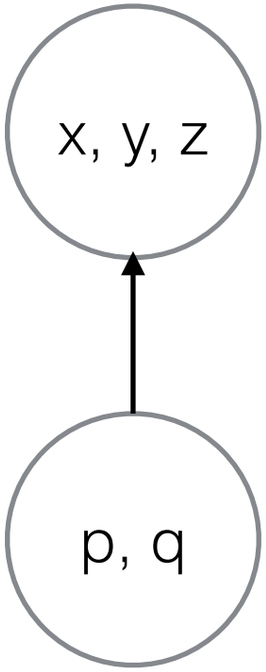


Example 3

```
1 q = &x
2 q = &y
3 p = q
4 q = &z
```



After Line 3



After Line 4

Simple example where Andersen's algorithm does mo more work.

$$
\begin{array}{lcl}
S & ::= & \mathsf{x} = \mathsf{y} \\
 & | & \mathsf{x} = \&\mathsf{y} \\
 & | & \mathsf{x} = *\mathsf{y} \\
 & | & \mathsf{x} = \mathsf{op}(\mathsf{y}_1 \ldots \mathsf{y}_n) \\
 & | & \mathsf{x} = \mathsf{allocate}(\mathsf{y}) \\
 & | & *\mathsf{x} = \mathsf{y} \\
 & | & \mathsf{x} = \mathsf{fun}(\mathsf{f}_1 \ldots \mathsf{f}_n) \rightarrow (\mathsf{r}_1 \ldots \mathsf{r}_m)\ S^* \\
 & | & \mathsf{x}_1 \ldots \mathsf{x}_m = \mathsf{p}(\mathsf{y}_1 \ldots \mathsf{y}_n)
\end{array}
$$

Syntax

Syntax of relevant statements in C-like language

$$
\begin{array}{lcl}
\alpha & ::= & \tau \times \lambda \\
\tau & ::= & \bot \mid \mathbf{ref}(\alpha) \\
\lambda & ::= & \bot \mid \mathbf{lam}(\alpha_1 \ldots \alpha_n)(\alpha_{n+1} \ldots \alpha_{n+m})
\end{array}
$$

locs or pointers to locs

functions or pointers to functions

Note on type equality:

t_i = t_j iff \bottom or i = j

Types

Non-standard

Typing Rule Example

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\alpha_1) \quad A \vdash \mathsf{y} : \mathbf{ref}(\alpha_2) \quad \alpha_2 \boxed{\trianglelefteq} \alpha_1}{A \vdash \mathit{welltyped}(\mathsf{x} = \mathsf{y})}$$

$$t_1 \boxed{\trianglelefteq} t_2 \Leftrightarrow (t_1 = \bot) \vee (t_1 = t_2)$$

$$\times\ t_2) \boxed{\trianglelefteq} (t_3 \times t_4) \Leftrightarrow (t_1 \trianglelefteq t_3) \wedge (t_2 \trianglelefteq$$

All Typing Rules

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\alpha_1) \quad A \vdash \mathsf{y} : \mathbf{ref}(\alpha_2) \quad \alpha_2 \trianglelefteq \alpha_1}{A \vdash \mathit{welltyped}(\mathsf{x} = \mathsf{y})}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\tau \times \_) \quad A \vdash \mathsf{y} : \tau}{A \vdash \mathit{welltyped}(\mathsf{x} = \&\mathsf{y})}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\alpha_1) \quad A \vdash \mathsf{y} : \mathbf{ref}(\mathbf{ref}(\alpha_2) \times \_) \quad \alpha_2 \trianglelefteq \alpha_1}{A \vdash \mathit{welltyped}(\mathsf{x} = *\mathsf{y})}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\alpha) \quad A \vdash \mathsf{y}_i : \mathbf{ref}(\alpha_i) \quad \forall i \in [1 \ldots n] : \alpha_i \trianglelefteq \alpha}{A \vdash \mathit{welltyped}(\mathsf{x} = \mathsf{op}(\mathsf{y}_1 \ \ldots \ \mathsf{y}_n))}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\mathbf{ref}(\_) \times \_)}{A \vdash \mathit{welltyped}(\mathsf{x} = \mathsf{allocate}(\mathsf{y}))}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\mathbf{ref}(\alpha_1) \times \_) \quad A \vdash \mathsf{y} : \mathbf{ref}(\alpha_2) \quad \alpha_2 \trianglelefteq \alpha_1}{A \vdash \mathit{welltyped}(*\mathsf{x} = \mathsf{y})}$$

$$\frac{A \vdash \mathsf{x} : \mathbf{ref}(\_ \times \mathbf{lam}(\alpha_1 \ldots \alpha_n)(\alpha_{n+1} \ldots \alpha_{n+m})) \quad A \vdash \mathsf{f}_i : \mathbf{ref}(\alpha_i) \quad A \vdash \mathsf{r}_j : \mathbf{ref}(\alpha_{n+j}) \quad \forall s \in S^* : A \vdash \mathit{welltyped}(s)}{A \vdash \mathit{welltyped}(\mathsf{x} = \mathsf{fun}(\mathsf{f}_1 \ldots \mathsf{f}_n) \rightarrow (\mathsf{r}_1 \ldots \mathsf{r}_m)\ S^*)}$$

$$\frac{A \vdash \mathsf{x}_j : \mathbf{ref}(\alpha'_{n+j}) \quad A \vdash \mathsf{p} : \mathbf{ref}(\_ \times \mathbf{lam}(\alpha_1 \ldots \alpha_n)(\alpha_{n+1} \ldots \alpha_{n+m})) \quad A \vdash \mathsf{y}_i : \mathbf{ref}(\alpha'_i) \quad \forall i \in [1 \ldots n] : \alpha'_i \trianglelefteq \alpha_i \quad \forall j \in [1 \ldots m] : \alpha_{n+j} \trianglelefteq \alpha'_{n+j}}{A \vdash \mathit{welltyped}(\mathsf{x}_1 \ldots \mathsf{x}_m = \mathsf{p}(\mathsf{y}_1 \ldots \mathsf{y}_n))}$$

Example 2 (revisit)

```
a = &x
b = &y
if p then
  y = &z;
else
  y = &x
fi
c = &y
```

a: $\tau_1 = \mathbf{ref}(\tau_4 \times \bot)$

b: $\tau_2 = \mathbf{ref}(\tau_5 \times \bot)$

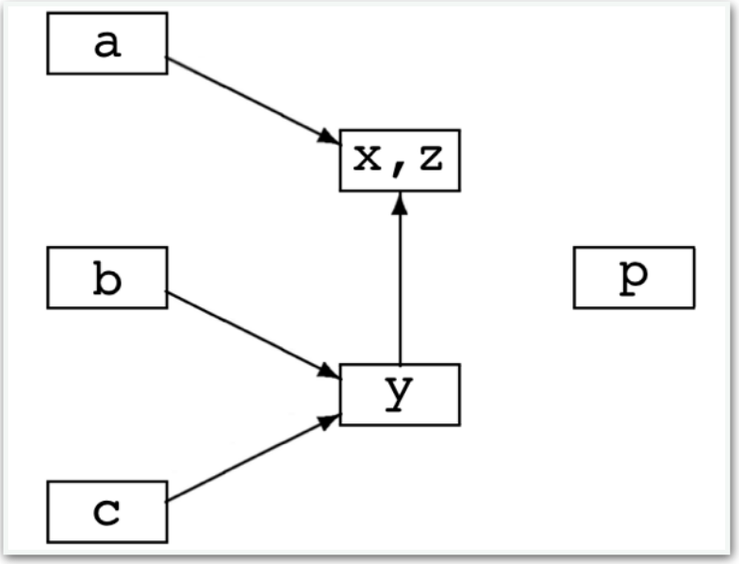
c: $\tau_3 = \mathbf{ref}(\tau_5 \times \bot)$

x: $\tau_4 = \mathbf{ref}(\bot \times \bot)$

y: $\tau_5 = \mathbf{ref}(\tau_4 \times \bot)$

z: τ_4

p: $\tau_6 = \mathbf{ref}(\bot \times \bot)$



Algorithm

- Find all pointer assignments in program
- Form points-to graph nodes for pointers to variables and functions, and variables.
- For each statement (arbitrary order)
  - construct points-to edges
  - Merge nodes (and edges) where indicated by unification constraints

Bottom Line

- Small programs - Andersen
- Large programs - Steensgaard