CPCC

Methodology

对于某一应用程序涉及到的任意一条上下文而言，该上下文中的元素可以根据受到调度过程生成的某一上下文变化序列的影响划分为若干集合，从而为后续的修改 CCT，真值计算，以及约束检测提供便利。

AddS, DelS, and UpdS. 设 Ctx_0 表示在应用某一上下文变化序列前某一应用程序的上下文池中的某一条上下文。Ctx 表示在应用该上下文变化序列后该应用程序上下文池中的这一条上下文。用 seq 表示该上下文变化序列投影到这一条上下文后的结果，投影就是保留该上下文变化序列中和这条上下文相关的那些上下文变化，且这些上下文变化的相对顺序不变。我们为上下文池中的每一条上下文都维护三个集合，*Add* 集合 (用”$AddS$” 表示)，*Del* 集合 (用”$DelS$” 表示)，*Upd* 集合 (用”$UpdS$” 表示)。

Add 集合包含了应用该上下文变化序列后，这一条上下文中实际需要添加的元素，即有 $AddS = Ctx - Ctx_0$。

Del 集合则包含了应用该上下文变化序列后，这一条上下文中实际需要删除的元素，即有 $DelS = Ctx_0 - Ctx$。

Upd 集合则包含了满足如下条件的元素 e：(1) Ctx_0 包含该元素 e。(2) seq 中有若干上下文变化二元组，二元组中第一个上下文变化为 $< -, Ctx_0, e >$，第二个上下文变化为 $< +, Ctx_0, e >$，这两个上下文变化不一定相邻。

因为 CPCC 技术基于增量，所以和 PCC 一样，需要一个函数来指明该应用程序涉及到的所有一致性约束公式中哪些公式受到了上下文变化序列的影响。

affected. $\texttt{affected}(f)$ 表示给定的约束公式 f 是否受到上下文变化序列中任意上下文变化的影响从而是否需要重新评估。$\top$ 表示需要，$\bot$ 表示不需要。`affected` 函数的值是通过在静态语法树上自底向上的方式决定的，函数规则如下：

给定公式 $\forall\gamma \in Ctx\ (f)$。如果 Ctx 对应的 $AddS$，$DelS$，$UpdS$ 三个集合中至少有一个不为空或者 $\texttt{affected}(f) = \top$，则 $\texttt{affected}(\forall\gamma \in Ctx\ (f)) = \top$. 反之则为 $\bot$。该规则同样适用于 $\exists\gamma \in Ctx\ (f)$

给定公式 (f_1) **and** (f_2)。如果 $\texttt{affected}(f_1) = \top$ 或者 $\texttt{affected}(f_2) = \top$，则 $\texttt{affected}((f_1)\ \textbf{and}\ (f_2)) = \top$。反之则为 $\bot$。该规则同样适用于 (f_1) **or** (f_2) 和 (f_1) **implies** (f_2)

给定公式 **not** (f)。如果 $\texttt{affected}(f) = \top$，则 $\texttt{affected}(\textbf{not}\ (f)) = \top$。反之则为 $\bot$。

给定公式 $bfunc(\gamma_1, \cdots, \gamma_n)$，因为该公式既不和上下文变化相关又不含有子公式，所以 $\texttt{affected}(bfunc(\gamma_1, \cdots, \gamma_n))$ 的缺省值设为 $\bot$。

和 ConC 类似，CPCC 把 $\forall$ 和 $\exists$ 公式对应运行时结点视作可能发生并发的结点。但并非所有 $\forall$ 和 $\exists$ 公式对应的运行时结点都适合并发，我们称对应运行时结点适合并发的 $\forall$ 和 $\exists$ 公式为关键公式 *(critical formula)*，使用算法 1 来寻找关键公式。算法 1 按照自顶向下的方式遍

历该应用程序的某一一致性约束对应的整个静态语法树 (第 2 行)。对所有该一致性约束中可能并发的 ∀ 和 ∃ 公式 (第 3 行)，算法都会检查该公式是否满足开启并发的条件 (第 6 行)，满足开启并发条件的公式即为关键公式。

Algorithm 1 (critical-formula-finding) Finding critical formulas.

Input: consistency constraint c.

Output: a set of static nodes representing critical formulas S_{cf}.

1: $S_{cf} \leftarrow \emptyset$
2: **for** $sNode$ from the syntax tree of c in a top-down-way **do**
3:     **if** $sNode$ represents ∀ formula or ∃ formula **then**
4:         $Ctx \leftarrow$ the context related with $sNode$.
5:         $sf \leftarrow$ the formula represented by sub-node of $sNode$.
6:         **if** ($Ctx.AddS \neq \emptyset$ **or** $Ctx.UpdS \neq \emptyset$ **or** $\texttt{affected}(sf) = \top$)
            and (none of ancestors of $sNode$ is in S_{cf}) **then**
7:             $S_{cf} \leftarrow \texttt{add}(S_{cf}, sNode)$.
8:         **end if**
9:     **end if**
10: **end for**
11: **return** S_{cf}

关于开启并发的条件，首先需要满足”$Ctx.AddS \neq \emptyset$ **or** $Ctx.UpdS \neq \emptyset$ **or** $\texttt{affected}(sf) = \top$”，其中 $Ctx.AddS \neq \emptyset$ 是因为对于 Add 集合中的元素，在进行约束检测时需要先在相应的运行时结点上添加这些元素对应的分支并进行真值计算和链接生成，这部分的操作时间开销较大，适合并发执行。$Ctx.UpdS \neq \emptyset$ 类似，Upd 集合中的元素由于受到上下文变化序列的影响，元素部分域中的值可能被更新，虽然不需要重新构建这些元素对应分支，但仍然需要重新计算真值和生成链接，因此这部分操作同样值得并发处理。对于第 3 部分 $\texttt{affected}(sf) = \top$，当子公式受到上下文变化序列的影响时，当前公式对应的运行时结点的那些原有的分支都需要以增量的方式进行修改分支，计算真值和生成链接，虽然增量的机制带来了一部分性能的提升，但链接生成阶段仍然包含了大量的集合操作，性能瓶颈依旧存在，也能使用并发进行加速。最后，对于”none of ancestors of $sNode$ is in S_{cf}”，我们希望开启并发的运行时结点尽可能高，从而使得每个线程的工作量尽可能多。

$\tau_{Partial}[\forall\gamma \in Ctx\ (f)]_\alpha =$

(1) $\tau_0[\forall\gamma \in Ctx\ (f)]_\alpha$

$if\ AddS = \emptyset\ and\ DelS = \emptyset\ and\ UpdS = \emptyset\ and\ \texttt{affected}(f) = \perp$

(2) $\{(t_1, \cdots, t_a) = TruthEvaluation_{Entire}(\tau[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS);$

$return\ (\tau_0[\forall\gamma \in Ctx\ (f)]_\alpha \wedge t_1 \wedge \cdots \wedge t_a);\}$

$if\ AddS \neq \emptyset\ and\ UpdS = \emptyset\ and\ DelS = \emptyset\ and\ \texttt{affected}(f) = \perp$

(3) $\{(t_1, \cdots, t_{a+u}) = TruthEvaluation_{Entire}(\tau[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS \cup UpdS);$

$return\ (\top \wedge \tau_0[f]_{bind((\gamma,x_1),\alpha)} \wedge \cdots \wedge \tau_0[f]_{bind((\gamma,x_{n-a-u}),\alpha)} \wedge t_1 \wedge \cdots \wedge t_{a+u} \mid x_i \in Ctx - (AddS \cup UpdS))\}$

$if\ DelS \neq \emptyset\ and\ \texttt{affected}(f) = \perp$

(4) $\{(t_1, \cdots, t_n) = TruthEvaluation_{Partial}(\tau[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in Ctx);$

$return\ (\top \wedge t_1 \wedge \cdots \wedge t_n);\}$

$if\ AddS = \emptyset\ and\ DelS = \emptyset\ and\ UpdS = \emptyset\ and\ \texttt{affected}(f) = \top$

(5) $\{(t_1, \cdots, t_{a+u}) = TruthEvaluation_{Entire}(\tau[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS \cup UpdS);$

$(t_{a+u+1}, \cdots, t_n) = TruthEvaluation_{Partial}(\tau[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in Ctx - (AddS \cup UpdS));$

$return\ (\top \wedge t_1 \wedge \cdots \wedge t_n);\}$

$if\ (AddS \neq \emptyset\ or\ UpdS \neq \emptyset\ or\ DelS \neq \emptyset)\ and\ \texttt{affected}(f) = \top$

$\tau_{Entire}[\forall\gamma \in Ctx\ (f)]_\alpha = \top \wedge \tau_{Entire}[f]_{bind((\gamma,x_1),\alpha)} \wedge \cdots \wedge \tau_{Entire}[f]_{bind((\gamma,x_n),\alpha)} \mid x_i \in Ctx.$

$TruthEvaluation_{Entire}(\tau[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in S) =$

(i) $\tau_{Entire}[f]_{bind((\gamma,x_1),\alpha)}\ ;\ \cdots\ ;\ \tau_{Entire}[f]_{bind((\gamma,x_s),\alpha)}$

$if\ \forall\gamma \in Ctx(f)\ is\ not\ a\ critical\ formula$

(ii) $\tau_{Entire}[f]_{bind((\gamma,x_1),\alpha)} \parallel \cdots \parallel \tau_{Entire}[f]_{bind((\gamma,x_s),\alpha)}$

$if\ \forall\gamma \in Ctx(f)\ is\ a\ critical\ formula$

$TruthEvaluation_{Partial}(\tau[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in S) =$

(i) $\tau_{Partial}[f]_{bind((\gamma,x_1),\alpha)}\ ;\ \cdots\ ;\ \tau_{Partial}[f]_{bind((\gamma,x_s),\alpha)}$

$if\ \forall\gamma \in Ctx(f)\ is\ not\ a\ critical\ formula$

(ii) $\tau_{Partial}[f]_{bind((\gamma,x_1),\alpha)} \parallel \cdots \parallel \tau_{Partial}[f]_{bind((\gamma,x_s),\alpha)}$

$if\ \forall\gamma \in Ctx(f)\ is\ a\ critical\ formula$

上面所给的是 $\forall\gamma \in Ctx(f)$ 公式的真值计算语义，共分为 3 个部分，第 1 部分为 $\tau_{Parital}[\forall\gamma \in Ctx(f)]_\alpha$，该部分描述的是以增量的方式对 $\forall$ 公式进行真值计算时的语义动作。根据 3 个集合以及 `affected` 函数一共分为 5 种情况。

1 上下文变化序列没有影响 f 且 Ctx 对应的 $AddS$，$DelS$ 以及 $UpdS$ 都为空集，则可以复用之前的真值 τ_0。

2 上下文变化序列没有影响 f 且 Ctx 对应的 $DelS$ 和 $UpdS$ 为空集，但 Ctx 对应的 $AddS$ 不为空集，则仍然可以复用之前的真值 τ_0，但需要加上 $AddS$ 中元素的影响，我们使用函数 $TruthEvaluation_{Entire}$ 来计算 $AddS$ 中元素对应的子分支的真值，因为这些元素对应的子分支的真值需要完全计算，所以我们使用下标 $Entire$ 来表示这一点。

3 上下文变化序列没有影响 f 但 Ctx 对应的 $DelS$ 不为空集，此处省略了对 $AddS$ 和 $UpdS$ 的描述，因为此时不管这两个集合是什么样的状态，都符合语义。在这种情况下，那些未被删除的旧子分支的真值可以复用，$Ctx - AddS - UpdS$ 就是那些还未被删除的子分支对应的元素，除此之外还需考虑 $AddS$ 和 $UpdS$ 可能不为空集，所以还需要加上这两个集合中元素的影响，我们同样使用 $TruthEvaluation_{Entire}$ 函数来计算。

4 上下文变化序列影响了 f 但 Ctx 对应的 $AddS$，$DelS$ 以及 $UpdS$ 都为空集，则之前的真值已经不可复用，因为所有的子分支都需要重新进行真值计算。而此时并不需要完全重新计算真值，可以以增量的方式进行计算，我们使用函数 $TruthEvaluation_{Partial}$ 来计算这些子分支的真值，因为通过增量的方式进行计算，所以我们使用下标 $Partial$ 来表示这一点。

5 上下文变化序列影响了 f 且 Ctx 对应的 $AddS$，$DelS$ 以及 $UpdS$ 至少有一个不为空集，之前的真值已经不可复用。在这种情况下，Ctx 中的元素可以划分为两类，一类由 $AddS$ 和 $UpdS$ 中元素构成，这一类的元素需要完全重新计算它们所对应的子分支的真值，因此使用函数 $TruthEvaluation_{Entire}$。剩下的元素构成另一类，可以通过增量的方式对相应的子分支计算真值，因此使用函数 $TruthEvaluation_{Partial}$。

第 2 部分为 $\tau_{Entire}[\forall\gamma \in Ctx(f)]_\alpha$，该部分描述的是完全计算 $\forall$ 公式的真值时的语义动作，该部分语义和 ECC. 语义一致。

第 3 部分是对 $TruthEvaluation_{Entire}$ 和 $TruthEvaluation_{Partial}$ 两个函数的具体描述，这两个函数根据 f 的父亲公式 (此时为 $\forall\gamma \in Ctx(f)$) 是否是关键公式来决定以串行还是并行的机制去对集合 S 中元素对应的分支进行真值计算。我们采用进程代数中的定义，用”;” 表示串行，用”||” 表示并行。

$\exists$ 公式的语义和 $\forall$ 公式的语义类似，接下来考虑 **and** 公式，其真值计算语义如下

$$
\begin{aligned}
&\tau_{Partial}[(f_1)\ \textbf{and}\ (f_2)]_\alpha = \\
&\qquad (1)\ \ \tau_0[(f_1)\ \textbf{and}\ (f_2)]_\alpha,\ \ if\ \texttt{affected}(f_1) = \texttt{affected}(f_2) = \bot \\
&\qquad (2)\ \ \tau_0[f_1]_\alpha \wedge \tau_{Partial}[f_2]_\alpha,\ \ if\ \texttt{affected}(f_1) = \bot, \texttt{affected}(f_2) = \top \\
&\qquad (3)\ \ \tau_{Partial}[f_1]_\alpha \wedge \tau_0[f_2]_\alpha,\ \ if\ \texttt{affected}(f_1) = \top, \texttt{affected}(f_2) = \bot \\
&\qquad (4)\ \ \tau_{Partial}[f_1]_\alpha \wedge \tau_{Partial}[f_2]_\alpha,\ \ if\ \texttt{affected}(f_1) = \texttt{affected}(f_2) = \top
\end{aligned}
$$

$$
\tau_{Entire}[(f_1)\ \textbf{and}\ (f_2)]_\alpha = tau_{Entire}[f_1]_\alpha \wedge \tau_{Entire}[f_2]_\alpha
$$

上述语义共分为 2 个部分，第 1 部分描述的是以增量的方式对 **and** 公式进行真值计算时的语义动作，该部分语义前 3 点和 PCC 的语义一样，而考虑到上下文变化序列中也可能是多个上下文变化，则可能对 f_1, f_2 均有影响，第 4 点则是这种情况。第 2 部分则为对 **and** 公式完全真值计算的语义动作，该部分语义动作和 ECC 语义一致。or 公式和 $implies$ 公式语义和上述类似。接下来考虑 **not** 公式，其真值计算语义如下

$$
\begin{aligned}
&\tau_{Partial}[\textbf{not}\ (f)]_\alpha = \\
&\qquad (1)\ \ \tau_0[\textbf{not}\ (f)]_\alpha,\ \ if\ \texttt{affected}(f) = \bot \\
&\qquad (2)\ \ \tau_{Partial}[\textbf{not}\ (f)]_\alpha,\ \ if\ \texttt{affected}(f) = \top
\end{aligned}
$$

$$
\tau_{Entire}[\textbf{not}\ (f)]_\alpha = \tau_{Entire}[\textbf{not}\ (f)]_\alpha
$$

上述语义同样分为 2 个部分，第 1 部分描述的是以增量的方式对 **not** 公式进行真值计算时的语义动作，与 PCC 语义一致，第 2 部分则为对 **not** 公式完全真值计算的语义动作，和 ECC 语义一致。最后考虑 $bfunc$ 公式，其真值计算语义如下

$$
\tau_{Partial}[bfunc(\gamma_1, \cdots, \gamma_n)]_\alpha = \tau_0[bfunc(\gamma_1, \cdots, \gamma_n)]_\alpha
$$

$$
\tau_{Entire}[bfunc(\gamma_1, \cdots, \gamma_n)]_\alpha = bfunc(\gamma_1, \cdots, \gamma_n)
$$

上述语义同样分为 2 个部分，第 1 部分描述的是以增量的方式对 $bfunc$ 公式进行真值计算时的语义动作，与 PCC 语义一致，第 2 部分则为对 $bfunc$ 公式完全真值计算的语义动作，和 ECC 语义一致。

Links generation. 对链接生成语义的分析和真值计算类似，故省略。下面给出 $\forall\gamma \in Ctx(f)$ 公式的链接生成语义。

$$\mathcal{L}_{Partial}[\forall\gamma \in Ctx(f)]_\alpha =$$

$$(1)\mathcal{L}_0[\forall\gamma \in Ctx(f)]_\alpha$$

$$if\ AddS = \emptyset\ and\ DelS = \emptyset\ and\ UpdS = \emptyset\ and\ \texttt{affected}(f) = \bot$$

$$(2)\{(l_1, \cdots, l_a) = LinksGeneration_{Entire}(\mathcal{L}[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS \wedge \tau[f]_{bind((\gamma,y_j),\alpha)} = \bot);$$

$$return\ (\mathcal{L}_0[\forall\gamma \in Ctx(f)]_\alpha \cup (\{(violated, \{\gamma, y_1\})\} \otimes l_1) \cup \cdots \cup (\{(violated, \{\gamma, y_a\})\} \otimes l_a)); \}$$

$$if\ AddS \neq \emptyset\ and\ UpdS = \emptyset\ and\ DelS = \emptyset\ and\ \texttt{affected}(f) = \bot$$

$$(3)\{(l_1, \cdots, l_{a+u}) = LinksGeneration_{Entire}(\mathcal{L}[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS \cup UpdS \wedge \tau[f]_{bind((\gamma,y_j),\alpha)} = \bot);$$

$$return\ ((\{(violated, \{\gamma, y_1\})\} \otimes l_1) \cup \cdots \cup (\{(violated, \{\gamma, y_{a+u}\})\} \otimes l_{a+u}) \cup$$

$$\{l \mid l \in \{(\textbf{violated}, \{(\gamma, x_i)\})\} \otimes \mathcal{L}_0[f]_{bind((\gamma,x_i),\alpha)}\} \mid x_i \in Ctx - (AddS \cup UpdS) \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot); \}$$

$$if\ DelS \neq \emptyset\ and\ \texttt{affected}(f) = \bot$$

$$(4)\{(l_1, \cdots, l_n) = LinksGeneration_{Partial}(\mathcal{L}[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in Ctx \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot);$$

$$return\ (\emptyset \cup (\{(vioalted, \{\gamma, x_1\})\} \otimes l_1) \cup \cdots \cup (\{(vioalted, \{\gamma, x_n\})\} \otimes l_n)\ ); \}$$

$$if\ AddS = \emptyset\ and\ DelS = \emptyset\ and\ UpdS = \emptyset\ and\ \texttt{affected}(f) = \top$$

$$(5)\{(l_1, \cdots, l_{a+u}) = LinksGeneration_{Entire}(\mathcal{L}[f]_{bind((\gamma,y_j),\alpha)} \mid y_j \in AddS \cup UpdS \wedge \tau[f]_{bind((\gamma,y_j),\alpha)} = \bot);$$

$$(l_{a+u+1}, \cdots l_n) = LinksGeneration_{Partial}(\mathcal{L}[f]_{bind((\gamma,x_i),\alpha)}$$

$$\mid x_i \in Ctx - (AddS \cup UpdS) \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot);$$

$$return\ (\emptyset \cup (\{(violated, \{\gamma, y_1\})\} \otimes l_1) \cup \cdots \cup (\{(violated, \{\gamma, y_n\})\} \otimes l_n)); \}$$

$$if\ (AddS \neq \emptyset\ or\ UpdS \neq \emptyset\ or\ DelS \neq \emptyset)\ and\ \texttt{affected}(f) = \top$$

$$\mathcal{L}_{Entire}[\forall\gamma \in Ctx(f)]_\alpha =$$

$$\{l \mid l \in \{(\textbf{violated}, \{(\gamma, x_i)\})\} \otimes \mathcal{L}_{Entire}[f]_{bind((\gamma,x_i),\alpha)}\} \mid x_i \in Ctx \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot)$$

$$LinksGeneration_{Entire}(\mathcal{L}[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in S \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot)$$

$$(i)\ \ \mathcal{L}_{Entire}[f]_{bind((\gamma,x_1),\alpha)}\ ;\ \cdots\ ;\ \mathcal{L}_{Entire}[f]_{bind((\gamma,x_s),\alpha)}$$

$$if\ \forall\gamma \in Ctx(f)\ is\ not\ a\ critical\ formula$$

$$(ii)\ \ \mathcal{L}_{Entire}[f]_{bind((\gamma,x_1),\alpha)} \parallel \cdots \parallel \mathcal{L}_{Entire}[f]_{bind((\gamma,x_s),\alpha)}$$

$$if\ \forall\gamma \in Ctx(f)\ is\ a\ critical\ formula$$

$$LinksGeneration_{Partial}(\mathcal{L}[f]_{bind((\gamma,x_i),\alpha)} \mid x_i \in S \wedge \tau[f]_{bind((\gamma,x_i),\alpha)} = \bot)$$

$$(i)\ \ \mathcal{L}_{Partial}[f]_{bind((\gamma,x_1),\alpha)}\ ;\ \cdots\ ;\ \mathcal{L}_{Parital}[f]_{bind((\gamma,x_s),\alpha)}$$

$$if\ \forall\gamma \in Ctx(f)\ is\ not\ a\ critical\ formula$$

$$(ii)\ \ \mathcal{L}_{Partial}[f]_{bind((\gamma,x_1),\alpha)} \parallel \cdots \parallel \mathcal{L}_{Partial}[f]_{bind((\gamma,x_s),\alpha)}$$

$$if\ \forall\gamma \in Ctx(f)\ is\ a\ critical\ formula$$

$\exists$ 公式链接生成语义与上类似，接下来给出 **and** 公式的链接生成语义。

$$
\begin{aligned}
&\mathcal{L}_{Partial}[(f_1)\ \textbf{and}\ (f_2)]_\alpha = \\
&\quad (1)\ \ \mathcal{L}_0[(f_1)\ \textbf{and}\ (f_2)]_\alpha,\ \ if\ \texttt{affected}(f_1) = \texttt{affected}(f_2) = \bot \\
&\quad (2)\ \ a.\ \mathcal{L}_{Partial}[f_1]_\alpha \otimes \mathcal{L}_0[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \top \\
&\qquad\ \ b.\ \mathcal{L}_{Partial}[f_1]_\alpha \cup \mathcal{L}_0[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \bot \\
&\qquad\ \ c.\ \mathcal{L}_0[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \top\ and\ \tau[f_2]_\alpha = \bot \\
&\qquad\ \ d.\ \mathcal{L}_{Partial}[f_1]_\alpha,\ \ if\ \tau[f_1]_\alpha = \bot\ and\ \tau[f_2]_\alpha = \top \\
&\qquad\ \ if\ \texttt{affected}(f_1) = \top, \texttt{affected}(f_2) = \bot \\
&\quad (3)\ \ a.\ \mathcal{L}_0[f_1]_\alpha \otimes \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \top \\
&\qquad\ \ b.\ \mathcal{L}_0[f_1]_\alpha \cup \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \bot \\
&\qquad\ \ c.\ \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \top\ and\ \tau[f_2]_\alpha = \bot \\
&\qquad\ \ d.\ \mathcal{L}_0[f_1]_\alpha,\ \ if\ \tau[f_1]_\alpha = \bot\ and\ \tau[f_2]_\alpha = \top \\
&\qquad\ \ if\ \texttt{affected}(f_1) = \bot, \texttt{affected}(f_2) = \top \\
&\quad (4)\ \ a.\ \mathcal{L}_{Partial}[f_1]_\alpha \otimes \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \top \\
&\qquad\ \ b.\ \mathcal{L}_{Partial}[f_1]_\alpha \cup \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \bot \\
&\qquad\ \ c.\ \mathcal{L}_{Partial}[f_2]_\alpha,\ \ if\ \tau[f_1]_\alpha = \top\ and\ \tau[f_2]_\alpha = \bot \\
&\qquad\ \ d.\ \mathcal{L}_{Partial}[f_1]_\alpha,\ \ if\ \tau[f_1]_\alpha = \bot\ and\ \tau[f_2]_\alpha = \top \\
&\qquad\ \ if\ \texttt{affected}(f_1) = \texttt{affected}(f_2) = \bot
\end{aligned}
$$

$$
\begin{aligned}
&\mathcal{L}_{Entire}[(f_1)\ \textbf{and}\ (f_2)]_\alpha = \\
&\quad (1)\ \ \mathcal{L}_{Entire}[f_1]_\alpha \otimes \mathcal{L}_{Entire}[f_2]_\alpha,\quad if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \top \\
&\quad (2)\ \ \mathcal{L}_{Entire}[f_1]_\alpha \cup \mathcal{L}_{Entire}[f_2]_\alpha,\quad if\ \tau[f_1]_\alpha = \tau[f_2]_\alpha = \bot \\
&\quad (3)\ \ \mathcal{L}_{Entire}[f_2]_\alpha,\quad if\ \tau[f_1]_\alpha = \top\ and\ \tau[f_2]_\alpha = \bot \\
&\quad (4)\ \ \mathcal{L}_{Entire}[f_1]_\alpha,\quad if\ \tau[f_1]_\alpha = \bot\ and\ \tau[f_2]_\alpha = \top
\end{aligned}
$$

or 公式和 $implies$ 公式语义和上述类似。接下来考虑 **not** 公式，其链接生成语义如下

$$\begin{aligned}
&\mathcal{L}_{Partial}[\textbf{not}\ (f)]_\alpha = \\
&\quad (1)\mathcal{L}_0[\textbf{not}\ (f)]_\alpha, \quad if\ \texttt{affected}(f) = \bot \\
&\quad (2)\texttt{FlipSet}(\mathcal{L}_{Partial}[\textbf{not}\ (f)]_\alpha), \quad if\ \texttt{affected}(f) = \top
\end{aligned}$$

$$\mathcal{L}_{Entire}[\textbf{not}\ (f)]_\alpha = \texttt{FlipSet}(\mathcal{L}_{Entire}[\textbf{not}\ (f)]_\alpha)$$

最后给出 $bfunc$ 公式的链接生成语义

$$\mathcal{L}_{Partial}[bfunc(\gamma_1, \cdots, \gamma_n)]_\alpha = \emptyset$$

$$\mathcal{L}_{Entire}[bfunc(\gamma_1, \cdots, \gamma_n)]_\alpha = \emptyset$$