1 Formale Definition einer Value-Analysis CPA in Kombination mit ConstraintsCPA

Grundsätzliche Annahme: Zu analysierende Programme verwenden nur Integer-Werte.

1.1 Value-Analysis CPA

Bei der implementierten Value-Analysis CPA handelt es sich um eine Erweiterung der in [Beyer2014] beschriebenen Constant Propagation CPA.

Die Value-Analysis CPA

$$\mathbb{S} = (D_{\mathbb{S}},\ \rightsquigarrow_{\mathbb{S}},\ merge_{\mathbb{S}},\ stop_{\mathbb{S}})$$

mit

1. $D_{\mathbb{S}} = (C,\ \mathcal{E},\ [\![\cdot]\!])$, wobei:
   - C die Menge der konkreten Programmzustände,
   - $\mathcal{E} = (X \to (\mathcal{Z}_{\mathbb{CO}} \cup \mathcal{Z}_{\mathbb{S}}),\ \sqsubseteq,\ \sqcup,\ v_\top)$ mit
     - X Menge von Programmvariablen,
     - $\mathcal{Z}_{\mathbb{CO}} = \mathbb{Z} \cup \top_{\mathcal{Z}}$
     - $\mathcal{Z}_{\mathbb{S}} = S_I \cup S_E$.
       S_I ist dabei die Menge aller symbolischen Identifier und S_E die Menge aller symbolischen Ausdrücke, die induktiv wie folgt definiert ist:
       * Für $a, b \in (S_I \cup \mathbb{Z})$ gilt: $(a \circ b) \in S_E$
       * Für $c, d \in S_E$ gilt: $(c \circ d) \in S_E$

       mit $\circ \in \{+,\ -,\ *,\ /,\ \%,\ \ll,\ \gg, \&,\ |,\ \oplus\}$
     - $\sqsubseteq$ ist definiert mit
       $$v \sqsubseteq v' \text{ falls } \forall x \in X : v(x) = v'(x) \vee v'(x) = \top_{\mathcal{Z}}$$
     - $\sqcup$ hält die kleinste obere Schranke mit
       $$(v \sqcup v')(x) = \begin{cases} v(x) & \text{falls } v(x) = v'(x) \\ \top_{\mathcal{Z}} & \text{sonst} \end{cases}$$
     - $v_\top(x) = \top_{\mathcal{Z}}$ für alle $x \in X$.
   - Ein konkreter Zustand $c \in C$ ist zu einer abstrakten Variablenzuweisung v kompatibel, falls
     1. $c(x) = v(x)$
     2. $v(x) = \top_{\mathcal{Z}}$ oder

3. $v(x) \in \mathcal{Z}_{\mathbb{S}}$ und eine Zuweisung $v' : S_I \to \mathbb{Z}$ existiert, so dass $v''(x) = c(x)$, wobei $v''(x)$ durch Ersetzen aller $a \in S_I$ in $v(x)$ durch $v'(a)$ entsteht.

Die Konkretisierungsfunktion $[\![\cdot]\!]$ weist einem abstrakten Zustand v alle konkreten Zustände zu, die zu v kompatibel sind.

2. Die Übergangsfunktion $\rightsquigarrow_{\mathbb{S}}$ besitzt den Übergang $v \overset{g}{\rightsquigarrow} v'$ falls:

   1. $g = (l, assume(p), l')$, $\phi(p, v)$ erfüllbar und für alle $x \in X$ gilt:

   $$v'(x) = \begin{cases} c & \text{falls } c \text{ die einzige erfüllende Belegung für } \phi(p,v) \\ y & \text{falls } v(x) = \top_{\mathcal{Z}} \text{ und } x \text{ in } p \text{ vorkommt. Dabei} \\ & y \in S_I \text{ und y ein neuer Wert, der bisher in} \\ & \text{keinem Zustand verwendet wurde.} \\ v(x) & \text{sonst} \end{cases}$$

   mit

   $$\phi(p, v) := p \wedge (\bigwedge_{\substack{x \in X, \\ v(x) \in \mathbb{Z}}} x = v(x))$$

   ϕ führt eine Überapproximierung durch, da nur Werte $x \in X$ betrachtet werden, die einen expliziten Wert besitzen.

   2. $g = (l, w := e, l')$ und für alle $x \in X$ gilt:

   $$v'(x) = \begin{cases} eval(e, v) & \text{falls } x = w \\ v(x) & \text{sonst} \end{cases}$$

   Wobei für einen Ausdruck e über die Variablen in X gilt:

   $$eval(e, v) := \begin{cases} y & \text{falls } v(x) = \top_{\mathcal{Z}} \text{ für ein in } e \text{ enthaltenes } x \in X, \\ & \text{mit } y \in S_I \text{ und } v(a) \neq y \text{ für alle } a \in X \\ z & \text{sonst, wobei } e \text{ partiell zu } z \text{ ausgewertet wird,} \\ & \text{indem jede Variable } x \text{ in } e \text{ durch } v(x) \text{ ersetzt} \\ & \text{wird. (Evtl. gilt } z \in S_E) \end{cases}$$

   3. $v' = v_\top$.

3. $merge_{\mathbb{S}} = merge^{sep}$, d.h. Zustände werden nach Branches nicht gemerged.

4. $stop_{\mathbb{S}} = stop^{sep}$, d.h. jeder Zustand wird einzeln betrachtet.

1.2 ConstraintsCPA

Die ConstraintsCPA ist eine CPA

$$\mathbb{C} = (D_{\mathbb{C}},\ \rightsquigarrow_{\mathbb{C}},\ merge_{\mathbb{C}},\ stop_{\mathbb{C}})$$

welche Constraints (d.h. boolesche Formeln) verwendet, um die Erfüllbarkeit von assumes zu bestimmen.

1. $D_{\mathbb{C}} = (C,\ \mathcal{C},\ [\![\cdot]\!])$ mit:
   - C Menge der konkreten Programmzustände,
   - $\mathcal{C} = (2^{\gamma}, \sqsubseteq, \sqcup, \top)$. Dabei ist:
     - γ die Menge aller möglichen Constraints, die wie folgt definiert sind: Für $a, b \in \mathcal{Z}_{\mathbb{CO}} \cup \mathcal{Z}_{\mathbb{S}}$ gilt:
       * $(a \circ b) \in \gamma$ mit $\circ \in \{==,\ <,\ \leq,\ >,\ \geq\}$.
       * $!a \in \gamma$
     - $\sqsubseteq$ definiert als $a \sqsubseteq a'$, falls $a' \subseteq a$
     - $\sqcup$ definiert als $a \sqcup a' = a \cap a'$
   - $[\![\cdot]\!]$ ist definiert als

     $$[\![a]\!] = \{c \in C |\ c \vDash \bigwedge_{\varphi \in a} \varphi\}$$

     Bemerkung: Es gelte $\bigwedge_{\varphi \in \{\}} \varphi = true$

2. Die Übergangsfunktion $\rightsquigarrow_{\mathbb{C}}$ besitzt den Übergang $a \overset{g}{\rightsquigarrow}_{\mathbb{C}} a$ für alle $a \in \mathbb{C}$. Neue Zustände werden nur im Strengthening mit anderen CPAs (hier konkret der Value-Analysis) erzeugt.

3. $merge_{\mathbb{C}} = merge^{sep}$

4. $stop_{\mathbb{C}} = stop^{sep}$

1.3 Composition von LocationCPA, Value-Analysis und ConstraintsCPA

$\mathcal{C}_{\mathbb{LSC}} = (\mathbb{L},\ \mathbb{S},\ \mathbb{C},\ \rightsquigarrow_x,\ merge_x,\ stop_x)$ ist die CompositeCPA der Komposition von LocationCPA, Value-Analysis CPA und ConstraintsCPA.

1. $\rightsquigarrow_x$: $D_{\mathbb{S}} \times D_{\mathbb{C}} \rightarrow D_{\mathbb{S}} \times D_{\mathbb{C}}$ besitzt den Übergang $(l, v, a) \overset{g}{\rightsquigarrow}_x (l', v', a')$, falls:
   - $l \overset{g}{\rightsquigarrow}_{\mathbb{L}} l'$,
   - $v \overset{g}{\rightsquigarrow}_{\mathbb{S}} v'$,

- $a \overset{g}{\rightsquigarrow}_{\mathbb{C}} a$,
- $\downarrow_{\mathbb{C},\mathbb{S}} (a, v')$ definiert ist und
- $\downarrow_{\mathbb{C},\mathbb{S}} (a, v') = a'$.

2. $merge_x = merge^{sep}$
3. $stop_x = stop^{sep}$

1.3.1 Strengthening

$\downarrow_{\mathbb{C},\mathbb{S}}$: $\mathbb{C} \times \mathbb{S} \to \mathbb{C}$ benutzt konkrete Werte $v(x)$ aus dem zweiten gegebenen Zustand, um die Variablen, die in Constraints des ersten Zustands vorkommen, durch konkrete Werte zu ersetzen. $\downarrow_{\mathbb{C},\mathbb{S}} (a, v) = a'$ ist definiert, falls $\bigwedge_{\varphi \in a'} \varphi$ erfüllbar, wobei gilt, dass a' aus a durch Ersetzen aller $x \in X$ durch $v(x)$ entsteht.