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

Figure 1: 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 2: Control Flow Graph Grammar

1 CoPylot

1.1 Grammar

The grammar of the language to be analyzed appears in Figure 1.

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 2. *(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 $\hat{o} \overset{?}{\ll} \hat{o}'$ to denote $(\hat{o} \ll \hat{o}' \in \hat{G}$ when $\hat{G}$ is understood from

$$
\begin{array}{rcll}
\overset{*}{l} & ::= & l \mid \textsc{End} & \textit{labels} \\
T & ::= & [t, \ldots] & \textit{stack} \\
t & ::= & \overset{*}{l} \times S & \textit{stack frame} \\
S & ::= & [s, \ldots] & \textit{programs} \\
d & ::= & x = x \mid x = v \mid x = x(x) & \textit{directives} \\
B & ::= & x \mapsto m, \ldots & \textit{bindings} \\
H & ::= & m \mapsto v, \ldots & \textit{heap} \\
v & ::= & \texttt{int}^{+} \mid \texttt{int}^{-} \mid \texttt{int}^{0} \mid \langle \eta, fun(x) \mapsto S \rangle & \textit{values} \\
\eta & & & \textit{pointers to scopes} \\
E & ::= & \eta \mapsto B & \textit{environments} \\
P & ::= & \eta \mapsto \eta & \textit{parental maps}
\end{array}
$$

Figure 3: Operational Semantics Grammar

$$\textsc{Literal Assignment}\quad \frac{\begin{array}{c} S(l) = l : l' : x = v \qquad H' = H[m \mapsto v] \\ m \notin H \qquad B = E(\eta) \qquad B' = B[x \mapsto m] \qquad E' = E[\eta \mapsto B'] \qquad ll'' \end{array}}{[\langle l, S\rangle] \,||\, T, H, E, P, \eta \longrightarrow^1 [\langle l'', S\rangle] \,||\, T, H', E', P, \eta}$$

$$\textsc{Variable Assignment}\quad \frac{\begin{array}{c} S(l) = l : l' : x_1 = v_2 \qquad m = \textsc{Lookup}(\eta, P, E, x_1) \\ B = E(\eta) \qquad B' = B[x_1 \mapsto m] \qquad E' = E[\eta \mapsto B'] \qquad ll'' \end{array}}{[\langle l, S\rangle] \,||\, T, H, E, P, \eta \longrightarrow^1 [\langle l'', S\rangle] \,||\, T, H, E', P, \eta}$$

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

Figure 4: Operation Semantics

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

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

Figure 5: Control Flow Graph Closure

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 $\overset{?}{\ll}$ 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***

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

Figure 6: Value Lookup Grammar

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

2. ***Variable Lookup***

(a) $\boxed{\textsc{Value Discovery}}$
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) $\boxed{\textsc{Value Skip}}$
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) $\boxed{\textsc{Value Aliasing}}$
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* $\hat{G}(\hat{o}_1, [\hat{x}'] \,||\, \hat{K}) \subseteq \hat{V}$*.*