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 1.5. *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 1.6. *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 $\hat{G}(\hat{o}_1, [\hat{x}'] \,||\, \hat{K}) \subseteq \hat{V}$.

$$
\begin{array}{lll}
x & ::= & \textsc{AlphaNumeric} \mid \star\textsc{AlphaNumeric} \\
\overset{*}{\ell} & ::= & \ell \mid \textsc{End} \\
T & ::= & [t, \ldots] \\
t & ::= & \overset{*}{\ell} \times S \\
S & ::= & [s, \ldots] \\
d & ::= & x = e \mid \texttt{def}\ x(x, \ldots) = \{S\} \mid \texttt{return}\ x \mid \texttt{goto}\ \ell \mid \texttt{goto}\ \ell\ \texttt{if not}\ x \\
B & ::= & \{x \mapsto m, \ldots\} \\
H & ::= & \{m \mapsto v, \ldots\} \\
v & ::= & \mathbb{Z} \mid \langle m, \texttt{def}\ (x) \to S\rangle \mid \langle m, m, \texttt{def}\ (x) \to S\rangle \mid [m, \ldots] \mid (m, \ldots) \mid B \mid \mathfrak{F} \mid \langle m, \mathfrak{M}\rangle \\
e & ::= & v \mid x \mid x(x, \ldots) \mid [x, \ldots] \mid (x, \ldots) \\
P & ::= & m \mapsto m \\
m & & \\
\mathfrak{F} & & \\
\mathfrak{M} & &
\end{array}
$$

variables, *labels*, *stack*, *stack frames*, *programs*, *directives*, *bindings*, *heap*, *values*, *expressions*, *parental map*, *memory locations*, *magic functions*, *magic methods*

Definition 1.1.

$$\textsc{Bind}(H, m_0, x, m) = H' \textit{ such that } B = H[m_0], B' = B[x \mapsto m], H' = H[m_0 \mapsto B']$$

Definition 1.2.

$$\textsc{Bind}(H, m_0, x_1, m_1, \ldots, x_n, m_n) = \textsc{Bind}(\ldots((\textsc{Bind}((\textsc{Bind}(H, m_0, x_1, m_1)), m_0, x_2, m_2), \ldots), m_0, x_n, m_n)$$

Definition 1.3.

$$\textsc{Lookup}(m_0, P, H, x_2) =$$

Definition 1.4.

$$\textsc{Catch}(\langle \ell, S\rangle) =$$

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

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

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

Figure 1: Operational Semantics

$$\dfrac{\begin{array}{c} S(\ell) = \ell : \ell' : x = (x_1, \dots, x_n) \qquad \forall i \in \{1, \dots, n\}, m_i = \textsc{Lookup}(m_0, P, H, x_i) \\ v = (m_1, \dots, m_n) \qquad B_{\texttt{obj}} = \{\star x_{\texttt{value}} \mapsto m\} \qquad H' = H[m \mapsto v, m' \mapsto B_{\texttt{obj}}] \\ m, m' \notin H \qquad \textsc{Bind}(H', m_0, x, m') = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{\langle \ell, S \rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S \rangle] \,||\, T, H'', P, m_0} \;\textsc{Tuple Assignment}$$

$$\dfrac{\begin{array}{c} S(\ell) = \ell : \ell' : x_1 = \texttt{def}\ (x_2, \dots, x_n) = \{S\} \qquad v = \langle m_0,\ \texttt{def}\ (x_2, \dots, x_n) \to S \rangle \\ B_{\texttt{obj}} = \{\star x_{\texttt{value}} \mapsto m\} \qquad H' = H[m \mapsto v, m' \mapsto B_{\texttt{obj}}] \\ m, m' \notin H \qquad \textsc{Bind}(H', m_0, x_1, m') = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}'' \end{array}}{[\langle \ell, S \rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}'', S \rangle] \,||\, T, H'', P, m_0} \;\textsc{Function Assignment}$$

$$\dfrac{\begin{array}{c} S(\ell) = \ell : \ell' : x_1 = x_2(x_3, \dots, x_n) \\ m = \textsc{Lookup}(m_0, P, H, x_2) \qquad H[m] = \langle m_0',\ \texttt{def}\ (x_3', \dots, x_n') \mapsto S' \rangle \\ m_0'' \notin H \qquad H' = H[m_0'' \mapsto \{\ \}] \qquad \forall i, 3 \le i \le n, m_i' = \textsc{Lookup}(m_0, P, H, x_i) \\ \textsc{Bind}(H', m_0'', x_1', m_1', \dots, x_n', m_n') = H'' \\ P' = P \cup \{m_0'' \mapsto m_0'\} \qquad S' = [\ell'' : \ell''' : d, \dots] \end{array}}{[\langle \ell, S \rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \ell'', S' \rangle, \langle l, S \rangle] \,||\, T, H'', P', m_0''} \;\textsc{Function Call Assignment}$$

$$\dfrac{\begin{array}{c} S(\ell) = \ell : \ell' : x_1 = x_2.x_3 \\ m = \textsc{Lookup}(m_0, P, H, x_2) \qquad H[m] = B \qquad B[x_3] = m' \qquad B_{\texttt{obj}} = \textsc{GetObj}(m') \\ m'' \notin H \qquad H' = H[m'' \mapsto B_{\texttt{obj}}] \qquad \textsc{Bind}(H', m_0, x_1, m'') = H'' \qquad \ell \overset{s}{\blacktriangleleft} \overset{*}{\ell}' \end{array}}{[\langle \ell, S \rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*}{\ell}', S \rangle] \,||\, T, H'', P, m_0} \;\textsc{Attribute Assignment}$$

Figure 1: Operational Semantics (cont.)

Raise Exception Caught

$$\frac{\begin{array}{c} S(\ell) = \ell : \ell' : \ \texttt{raise}\ x \qquad T = [\langle \ell_1, S_1\rangle, \ldots, \langle \ell_n, S_n\rangle] \\ \ell_1 = \ell' \qquad \forall i, 1 \leq i \leq k, k \leq n, \textsc{Catch}(\langle \ell_i, S_i\rangle) = \ \texttt{undefined} \\ \textsc{Catch}(\langle \ell_{k+1}, S_{k+1}\rangle) = \ell', x' \\ m = \textsc{Lookup}(m_0, P, H, x) \qquad \textsc{Bind}(H, m_0, x', m) = H' \qquad \ell_(k+1)' \overset{S_{k+1}}{\blacktriangleleft} \overset{*''}{\ell} \end{array}}{[\langle \ell, S\rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\langle \overset{*''}{\ell}, S_{k+1}\rangle] \,||\, T, H', P, m_0}$$

Raise Exception Escaped

$$\frac{\begin{array}{c} S(\ell) = \ell : \ell' : \ \texttt{raise}\ x \qquad T = [\langle \ell_1, S_1\rangle, \ldots, \langle \ell_n, S_n\rangle] \\ \ell_1 = \ell' \qquad \forall i, 1 \leq i \leq n, \textsc{Catch}(\langle \ell_i, S_i\rangle) = \ \texttt{undefined} \end{array}}{[\langle \ell, S\rangle] \,||\, T, H, P, m_0 \longrightarrow^1 [\ ], H, P, m_0}$$

Pass

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

Return

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

Return with Arguments

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

Goto

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

GotoIfNot

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

Name Statement

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

Figure 1: Operational Semantics (cont.)

End of Function

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

End of Program

$$\frac{T = [(\textsc{End}, t)]}{T, H, P, m_0 \longrightarrow^1 [\,], H, P, m_0}$$

Figure 1: Operational Semantics (cont.)

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

Figure 2: Normalized Python Language Grammar

$\hat{G}$	$::=$	$\{\hat{g}, \ldots\}$	*control flow graphs*
$\hat{g}$	$::=$	$\hat{o} \blacktriangleleft \hat{o} \mid \hat{o} \ll \hat{o}$	*control flow graph edge*
$\hat{o}$	$::=$	$\textsc{Start} \mid \textsc{End} \mid \hat{s}$	*control flow graph nodes*

Figure 3: Control Flow Graph Grammar

Lexical Start

$$\frac{\textsc{Start} \overset{?}{\blacktriangleleft} \hat{o}}{\hat{G} \longrightarrow^1 \hat{G} \cup \{\textsc{Start} \ll \hat{o}\}}$$

Literal Assignment

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

Variable Accessible

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

Figure 4: Control Flow Graph Closure

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

Figure 5: Value Lookup Grammar