1 CoPylot

1.1 Grammar

The grammar of the language to be analyzed appears in Figure 2.

We assume throughout the rest of this document that a fixed program $\hat{S}$ is under analysis. *(TODO: describe here the idea of a bijection between labels and statements in this fixed program. – ZP)*

1.2 Control Flow

The grammar of control flow graphs appears in Figure 3. *(Discuss construction of initial graph. – ZP)*

We write $\hat{o} \overset{?}{\blacktriangleleft} \hat{o}'$ to denote $(\hat{o} \blacktriangleleft \hat{o}' \in \hat{G}$ when $\hat{G}$ is understood from context. Likewise, we write $\hat{o} \overset{?}{\lll} \hat{o}'$ to denote $(\hat{o} \lll \hat{o}' \in \hat{G}$ when $\hat{G}$ is understood from context.

We define a relation $\longrightarrow^1$ to perform control flow graph closure.

Definition 1.1. *Let $\hat{G} \longrightarrow^1 \hat{G}'$ be the least relation satisfying the rules appearing in Figure 4. Throughout these rules, the predicates $\overset{?}{\lll}$ and $\overset{?}{\blacktriangleleft}$ refer to graph $\hat{G}$.*

1.3 Value Lookup

The value lookup function uses the additional grammar in Figure 5.

Definition 1.2. *Given a control-flow graph $\hat{G}$, let $\hat{G}(\hat{o}_0, \hat{K})$ be the function returning the least set $\hat{V}$ which satisfies the following conditions:*

1. ***Value Manipulation***

   (a) $\boxed{\text{Result}}$
   If $\hat{K} = [\hat{v}]$, then $\hat{v} \in \hat{V}$.

2. ***Variable Lookup***

   (a) $\boxed{\text{Value Discovery}}$
   If $\hat{o}_1 \overset{?}{\lll} \hat{o}_0$, $\hat{o}_1 = \hat{\ell}_1 : \hat{\ell}_2 : \hat{x} = \hat{v}$, and $\hat{K} = [\hat{x}] \,||\, \hat{K}'$, then $\hat{G}(\hat{o}_1, [\hat{v}] \,||\, \hat{K}') \subseteq \hat{V}$.

   (b) $\boxed{\text{Value Skip}}$
   If $\hat{o}_1 \overset{?}{\lll} \hat{o}_0$, $\hat{o}_1 = \hat{\ell}_1 : \hat{\ell}_2 : \hat{x}' = \hat{v}$, $\hat{K} = [\hat{x}] \,||\, \hat{K}'$, and $\hat{x} \neq \hat{x}'$, then $\hat{G}(\hat{o}_1, \hat{K}) \subseteq \hat{V}$.

   (c) $\boxed{\text{Value Aliasing}}$
   If $\hat{o}_1 \overset{?}{\lll} \hat{o}_0$, $\hat{o}_1 = \hat{\ell}_1 : \hat{\ell}_2 : \hat{x} = \hat{x}'$, and $\hat{K} = [\hat{x}] \,||\, \hat{K}'$, then $\hat{G}(\hat{o}_1, [\hat{x}'] \,||\, \hat{K}) \subseteq \hat{V}$.

$$
\begin{array}{rcll}
\overset{*}{\ell} & ::= & \ell \mid \textsc{End} & \textit{labels} \\
T & ::= & [t, \ldots] & \textit{stack} \\
t & ::= & \overset{*}{\ell} \times S & \textit{stack frame} \\
S & ::= & [s, \ldots] & \textit{programs} \\
d & ::= & x = e \mid \texttt{def}\ x(x, \ldots) = \{S\} \mid \texttt{return}\ x \mid \texttt{goto}\ \ell \mid \texttt{goto}\ \ell\ \texttt{if}\ \texttt{not}\ x & \textit{directives} \\
B & ::= & \{x \mapsto m, \ldots\} & \textit{bindings} \\
H & ::= & \{m \mapsto v, \ldots\} & \textit{heap} \\
v & ::= & \mathbb{Z} \mid \langle m,\ \texttt{def}\ (x) \to S\rangle \mid \langle m, m,\ \texttt{def}\ (x) \to S\rangle \mid [m, \ldots] \mid (m, \ldots) \mid B & \textit{values} \\
e & ::= & \mathbb{Z} \mid x \mid x(x, \ldots) \mid [x, \ldots] \mid (x, \ldots) & \textit{expressions} \\
P & ::= & m \mapsto m & \textit{parental maps}
\end{array}
$$

Literal Assignment

$$
\frac{\begin{array}{c} S(\ell) = \ell : \ell' : x = v \\ H' = H[m \mapsto v] \qquad m \notin H \qquad \textsc{Bind}(H', m_0, x, m) = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{[\langle \ell, S\rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S\rangle] \,||\, T, H'', P, m_0}
$$

Name Assignment

$$
\frac{\begin{array}{c} S(\ell) = \ell : \ell' : x_1 = v_2 \\ m = \textsc{Lookup}(m_0, P, H, x_2) \qquad \textsc{Bind}(H, m_0, x_1, m) = H' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{[\langle \ell, S\rangle] \,||\, T, H, oparent, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S\rangle] \,||\, T, H', oparent, m_0}
$$

List Assignment

$$
\frac{\begin{array}{c} S(\ell) = \ell : \ell' : x = [x_1, \ldots, x_n] \\ \forall i \in \{1, \ldots, n\}, m_i = \textsc{Lookup}(m_0, P, H, x_i) \qquad v = [m_1, \ldots, m_2] \\ H' = H[m' \mapsto v] \qquad m' \notin H \qquad \textsc{Bind}(H', m_0, x, m') = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{\langle \ell, S\rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S\rangle] \,||\, T, H'', P, m_0}
$$

Tuple Assignment

$$
\frac{\begin{array}{c} S(\ell) = \ell : \ell' : x = (x_1, \ldots, x_n) \\ \forall i \in \{1, \ldots, n\}, m_i = \textsc{Lookup}(m_0, P, H, x_i) \qquad v = (m_1, \ldots, m_2) \\ H' = H[m' \mapsto v] \qquad m' \notin H \qquad \textsc{Bind}(H', m_0, x, m') = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{\langle \ell, S\rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S\rangle] \,||\, T, H'', P, m_0}
$$

Function Definition

$$
\frac{\begin{array}{c} S(\ell) = \ell : \ell' :\ \texttt{def}\ x_1(x_2) = \{S\} \qquad v = \langle \eta,\ \texttt{def}\ (x_2) \to S\rangle \\ H' = H[m \mapsto v] \qquad m \notin H \qquad \textsc{Bind}(H', m_0, x_1, m) = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{[\langle \ell, S\rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S\rangle] \,||\, T, H', E', P, m_0}
$$

Figure 1: Operation Semantics

Function Call

$$\frac{\begin{array}{c} S(\ell) = \ell : \ell' : x_1 = x_2(x_3) \\ m = \textsc{Lookup}(m_0, P, E, x_2) \qquad H[m] = \langle m_0', \ \texttt{def} \ (x_4) \mapsto S' \rangle \\ m_0'' \notin E \qquad m' = \textsc{Lookup}(m_0, P, E, x_3) \qquad B' = B[x_4 \mapsto m'] \\ E' = E[m_0'' \mapsto B'] \qquad P' = P \cup \{m_0'' \mapsto m_0'\} \qquad S' = [\ell'' : \ell''' : d] \,||\, S'' \end{array}}{[\langle \ell, S \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S' \rangle, \langle l, S \rangle] \,||\, T, H', E', P', m_0''}$$

Pass

$$\frac{S(\ell) = \ell : \ell' : pass \qquad \ell \overset{s}{\blacktriangleleft} \overset{*''}{\ell}}{[\langle \ell, S \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S \rangle] \,||\, T, H, E, P, m_0}$$

Return

$$\frac{S(\ell) = \ell : \ell' : \ \texttt{return} \qquad T = [t, \langle \ell'', S' \rangle] \,||\, T' \qquad m_0' = P[m_0] \qquad \ell'' \overset{s'}{\blacktriangleleft} \overset{*'''}{\ell}}{[t, \langle \ell'', S' \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S' \rangle] \,||\, T', H, E, P, m_0'}$$

Return with Arguments

$$\frac{\begin{array}{c} S(\ell) = \ell : \ell' : \ \texttt{return} \ x \qquad T = [t, \langle \ell'', S' \rangle] \,||\, T' \\ m = \textsc{Lookup}(m_0, P, E, x) \qquad S'(\ell'') = \ell'' : \ell''' : x_1 = x_2(x_3) \qquad m_0' = P[m_0] \\ B = E(m_0') \qquad B' = B[x_1 \mapsto m] \qquad E' = E[m_0' \mapsto B'] \qquad \ell'' \overset{s'}{\blacktriangleleft} \overset{*''''}{\ell} \end{array}}{[t, \langle \ell'', S' \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S' \rangle] \,||\, T, H, E', P, m_0'}$$

Goto

$$\frac{S(\ell) = \ell : \ell' : \ \texttt{goto} \ \ell'' \qquad S = [\ell'' : \ell''' : d] \,||\, S'}{[\langle \ell, S \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S \rangle] \,||\, T, H, E, P, m_0}$$

GotoIfNot

$$\frac{\begin{array}{c} S(\ell) = \ell : \ell' : \ \texttt{goto} \ \ell'' \ \texttt{if not} \ x \\ m = \textsc{Lookup}(m_0, P, E, x) \qquad H[m] = \textsc{False} \qquad S = [\ell'' : \ell''' : d] \,||\, S' \end{array}}{[\langle \ell, S \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell'', S \rangle] \,||\, T, H, E, P, m_0}$$

Name Statement

$$\frac{S(\ell) = \ell : \ell' : e \qquad \forall x \in e \exists B[x] \qquad \ell \overset{s'}{\blacktriangleleft} \overset{*''}{\ell}}{[\langle \ell, S \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell', S \rangle] \,||\, T, H, E, P, m_0}$$

End of Function

$$\frac{T = [\langle \textsc{End}, S \rangle, \langle \ell, S' \rangle] \,||\, T' \qquad m_0' = P[m_0] \qquad \ell \overset{s'}{\blacktriangleleft} \overset{*'}{\ell}}{[\langle \textsc{End}, S \rangle, \langle \ell, S' \rangle] \,||\, T, H, E, P, m_0 \longrightarrow^1 [\langle \ell', S' \rangle] \,||\, T', H, E, P, m_0'}$$

End of Program

$$\frac{T = [(\textsc{End}, t)]}{T, H, E, P, m_0 \longrightarrow^1 [], H, E, P, m_0'}$$

Figure 1: Operation Semantics (cont.)

Figure 1: Operation Semantics (cont.)

$$
\begin{array}{rcll}
\hat{S} & ::= & [\hat{s}, \ldots] & \textit{abstract programs} \\
\hat{s} & ::= & \hat{\ell} : \hat{\ell} : \hat{d} & \textit{abstract statements} \\
\hat{d} & ::= & \hat{x} = \hat{v} \mid \hat{x} = \hat{x} & \textit{abstract directives} \\
\hat{v} & ::= & \texttt{int}^{+} \mid \texttt{int}^{-} \mid \texttt{int}^{0} & \textit{abstract values} \\
\hat{x} & & & \textit{abstract variables} \\
\hat{\ell} & & & \textit{abstract labels}
\end{array}
$$

Figure 2: Normalized Python Language Grammar

$$
\begin{array}{rcll}
\hat{G} & ::= & \{\hat{g}, \ldots\} & \textit{control flow graphs} \\
\hat{g} & ::= & \hat{o} \blacktriangleleft \hat{o} \mid \hat{o} \ll \hat{o} & \textit{control flow graph edge} \\
\hat{o} & ::= & \textsc{Start} \mid \textsc{End} \mid \hat{s} & \textit{control flow graph nodes}
\end{array}
$$

Figure 3: Control Flow Graph Grammar

$$
\textsc{Lexical Start}\quad
\frac{\textsc{Start} \overset{?}{\blacktriangleleft} \hat{o}}{\hat{G} \longrightarrow^{1} \hat{G} \cup \{\textsc{Start} \ll \hat{o}\}}
\qquad\qquad
\textsc{Literal Assignment}\quad
\frac{\hat{o}_1 = (\hat{x} = \hat{v}) \qquad \hat{o}_1 \overset{?}{\blacktriangleleft} \hat{o}_2}{\hat{G} \longrightarrow^{1} \hat{G} \cup \{\hat{o}_1 \ll \hat{o}_2\}}
$$

$$
\textsc{Variable Accessible}\quad
\frac{\hat{o}_1 = (\hat{x} = \hat{x}') \qquad \hat{o}_1 \overset{?}{\blacktriangleleft} \hat{o}_2 \qquad \hat{v} \in \hat{G}(\hat{o}_1, [\hat{x}']) \qquad \hat{v} \neq \textsc{Undefined}}{\hat{G} \longrightarrow^{1} \hat{G} \cup \{\hat{o}_1 \ll \hat{o}_2\}}
$$

Figure 4: Control Flow Graph Closure

$$
\begin{array}{rcll}
\hat{K} & ::= & [\hat{k}, \ldots] & \textit{lookup stacks} \\
\hat{k} & ::= & \hat{x} \mid \hat{v} \mid \textsc{Capture}(\mathbb{N}) \mid \textsc{Jump}(\hat{o}) & \textit{lookup stack elements}
\end{array}
$$

Figure 5: Value Lookup Grammar