Static Dependent Costs for Estimating Execution Time

Brian Reistad and David K. Gifford
LFP 1994

Sukyoung Ryu

ROPAS, KAIST

April 19, 2000

Type and Language

Id	x			
Descr	d	$::=$	$\tau \mid c \mid n$	
Type	τ	$::=$	$x_t \mid \texttt{float} \mid \texttt{bool}$	type variable, float, boolean
		$\mid$	$\forall\{x_i\}.\tau$	polymorphic type
		$\mid$	$\tau_1 \xrightarrow{c} \tau_2$	abstraction type
		$\mid$	$\texttt{num}\ n$	number type
		$\mid$	$\texttt{list}\ \tau\ n$	list type
Cost	c	$::=$	x_c	cost variable
		$\mid$	$\texttt{C\_num} \mid \texttt{C\_lambda} \mid \cdots$	symbolic constant
		$\mid$	$\texttt{long}$	
		$\mid$	$c_1 \texttt{+} c_2 \mid n \texttt{*} c \mid \texttt{max}\, c_1\, c_2$	
Size	n	$::=$	x_n	size variable
		$\mid$	$1 \mid 2 \mid 3 \mid \cdots \mid \texttt{long}$	
		$\mid$	$n_1 \texttt{+} n_2 \mid n_1 \texttt{*} n_2 \mid \texttt{max}\, n_1\, n_2$	
Expr	e	$::=$	$x \mid i \mid \texttt{true} \mid \texttt{false}$	variable, number, boolean
		$\mid$	$\lambda x.e$	immediate function
		$\mid$	$\texttt{fix}\ f\ \lambda x.e$	recursive function
		$\mid$	$e_1\ e_2$	application
		$\mid$	$\texttt{let}\ x \texttt{=} e_1\ \texttt{in}\ e_2$	let binding
		$\mid$	$\texttt{if}\ e_1\ e_2\ e_3$	conditional

Dynamic Semantics

$$\sigma \vdash i \rightarrow i, \texttt{C\_num} \quad \texttt{(num)}$$

$$\frac{\sigma(x) = v}{\sigma \vdash x \rightarrow v, \texttt{C\_var}} \; \texttt{(var)}$$

$$\sigma \vdash \lambda x.e \rightarrow \langle x, e, \sigma, [\,]\rangle, \texttt{C\_lambda} \quad \texttt{(lambda)}$$

$$\sigma \vdash \texttt{fix}\ f\ \lambda x.e \rightarrow \langle x, e, \sigma, [f \mapsto \langle x, e, \sigma, [\,]\rangle]\rangle, \texttt{C\_rec} \quad \texttt{(rec)}$$

$$\frac{\begin{array}{c}\sigma \vdash e_1 \rightarrow \langle x, e', \sigma', \sigma''\rangle, c_1 \qquad \sigma \vdash e_2 \rightarrow v', c_2 \\ \sigma'[x \mapsto v'] :: \mathit{Rec}(\sigma'') \vdash e' \rightarrow v, c\end{array}}{\sigma \vdash e_1\ e_2 \rightarrow v, \texttt{C\_call} + c + c_1 + c_2} \; \texttt{(call)}$$

$$\frac{\sigma \vdash e_1 \rightarrow v_1, c_1 \qquad \sigma[x \mapsto v_1] \vdash e_2 \rightarrow v, c}{\sigma \vdash \texttt{let}\ x\ \texttt{=}\ e_1\ \texttt{in}\ e_2 \rightarrow v, \texttt{C\_let} + c_1 + c} \; \texttt{(let)}$$

$$\frac{\sigma \vdash e_1 \rightarrow \texttt{true}, c_1 \qquad \sigma \vdash e_2 \rightarrow v, c}{\sigma \vdash \texttt{if}\ e_1\ e_2\ e_3 \rightarrow v, \texttt{C\_if} + c_1 + c} \; \texttt{(if-true)}$$

$$\frac{\sigma \vdash e_1 \rightarrow \texttt{false}, c_1 \qquad \sigma \vdash e_3 \rightarrow v, c}{\sigma \vdash \texttt{if}\ e_1\ e_2\ e_3 \rightarrow v, \texttt{C\_if} + c_1 + c} \; \texttt{(if-false)}$$

$$\mathit{Rec}([f \mapsto \langle x, e, \sigma, [\,]\rangle]) = [f \mapsto \langle x, e, \sigma, [f \mapsto \langle x, e, \sigma, [\,]\rangle]\rangle]$$
$$\mathit{Rec}([\,]) = [\,]$$

Static Semantics (I)

$$\frac{i \sqsubseteq_{size} n}{\Gamma \vdash i : \texttt{num}\ n / \texttt{C\_num}}\ (\texttt{num})$$

$$\frac{\Gamma(x) = \forall\{x_i\}.\tau \qquad [d'_i/x_i]\tau \sqsubseteq_{type} \tau'}{\Gamma \vdash x : \tau' / \texttt{C\_var}}\ (\texttt{var})$$

$$\frac{\Gamma[x\!: \tau_1] \vdash e : \tau_2 / c \qquad c \sqsubseteq_{cost} c'}{\Gamma \vdash \lambda x.e : \tau_1 \stackrel{c'}{\rightarrow} \tau_2 / \texttt{C\_lambda}}\ (\texttt{lambda})$$

$$\frac{\Gamma[f\!: \tau_1 \stackrel{c'}{\rightarrow} \tau_2][x\!: \tau_1] \vdash e : \tau_2 / c \qquad c \sqsubseteq_{cost} c'}{\Gamma \vdash \texttt{fix}\ f\ \lambda x.e : \tau_1 \stackrel{c'}{\rightarrow} \tau_2 / \texttt{C\_rec}}\ (\texttt{rec})$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \stackrel{c}{\rightarrow} \tau_2 / c_1 \qquad \Gamma \vdash e_2 : \tau_1 / c_2 \qquad \tau_2 \sqsubseteq_{type} \tau'_2}{\Gamma \vdash e_1\ e_2 : \tau'_2 / \texttt{C\_call} \texttt{+} c_1 \texttt{+} c_2 \texttt{+} c}\ (\texttt{call})$$

$$\frac{\begin{array}{c} e_1 \quad \text{non-expansive} \\ \Gamma \vdash e_1 : \tau_1 / c_1 \qquad \Gamma[x : Gen(\tau_1, \Gamma)] \vdash e_2 : \tau_2 / c_2 \end{array}}{\Gamma \vdash \texttt{let}\ x\ \texttt{=}\ e_1\ \texttt{in}\ e_2 : \tau_2 / \texttt{C\_let} \texttt{+} c_1 \texttt{+} c_2}\ (\texttt{let})$$

$$\frac{\Gamma \vdash e_1 : \texttt{bool} / c_1 \qquad \Gamma \vdash e_2 : \tau / c_2 \qquad \Gamma \vdash e_3 : \tau / c_3}{\Gamma \vdash \texttt{if}\ e_1\ e_2\ e_3 : \tau / \texttt{C\_if} \texttt{+} c_1 \texttt{+} (\texttt{max}\, c_2\, c_3)}\ (\texttt{if})$$

$$Gen(\tau, \Gamma) = \forall\{x_i\}.\tau \qquad \text{where } \{x_i\} = FV(\tau) \setminus FV(\Gamma)$$

Static Semantics (II)

- Subtyping Relation

$$\frac{n \sqsubseteq_{size} n'}{\texttt{num}\ n \sqsubseteq_{type} \texttt{num}\ n'}\ \texttt{(subtype-num)}$$

$$\frac{n \sqsubseteq_{size} n'}{\texttt{list}\ \tau\ n \sqsubseteq_{type} \texttt{list}\ \tau\ n'}\ \texttt{(subtype-list)}$$

$$\frac{c \sqsubseteq_{cost} c' \qquad \tau_2' \sqsubseteq_{type} \tau_2 \qquad \tau_1 \sqsubseteq_{type} \tau_1'}{\tau_1 \xrightarrow{c} \tau_2 \sqsubseteq_{type} \tau_1' \xrightarrow{c'} \tau_2'}\ \texttt{(subtype-arrow)}$$

- Primitive Operator Types

$$\begin{array}{rcl}
\texttt{+} & : & \forall\{n_1, n_2\}.\texttt{num}\ n_1 \to \texttt{num}\ n_2 \xrightarrow{\texttt{C_+}} \texttt{num}\ (n_1 + n_2) \\
\texttt{*} & : & \forall\{n_1, n_2\}.\texttt{num}\ n_1 \to \texttt{num}\ n_2 \xrightarrow{\texttt{C_*}} \texttt{num}\ (n_1 * n_2) \\
\texttt{-} & : & \forall\{n_1, n_2\}.\texttt{num}\ n_1 \to \texttt{num}\ n_2 \xrightarrow{\texttt{C_-}} \texttt{num}\ n_1 \\
\texttt{map} & : & \forall\{t_1, t_2, c, len\}.(\tau_1 \xrightarrow{c} \tau_2) \to \texttt{list}\ \tau_1\ len \to \texttt{list}\ \tau_2\ len
\end{array}$$

$$\texttt{C\_map-overhead} + (len * (\texttt{C\_map-per-elem} + c))$$

Cost Reconstruction (I)

$\mathcal{R}$: $\mathit{TypEnv} \times \mathit{Expr} \to \mathit{Type} \times \mathit{Cost} \times \mathit{Subst} \times \mathit{Constraints}$

$$\mathcal{R}(\Gamma, i) \;=\; (\mathtt{num}\; x_n, \mathtt{C\_num}, [\,], \{i \sqsubseteq_{size} x_n\}),\ \text{new } x_n$$

$$\begin{array}{lll}\mathcal{R}(\Gamma, x) & = & \text{let}\ \ \forall\{x_i\}.(\tau, K) = \Gamma(x) \\ & & \qquad S = [y_i/x_i],\ \text{new } y_i \\ & & \qquad (\tau', K') = \mathit{lift\_type}(S\tau) \\ & & \text{in}\ \ (\tau', \mathtt{C\_var}, [\,], SK \cup K')\end{array}$$

$$\begin{array}{l}\mathcal{R}(\Gamma, \lambda(x : \tau_s).e) = \\ \qquad\qquad \text{let}\ \ \tau' = \mathit{newtype}(\tau_s) \\ \qquad\qquad \qquad (\tau, c, S, K) = \mathcal{R}(\Gamma[x : \tau'], e) \\ \qquad\qquad \text{in}\ \ (S\tau' \xrightarrow{x_c} \tau, \mathtt{C\_lambda}, S, K \cup \{c \sqsubseteq_{cost} x_c\}),\ \text{new } x_c\end{array}$$

$$\begin{array}{l}\mathcal{R}(\Gamma, \mathtt{fix}\ f\ \lambda(x : \tau_x).(e : \tau_1)) = \\ \qquad\qquad \text{let}\ \ \tau_x' = \mathit{newtype}(\tau_x) \\ \qquad\qquad \qquad \tau_1' = \mathit{newtype}(\tau_1) \\ \qquad\qquad \qquad \Gamma' = \Gamma[f : \tau_x' \xrightarrow{x_c} \tau_1'][x : \tau_x'],\ \text{new } x_c \\ \qquad\qquad \qquad (\tau_2, c_2, S_2, K_2) = \mathcal{R}(\Gamma', e) \\ \qquad\qquad \qquad S = \mathcal{U}(\tau_2, S_2\tau_1') \\ \qquad\qquad \text{in}\ \ S(S_2(\tau_x' \xrightarrow{x_c} \tau_1'), \mathtt{C\_rec}, S_2, K_2 \cup \{c_2 \sqsubseteq_{cost} S_2 x_c\})\end{array}$$

Cost Reconstruction (II)

$$
\begin{array}{lll}
\mathcal{R}(\Gamma, e_1\ e_2) & = & \text{let } (\tau_1, c_1, S_1, K_1) = \mathcal{R}(\Gamma, e_1) \\
& & \quad (\tau_2, c_2, S_2, K_2) = \mathcal{R}(S_1\Gamma, e_2) \\
& & \quad S = \mathcal{U}(S_2\tau_1, \tau_2 \xrightarrow{x_c} x_t), \text{ new } x_c, x_t \\
& & \quad (\tau', K') = \mathit{lift\_type}(Sx_t) \\
& & \text{in } (\tau', S(\texttt{C\_call} + S_2c_1 + c_2 + x_c), SS_2S_1, \\
& & \quad SS_2K_1 \cup SK_2 \cup K')
\end{array}
$$

$\mathcal{R}(\Gamma, \texttt{let}\ x = e_1\ \texttt{in}\ e_2) =$

if e expansive then

$$
\begin{array}{lll}
& = & \text{let } (\tau_1, c_1, S_1, K_1) = \mathcal{R}(\Gamma, e_1) \\
& & \quad (\tau_2, c_2, S_2, K_2) = \mathcal{R}(S_1\Gamma[x : \tau_1], e_2) \\
& & \text{in } (\tau_2, S(\texttt{C\_let} + S_2c_1 + c_2), S_2S_1, S_2K_1 \cup K_2)
\end{array}
$$

else (e non-expansive)

$$
\begin{array}{lll}
& = & \text{let } (\tau_1, c_1, S_1, K_1) = \mathcal{R}(\Gamma, e_1) \\
& & \quad \{x_1, \cdots, x_n\} = (FV(\tau_1) \cup FV(K_1)) \setminus FV(S_1\Gamma) \\
& & \quad (\tau_2, c_2, S_2, K_2) = \mathcal{R}(S_1\Gamma[x : \forall\{x_i\}.(\tau_1, K_1)], e_2) \\
& & \text{in } (\tau_2, S(\texttt{C\_let} + S_2c_1 + c_2), S_2S_1, S_2K_1 \cup K_2)
\end{array}
$$

$\mathcal{R}(\Gamma, \texttt{if}\ e_1\ e_2\ e_3) =$

$$
\begin{array}{lll}
& = & \text{let } (\tau_1, c_1, S_1, K_1) = \mathcal{R}(\Gamma, e_1) \\
& & \quad (\tau_2, c_2, S_2, K_2) = \mathcal{R}(S_1\Gamma, e_2) \\
& & \quad (\tau_3, c_3, S_3, K_3) = \mathcal{R}(S_2S_1\Gamma, e_3) \\
& & \quad S = \mathcal{U}(S_3S_2\tau_1, \texttt{bool}) \\
& & \quad S' = \mathcal{U}(SS_3\tau_2, S\tau_3) \\
& & \text{in } S'S(\tau_3, \texttt{C\_if} + S_3S_2c_1 + (\texttt{max}\ (S_3c_2)\ c_3), S_3S_2S_1, \\
& & \qquad S_3S_2K_1 \cup S_3K_2 \cup K_3)
\end{array}
$$

Cost Reconstruction (III)

- $\mathit{lift\_type}$: $\mathit{Type} \rightarrow \mathit{Type} \times \mathit{Constraints}$
  $\mathit{lift\_type}(\tau) = (\tau', \{\tau \sqsubseteq_{type} \tau'\})$

- $\mathit{newtype}$: $\mathit{Type}^{\setminus\{c,n\}} \rightarrow \mathit{Type}$
  $\mathit{newtype}(\mathtt{num}) = \mathtt{num}\ x_n,$ new x_n
  $\mathit{newtype}(\mathtt{list}\ \tau) = \mathtt{list}\ (\mathit{newtype}(\tau))\ x_n, ,$ new x_n
  $\mathit{newtype}(\tau_1 \rightarrow \tau_2) = \mathit{newtype}(\tau_1) \xrightarrow{x_c} \mathit{newtype}(\tau_2),$ new x_c

- $\mathcal{U}$: $\mathit{Type} \times \mathit{Type} \rightarrow \mathit{Subst}$

- Constraint Solving: least fixpoint algorithm

Discussion

- Correctness

  “We believe the reconstruction algorithm is sound even though we have not given a formal proof.”

- Limitation
  - Every recursive function takes `long` time.