Analysis-Based Program Transformations

Presentation for HOPL

2/3/17

	Reaching Definitions	Live Variables	Available Expressions
Domain	Sets of definitions	Sets of variables	Sets of expressions
Direction	Forwards	Backwards	Forwards
Transfer function	$gen_B \cup (x - kill_B)$	$use_B \cup (x - def_B)$	$e\_gen_B \cup (x - e\_kill_B)$
Boundary	$\text{OUT}[\text{ENTRY}] = \emptyset$	$\text{IN}[\text{EXIT}] = \emptyset$	$\text{OUT}[\text{ENTRY}] = \emptyset$
Meet ($\wedge$)	$\cup$	$\cup$	$\cap$
Equations	$\text{OUT}[B] = f_B(\text{IN}[B])$ $\text{IN}[B] = \bigwedge_{P,pred(B)} \text{OUT}[P]$	$\text{IN}[B] = f_B(\text{OUT}[B])$ $\text{OUT}[B] = \bigwedge_{S,succ(B)} \text{IN}[S]$	$\text{OUT}[B] = f_B(\text{IN}[B])$ $\text{IN}[B] = \bigwedge_{P,pred(B)} \text{OUT}[P]$
Initialize	$\text{OUT}[B] = \emptyset$	$\text{IN}[B] = \emptyset$	$\text{OUT}[B] = U$

Figure 9.21: Summary of three data-flow problems

The Inn at the Crossings

Semantics of Flow Analysis

What does a flow analysis mean? — More precisely: a locally consistent set of annotations.

Result of flow analysis is a proposition — To optimize e to e', want to show $P \supset \mathcal{E}[\![e]\!] = \mathcal{E}[\![e']\!]$

$e \simeq e'$ oper. sym.

What proposition does a flow analysis represent?
How to distinguish correct from incorrect data flow equations?

Use propositions as types — vs Abstract interp
- Get soundness from result in type theory (free!) — harder
- Now can justify optimization — not always done
- Don't get reconstruction/analysis algorithm — this comes for free

- Read abs interp fns as conjunctive types $f : (\forall x : \hat{D})(\hat{f}(x))$ eg $\times$: pos ⇒ pos ⇒ pos, $\times$: pos ⇒ neg ⇒ neg etc
- Local consistency mirrors type rules

Examples

Strictness Analysis	$\Delta, \{\bot\}$	ordinary types (Kuo-Mishra)	
Dead Code/Useless Var Anal	Δ, U	PER types	
Ref Ct Anal	$\{0,1,\text{lots}\}$ of types	Mauriec PER types - Correctness wrt	"instrumented semantics", not std semantics
Avail Exp, Red in Strength	$x = y + z$	invariants, not just types	
Binding Time Analysis		2-level types	Justification for optimization?

Open Question
- Higher Order / Closure Analysis — This should be clearer than Abs Interp case (?)
- Data Structures / Abstract Locations

- Does this break down on the next example?
- Does it help?

Semantics of Flow Analysis

What does a flow analysis *mean*? — More precisely: a *locally consistent* set of annotations.

Result of flow analysis is a *proposition* — To optimize e to e', want to show $P \supset \mathcal{E}[\![e]\!] = \mathcal{E}[\![e']\!]$

$e \simeq e'$ opn eqn

What proposition does a flow analysis represent?
How to distinguish correct from incorrect dataflow equations?

Selective and Lightweight Closure Conversion

Mitchell Wand and Paul Steckler*

College of Computer Science
Northeastern University
360 Huntington Avenue, 161CN
Boston, MA 02115, USA
{wand,steck}@ccs.neu.edu

POPL '94

Definition 1 The following definitions are mutually referential.

1. An *occurrence closure* (i, ψ) is a pair consisting of an occurrence index and an occurrence environment.
2. An *occurrence environment* ψ is a finite map from variables to occurrence closures, where for each $x \in Dom(\psi)$, $\psi(x) = (i, \psi')$ implies $[\![i]\!]$ is a value in Λ_{in}.

$$\frac{Var\,(i)}{(i,\psi) \underset{oc}{\Longrightarrow} \psi([\![i]\!])}$$

$$\frac{Const\,(i)}{(i,\psi) \underset{oc}{\Longrightarrow} (i,\emptyset)}$$

$$\frac{Abs\,(i)}{(i,\psi) \underset{oc}{\Longrightarrow} (i,\psi)}$$

$$\frac{\begin{array}{l} Cond\,(i) \\ (i.test,\psi) \underset{oc}{\Longrightarrow} (j,\psi') \\ [\![j]\!] = \mathbf{true} \\ (i.then,\psi) \underset{oc}{\Longrightarrow} (k,\psi'') \end{array}}{(i,\psi) \underset{oc}{\Longrightarrow} (k,\psi'')}$$

$$\frac{\begin{array}{l} Cond\,(i) \\ (i.test,\psi) \underset{oc}{\Longrightarrow} (j,\psi') \\ [\![j]\!] = \mathbf{false} \\ (i.else,\psi) \underset{oc}{\Longrightarrow} (k,\psi'') \end{array}}{(i,\psi) \underset{oc}{\Longrightarrow} (k,\psi'')}$$

$$\frac{\begin{array}{l} App\,(i) \\ (i.rator,\psi) \underset{oc}{\Longrightarrow} (j,\psi'), \quad Abs\,(j) \\ (i.rand,\psi) \underset{oc}{\Longrightarrow} (k,\psi'') \\ (j.body,\psi')[[\![j.bv]\!] \mapsto (k,\psi'')] \underset{oc}{\Longrightarrow} (m,\psi''') \end{array}}{(i,\psi) \underset{oc}{\Longrightarrow} (m,\psi''')}$$

Figure 2: Rules for the occurrence evaluator

$$Var\,(i) \Longrightarrow A_\iota([\![i]\!]) = \mathcal{P}_\iota$$

$$Const\,(i) \Longrightarrow \theta_\iota = \emptyset$$

$$Abs\,(i) \Longrightarrow \begin{cases} A_{\iota.body} = A_\iota[[\![i.bv]\!] \mapsto \mathcal{P}_{\iota\ bv}] \\ \{(i, A_\iota)\} \subseteq \phi_\iota, \text{ and} \\ \theta_\iota \subseteq Dom(A_\iota) \end{cases}$$

$$App\,(i) \text{ and } Const\,(i.rator) \Longrightarrow A_{i.rand} = A_\iota$$

$$App\,(i) \text{ and } \neg\, Const\,(i.rator) \Longrightarrow \begin{cases} A_{\iota.rator} = A_{\iota.rand} = A_\iota, \\ \theta_\iota \subseteq \theta_{i.rator}, \\ \pi_{\iota.rator} = cl_\sigma \Longrightarrow \lceil \sigma \rceil \subseteq \theta_{\iota.rator}, \text{ and} \\ \forall (j, B) \in \phi_{\iota.rator} \\ \qquad \begin{cases} Abs\,(j), \\ \mathcal{P}_{\iota.rand} \leq \mathcal{P}_{\jmath.bv}, \\ \mathcal{P}_{\jmath.body} \leq \mathcal{P}_\iota, \\ \theta_{\jmath.bv} \subseteq \theta_{\iota.rator}, \text{ and} \\ [\![j.bv]\!] \notin \theta_{\jmath.bv} \cup \theta_\iota \end{cases} \end{cases}$$

$$Cond\,(i) \Longrightarrow \begin{cases} A_{\iota\ test} = A_{i.then} = A_{\iota.else} = A_\iota, \\ \mathcal{P}_{\iota.then} \leq \mathcal{P}_\iota, \text{ and} \\ \mathcal{P}_{\iota.else} \leq \mathcal{P}_\iota, \end{cases}$$

Figure 3: Local Consistency Conditions for Annotations

$$Var\,(i) \Longrightarrow \Phi(i) = [\![i]\!]$$

$$Const\,(i) \Longrightarrow \Phi(i) = [\![i]\!]$$

$$Abs\,(i) \wedge \pi_1 = id \Longrightarrow \Phi(i) = \lambda x.\Phi(i.body)$$

$$Abs\,(i) \wedge \boxed{\pi_1 = cl_\sigma} \Longrightarrow \Phi(i) = \begin{array}{l} [(\lambda e \vec{v} x.\, \mathbf{destr}\ e\ (\lambda \vec{u}.\Phi(i.body))), [\vec{u}]] \\ \quad \text{where } e \text{ fresh} \\ \quad \boxed{\begin{array}{l} \text{and } \vec{v} = \sigma \\ \text{and } \lceil \vec{u} \rceil = FV([\![i]\!]) - \lceil \sigma \rceil \end{array}} \end{array}$$

$$App\,(i) \wedge Const\,(i.rator) \Longrightarrow \Phi(i) = \Phi(i.rator)\ \Phi(i.rand)$$

$$App\,(i) \wedge \neg\, Const\,(i.rator) \wedge \pi_{1\ rator} = id \Longrightarrow \Phi(i) = \Phi(i.rator)\ \Phi(i.rand)$$

$$App\,(i) \wedge \neg\, Const\,(i.rator) \wedge \pi_{1\ rator} = cl_\sigma \Longrightarrow \Phi(i) = \boxed{\begin{array}{l} \mathbf{app}\ \Phi(i.rator)\ v_1\ \ldots\ v_n\ \Phi(i.rand) \\ \quad \text{where } \vec{v} = \sigma \end{array}}$$

$$Cond\,(i) \Longrightarrow \Phi(i) = \mathbf{if}\ \Phi(i.test)\ \mathbf{then}\ \Phi(i.then)\ \mathbf{else}\ \Phi(i.else)$$

$$\Phi(i, \psi) \underset{t}{\Longrightarrow} \Phi(j, \psi')$$

Theorem 3 (Correctness) *Let Γ be a monovariant locally consistent annotation map, and Π the protocol assignment defined by $\forall i, \Pi(i) = \pi_i$. Let ψ be an occurrence environment such that $\psi \overset{\Pi}{\models} A_i$. If*

$$(i, \psi) \underset{oc}{\Longrightarrow} (j, \psi')$$

then

$$\hat{\Phi}(i, \psi) \underset{t}{\Longrightarrow} \hat{\Phi}(j, \psi')$$

Proof: (Sketch) The proof is by induction on the size of the derivation that $(i, \psi) \underset{oc}{\Longrightarrow} (j, \psi')$. The soundness theorem is used to guarantee that for each invocation of the induction hypothesis for an occurrence closure (k, ψ''), the needed condition $\psi'' \overset{\Pi}{\models} A_k$ is satisfied. ∎

Order-of-evaluation Analysis for Destructive Updates in Strict Functional Languages with Flat Aggregates

A.V.S. Sastry, William Clinger, and Zena Ariola
Department of Computer Science
University of Oregon
Eugene, OR 97403
email: [sastry, will, ariola]@cs.uoregon.edu

FPCA ‘93

Set Constraints for Destructive Array Update Optimization

Mitchell Wand and William D. Clinger

College of Computer Science
Northeastern University
Boston, MA 02115

IEEE Conference on Computer Languages 1998
long version in JFP 2001

$$\langle \alpha,\ \rho,\ \theta{:}\ \phi(v_1, \ldots, v_{i-1}, E_i, E_{i+1}, \ldots, E_n),\ K\rangle$$
$$\quad \rightarrow \quad \langle \alpha.i,\ \rho,\ E_i,\ \langle \alpha,\ \rho,\ \theta{:}\ \phi(v_1, \ldots, v_{i-1}, [\ ], E_{i+1|, \ldots, E_n)} K,\ \rangle\rangle$$

[push]

$$\langle \alpha,\ \rho,\ v,\ \langle \alpha',\ \rho',\ R,\ K\rangle\rangle$$
$$\quad \rightarrow \quad \langle \alpha',\ \rho',\ R[v],\ K\rangle$$

[return]

Definition 5 (Live Location)

- *No location is live in* **halt**.
- l *is live in* $\langle \alpha, \rho, R, K \rangle$ *iff either:*

  1. l *occurs in* R, *or*
  2. *there exists* $x \in \mathrm{fv}(R)$ *such that* $\rho(x) = l$, *or*
  3. l *is live in* K.

We can now state the soundness condition for a live variable analysis $\mathcal{L}[\![-]\!]$.

Definition 6 (Live Variable Analysis)
A live variable analysis $\mathcal{L}[\![-]\!]$ *is a map from expression labels* θ *to sets of variables.* $\mathcal{L}[\![-]\!]$ *is* sound *iff for each label* θ, $\mathcal{L}[\![\theta]\!]$ *is a set of variables such that for all reachable configurations of the form* $\langle \alpha, \rho, \theta{:}\, T, K, \Sigma \rangle$, $\rho(x)$ *live in* K *implies* $x \in \mathcal{L}[\![\theta]\!]$.

Theorem 9 (Correctness of Transformation)
If $\mathcal{L}[\![-]\!]$ is a sound live variable analysis, and $\langle \alpha_0, \rho_0, F_0, \mathbf{halt}, \Sigma_0 \rangle$ is an initial configuration, then

$$\langle \alpha_0, \rho_0, F_0, \mathbf{halt}, \Sigma_0 \rangle \rightarrow^n \langle \mathbf{halted}, v \rangle$$

if and only if

$$\langle \alpha_0, \rho_0, F_0^*, \mathbf{halt}, \Sigma_0 \rangle \Rightarrow^n \langle \mathbf{halted}, v \rangle$$

F_0^* is the transformed version of F_0, based on $\mathcal{L}$

Definition 12 (Alias Analysis) *An* alias set $\mathcal{A}$ *is a subset of* $Var \times Var$. *For* $S \subseteq Var$, *define* $\mathcal{A} \star S = \{x \mid (x, y) \in \mathcal{A} \wedge y \in S\}$.

Each alias set $\mathcal{A}$ induces a proposition $\langle \alpha, \rho, G, K \rangle \models \mathcal{A}$ on configurations of the environment semantics. Again, lack of space prevents us from giving a formal definition. Informally, however,

Definition 13 (Soundness of $\mathcal{A}$)
$\mathcal{A}$ is a sound alias analysis iff $\langle \alpha, \rho, G, K \rangle \models \mathcal{A}$ *for every reachable configuration* $\langle \alpha, \rho, G, K \rangle$.

Theorem 14 (Correctness of $\mathcal{A}$) *If* $\mathcal{P}[\![-]\!]$ *is a sound propagation analysis and* $\mathcal{A}$ *satisfies the constraints A1–A2, then* $\mathcal{A}$ *is a sound alias analysis.*