Rule-based Graph Repair using Minimally Restricted Consistency-Improving Transformations

Alexander Lauer

January 10, 2023

Abstract

Contents

1 Introduction

2 Related Work

3 Preliminaries

Our graph repair process is based on the concept of the double-pushout approach [1]. In this chapter we will introduce some formal prerequisites like graphs, graph morphisms, nested graph conditions and constraints and graph transformations.

3.1 Graphs and Graph morphisms

We start by introducing the notions of graphs and graph morphisms according to [1].

Definition 3.1 (**graph**). *A* graph $G = (V, E, \mathrm{src}, \mathrm{tar})$ *consists of a set of vertices (or nodes) V, a set of edges E and two mappings* $\mathrm{src}, \mathrm{tar} : E \to V$ *that assign the source and target vertices to an edge. That means, given an edge $e \in E$, then e connects the vertices* $\mathrm{tar}(e)$ *and* $\mathrm{src}(e)$.

If a tuple as above is not given, V_G, E_G, tar_G *and* src_G *denotes the sets of vertices, edges and the target and source mappings, respectively.*

For the remainder of this paper we assume that all graphs are finite, that means given a graph G the sets V_G and E_G are finite.

Definition 3.2 (**graph morphism**). *Let graphs G and H be given. A* graph morphism *$f : G \to H$ consists of two mappings $f_V : V_G \to V_H$ and $f_E : E_G \to E_H$ such that the source and target functions are preserved. That means,*

$$f_V \circ \mathrm{src}_G = \mathrm{src}_H \circ f_E \wedge$$
$$f_V \circ \mathrm{tar}_G = \mathrm{tar}_G \circ f_E$$

holds. A graph morphism f is called injective (surjective) if f_E and f_V are injective (surjective) mappings. If f is injective, it is denoted by $f : G \hookrightarrow H$ and we may also call it inclusion. *Two morphisms $f_1 : G_1 \to H$ and $f_2 : G_2 \to H$ are called* jointly surjective *if for each element e of H either an element $e' \in G_1$ with $f_1(e') = e$ or an element $e' \in G_2$ with $f_2(e') = e$ exists.*

For our newly introduced notions of consistency increase- and maintainment we also need to consider *sub-graphs*, *overlaps* of graphs and so-called *intermediate-graphs* which intuitively denotes graphs G' that lay in between two given graphs G and H. That means, G is a sub-graph of G' and G' is a sub-graph of H.

Definition 3.3 (**sub-graph**). *Let G and H be graphs. Then, G is called a* sub-graph *of H if an injective morphism $p : G \hookrightarrow H$ exists.*

Note that with this definition every graph G is a sub-graph of itself since the morphism p is also allowed to be surjective.

Definition 3.4 (**intermediate-graph**). *Let G and H be graphs such that G is a sub-graph of H. A graph C is called an* intermediate-graph *of G and H, if G is a sub-graph of C and C is a sub-graph of H. The set of intermediate-graphs of G and H is denoted by* $\mathrm{IG}(G, H)$.

Definition 3.5 (**overlap**). *Let G_1 and G_2 be graphs. A graph H is called an* overlap of G_1 and G_2 *if morphisms $i : G_1 \hookrightarrow H$ and $i' : G_2 \hookrightarrow H$ exist such that i and i' are jointly surjective and $i(G_1) \cap i'(G_2) \neq \emptyset$. The morphisms i and i' are called* overlap morphisms.

The set of all overlaps of G_1 and G_2 is denoted by $\mathrm{ol}(G_1, G_2)$. Given an overlap H of G_1 and G_2, the overlap morphisms are denoted by $i^H_{G_1}$ and $i^H_{G_2}$, respectively.

Note that, graphs H such that jointly surjective morphisms $i : G_1 \hookrightarrow H$ and $i' : G_2 \hookrightarrow H$ such that $i(G_1) \cap i'(G_2) = \emptyset$ can also be considered as overlap of G_1 and G_2. Since, for this paper we will only consider overlaps that satisfy $i(G_1) \cap i'(G_2) \neq \emptyset$, we directly embedded this property into the definition.

As mentioned above, our approach also considers intermediate-graphs. Therefore a notion to restrict given graph morphisms $p : G \to H$ is needed. For this, we introduce the notion of *restricted morphisms* which intuitively is the restriction of the domain and co-domain of p with sub-graphs of G and H, respectively.

Definition 3.6 (**restricted morphism**). *Let graphs G, H and a morphism $f : G \to H$ be given. Then, a morphism $f' : G' \to H'$ is called a* restricted morphism *of p if morphisms $i : G' \hookrightarrow G$ and $i' : H' \hookrightarrow H$ exists (G' is a subgraph of G and H' a sub-graph of H) and*

$$i'_E \circ f'_E = f_E \circ i_E \wedge$$
$$i'_V \circ f'_V = f_V \circ i_V$$

holds. Given an morphism p, we use the notation p^r to denote that p^r is a restricted morphism of p.

Note that, given a morphism $p : G \to H$ a restriction $p^r : G' \to H'$ of p is uniquely determined by G' and H' meaning that, given two restrictions $q : G' \to H'$ and $q' : G' \to H$ of p it follows immediately that $q = q'$.

3.2 Nested Graph Conditions and Constraints

Nested graph constraints are useful to specify graph properties. The more general notion of *nested graph conditions* allows the specification of properties for graph morphisms and the definition of graph constraints in a recursive manner. These conditions allow the use of quantifiers and boolean operators.

Definition 3.7 (**nested graph condition**[2]). *A* nested graph condition *over a graph C_0 is recursively defined as:*

1. true *is a graph condition over every graph.*

2. *$\exists(a_0 : C_0 \hookrightarrow C_1, d)$ is a graph condition over C_0 if a_0 is a injective graph morphism and d is a graph condition over C_1.*

3. *$\neg d$, $d_1 \wedge d_2$ and $d_1 \vee d_2$ are graph conditions over C_0 if d, d_1 and d_2 are graph conditions over C_0.*

Conditions over the empty graph $\emptyset$ are called constraints. *We use the abbreviations $\forall(a_0 : C_0 \hookrightarrow C_1, d) := \neg\exists(a_0 : C_0 \hookrightarrow C_1, \neg d)$ and $\mathsf{false} = \neg\, \mathsf{true}$.*

Conditions of the form $\exists(a_0 : C_0 \hookrightarrow C_1, d)$ are called existentially bound*, the graph C_1 is also called existentially bound. Conditions of the form $\forall(a_0 : C_0 \hookrightarrow C_1, d)$ are called* universally bound*, the graph C_1 is also called universally bound.*

Since these are the only types of conditions used in this paper we will only refer to it as *conditions* and *constraints*.

Definition 3.8 (**semantic of graph conditions**)**.** *Let a graph G, a condition c over C_0 and a graph morphism $p : C_0 \hookrightarrow G$ be given. Then, p satisfies c, denoted by $p \models c$ if*

1. *If $c = \mathsf{true}$, $p \models c$.*

2. *If $c = \exists(a_0 : C_0 \hookrightarrow C_1, d)$, then $p \models c$ if an injective morphism $q : C_1 \hookrightarrow G$ with $p = q \circ a$ exists, such that $q \models d$.*

3. *If $c = \neg d$, then $p \models c$ if $p \not\models d$.*

4. *If $c = d_1 \wedge d_2$, then $p \models c$ if $p \models d_1$ and $p \models d_2$.*

5. *If $c = d_1 \vee d_2$, then $p \models c$ if $p \models d_1$ or $p \models d_2$.*

A graph G satisfies a constraint c, denoted by $G \models c$, if the morphism $p : \emptyset \hookrightarrow G$ satisfies c.

Our approach is designed to repair a specific type of nested constraint, namely *linear conditions* which are conditions without any boolean operators. In particular, our approach is able to repair constraints in *alternating quantifier normal form (ANF)* which are linear conditions with alternating quantifiers, since every linear condition can be transformed into an equivalent condition in ANF [6].

Definition 3.9 (**alternating quantifier normal form (ANF)**[6])**.** *Conditions in* alternating quantifier normal form *(ANF) are inductively defined as*

1. *true and false are conditions in ANF.*

2. *$\exists(a_0 : C_0 \hookrightarrow C_1, d)$ is a condition in ANF if either d is universally bound or $d = \mathsf{true}$.*

3. *$\forall(a_0 : C_0 \hookrightarrow C_1, d)$ is a condition in ANF if either d is existentially bound or $d = \mathsf{false}$.*

In both cases, d is called a sub-condition *of* $\exists(a : C_0 \hookrightarrow C_1, d)$ *or* $\forall(a : C_0 \hookrightarrow C_1, d)$ *respectively. The* nesting level $\mathrm{nl}(c)$ *of a condition c is recursively defined as* $\mathrm{nl}(\textsf{\textit{true}}) = 0$ *and* $\mathrm{nl}(\exists(a : P \to Q, d)) := \mathrm{nl}(d) + 1$.

In literature conditions in ANF allow also conditions that end with $\exists(a_0 : C_0 \hookrightarrow C_1, \textsf{false})$ or $\forall(a_0 : C_0 \hookrightarrow C_1, \textsf{true})$. We exclude these cases such that conditions in ANF are only allowed to end with conditions of the type $\exists(a_0 : C_0 \hookrightarrow C_1, \textsf{true})$ or $\forall(a_0 : C_0 \hookrightarrow C_1, \textsf{false})$ since it can be easily seen that every morphism $p : C_0 \hookrightarrow G$ satisfies $\forall(a_0 : C_0 \hookrightarrow C_1, \textsf{true})$ and does not satisfy $\exists(a_0 : C_0 \hookrightarrow C_1, \textsf{false})$. Therefore, these conditions can be replaced by $\textsf{true}$ and $\textsf{false}$ respectively.

In the following, we assume that all conditions are finite. As a direct consequence of this, the nesting level is also finite.

3.3 Rules and Graph Transformations

Via *rules* and *graph transformation* to manipulate given graphs by the deletion and insertion of edges and vertices. For our approach, we use the concept of the double-pushout approach for rules and transformations [1]. A rule consists of the three graphs L, called the *left-hand side*,K, called *context* and R called *right-hand side* with K being a sub-graph of L and R. During a transformation, denoted by $G \Longrightarrow H$, elements of $L \setminus K$ will be removed and elements of $R \setminus K$ will be inserted such that a new morphism $p : R \hookrightarrow H$ has been created. Additionally, the so-called *dangling edge condition* has to be satisfied, meaning that for each edge $e \in E_H$ there do exists vertices $u, v \in V_H$ such that $\mathrm{tar}(e) = u$ and $\mathrm{src}(e) = v$ or vice versa. Also, we define application conditions which are nested conditions over L and R which are are able prevent transformation if these are not satisfied. We will use the concept of application conditions to guarantee that transformations are not able to decrease consistency. For example, application conditions that prevent a transformation if $G \models c$ and $H \not\models c$, given a constraint c.

Definition 3.10 (**rules and application conditions**)**.** *A* plain rule $\rho' = L \xhookleftarrow{l} K \xhookrightarrow{r} R$ *consists of graphs L, K, R and injective graph morphisms $l : K \hookrightarrow L$ and $r : K \hookrightarrow R$. The rule $\rho'^{-1} = R \xhookleftarrow{r} K \xhookrightarrow{l} L$ is called the* inverse rule of ρ'.

An application condition *is a nested condition over L or R, respectively. A* rule $(\mathrm{ap}_L, \rho', \mathrm{ap}_R)$ *contains of a plain rule ρ' and application conditions* ap_L *over L, called* left application condition, *and* ap_R *over R, called* right application condition, *respectively.*

Definition 3.11 (**graph transformation**)**.** *Let a rule $\rho = (\mathrm{ap}_L, \rho', \mathrm{ap}_R)$, a graph G and a morphism $m : L \hookrightarrow G$, called* match*, be given. Then, a* graph transformation $t : G \Longrightarrow_{\rho,m} H$ *is given in figure 1 if the squares* (1) *and* (2) *are pushouts in the sense of category theory, $m \models \mathrm{ap}_L$ and the morphism $n : L \to H$, called the co-match of t, satisfies* ap_R.

The presence of right applications conditions leads to unpleasant side-effects. The satisfaction of a right application condition can only be checked after the transformation. Therefore, in case that this application condition is not satisfied, the transformation must

$$\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, hook', "l"'] \arrow[r, hook, "r"] \arrow[d, "k"] \arrow[dr, phantom, "(2)"] & R \arrow[d, "n"] \\
G & D \arrow[l, "g"'] \arrow[r, "h"] & H
\end{tikzcd}$$

Figure 1: Diagram of a transformation in the double-pushout approach.

be reversed. To avoid this, we introduce the *shift over rule* construction which is able to transform a right into a left application condition [2]. This notion is based on the notion of *shift over morphism* which allows to shift a condition over C_0 over a morphism $p : C \hookrightarrow C_0$.

Definition 3.12 (**shift over rule**). *Let a plain rule* $\rho = L \xleftarrow{l} K \xrightarrow{r} R$ *and a right application condition* ap *of the form* $\neg\exists(m : R \hookrightarrow C, \textsf{true})$ *be given. Then,* ap *can be shifted into a left application condition with the transformation* $t : C \Longrightarrow_{\rho^{-1},m} C'$*. The shifted condition over* ρ*,* $\text{Left}(\text{ap}, \rho)$*, is given by* $\neg\exists(n : L \hookrightarrow C', \textsf{true})$ *with* n *being the co-match of* t*.*

Shift over rule produces an equivalent left application condition, meaning that, given a right application condition ap and a plain rule ρ , a match of a transformation satisfies $\text{Left}(\text{ap}, \rho)$ if and only if the co-match satisfies ap [2]. In general, every right application condition can be shifted into an equivalent left application condition and vice versa. Since we only need to shift conditions of the form $\neg\exists(a : R \hookrightarrow C, \textsf{true})$ we introduced the shift only for these kind of application conditions and assume that each rule only contains left application conditions, denoted by (ap, ρ).

Via the *track morphism* it is possible to track elements across a transformation [5].

Definition 3.13 (**track morphism**). *Consider the transformation* t *shown in figure 1. The* track morphism*,* $\text{tr}_t : G \to H$*, of* t *is defined as*

$$\text{tr}_t = \begin{cases} h(g^{-1}(e)) & \textit{if } e \in g(D) \\ \textit{undefined} & \textit{otherwise.} \end{cases}$$

For example, given a transformation $t : G \Longrightarrow H$, via the track morphism it can be checked whether a morphism $p : C \hookrightarrow G$ is still present in the derived graph H by checking whether $\text{tr}_t \circ p$ is total, or that a new morphism $q : C \hookrightarrow H$ has been inserted by checking that no morphism $p : C \hookrightarrow H$ with $q = \text{tr}_t \circ p$ exists.

4 Consistency Increase- and Maintainment

Definition 4.1 (**layer of a sub-condition**). *Let* c *be a condition in ANF and* d *a sub-condition of* c*. The* layer of d *is defined as* $\text{lay}(d) := \text{nl}(c) - \text{nl}(d)$*.*

Our approach is based on the idea that the consistency of a constraint is increased layer by layer and even slight improvements like the insertion of single elements of existentially bound graphs should be detectable as increasing. To formalise this, we introduce the notions of *consistency increasing* and *consistency maintaining* transformations and rules with consistency increasing indication that the consistency has indeed been increased and consistency maintaining indicating that the consistency has not been decreased.

4.1 Universally quantified ANF

The definition of consistency increase- and maintainment demands that every existentially bound sub-condition of a given condition in ANF is embedded into an universally bound condition. Otherwise, case discrimination is needed. This requirement is not met if and only if the given condition is existentially bound. Therefore, we will only consider a subset of the set of conditions in ANF, namely the set of universally quantified conditions in ANF, called *universally quantified ANF* (UANF). Additionally, we will show that these sets are expressively equivalent by showing that each condition in ANF can be transformed into an equivalent condition in UANF.

Definition 4.2 (universally quantified alternating quantifier normal form). *A conditions c in ANF is in* universally quantified ANF *(UANF) if it is universally bound.*

Note that, given a condition c in UANF, every sub-condition of c at layer $0 \leq k \leq \mathrm{nl}(c)$ is universally bound if k is an even number and existentially bound if k is an odd number. The extension of conditions that is used to show that a conditions in ANF can be transformed into an equivalent condition in UANF is an already known concept [4].

Lemma 4.3. *Any condition in ANF can be transformed into an equivalent condition in UANF.*

Proof. Let a graph G and a constraint c in ANF be given. If c is universally bound, c is already in UANF.

If $c = \exists(a_0 : C_0 \hookrightarrow C_1, d)$, we show that c is equivalent to $c' := \forall(\mathrm{id}_{C_0} : C_0 \hookrightarrow C_0, c)$.

1. Let $p : C_0 \hookrightarrow G$ be a morphism, such that $q \models c$. Then, $p \models c'$, since p is the only morphism from C_0 to G with $p = p \circ \mathrm{id}_{C_0}$ and $p \models c$.

2. Let $p : C_0 \hookrightarrow G$ be a morphism with $p \models c'$, therefore all morphisms $q : C_0 \hookrightarrow G$ with $p = q \circ \mathrm{id}_{C_0}$ satisfy c. Since $p = p \circ \mathrm{id}_{C_0}$, $p \models c$ follows immediately.

□

For the rest of this thesis, given a condition $c = \forall(a_0 : C_0 \hookrightarrow C_1, d)$ in UANF, we assume that no morphism in c, except a_0, is bijective since it can be shown that each condition in ANF can be transformed into an equivalent condition in ANF fulfilling this property by showing that $\exists(a_0 : C_0 \hookrightarrow C_0, \forall(a_1 : C_0 \hookrightarrow C_2, d)$ is equivalent to $\forall(a_1 \circ a_0 : C_0 \hookrightarrow C_2, d)$ and that $\forall(a_0 : C_0 \hookrightarrow C_0, \exists(a_1 : C_0 \hookrightarrow C_2, d)$ is equivalent to $\exists(a_1 \circ a_0 : C_0 \hookrightarrow C_2, d)$.

4.2 Conditions up to Layer

As already mentioned, the goal of our approach is to increase the consistency of a constraint layer by layer. For this, we introduce a notion of partial consistency, called *satisfaction at layer* which enables to check whether a constraint is satisfied at a certain layer by checking whether the so-called *truncated condition* at this layer is satisfied.

Definition 4.4 (**sub-condition at layer**). *Let c be a condition in ANF. The* sub-condition at layer $0 \leq k \leq \mathrm{nl}(c)$*, denoted by* $\mathrm{sub}_k(c)$ *, is the sub-condition d of c with* $\mathrm{lay}(d) = k$*.*

Example 4.1. *Consider the condition $c = \forall(a_0 : C_0 \hookrightarrow C_1, \exists(a_1 : C_1 \hookrightarrow C_2, \forall(a_2 : C_2 \hookrightarrow C_3, \mathit{false})))$. Then,* $\mathrm{sub}_1(c) = \exists(a_1 : C_1 \hookrightarrow C_2, \forall(a_2 : C_2 \hookrightarrow C_3, \mathit{false}))$*.*

First, we introduce an operator which allows to replace a sub-condition $\mathrm{sub}_k(c)$ by an arbitrary condition over C_k, called *replacement at layer*.

Definition 4.5 (**replacement at layer**)**.** *Let a condition $c = Q(a_0 : C_0 \hookrightarrow C_1, d)$, with $Q \in \{\forall, \exists\}$ in ANF and a condition e over C_k in ANF be given. The* replacement in c at layer k with e*, denoted by* $\mathrm{rep}_k(c, e)$*, is recursively defined as:*

$$\mathrm{rep}_k(c, e) := \begin{cases} e & \text{if } k = 0 \\ Q(a_0 : C_0 \hookrightarrow C_1, \mathrm{rep}_{k-1}(d, e)) & \text{otherwise} \end{cases}$$

Example 4.2. *Let the conditions $c := \forall(a_0 : C_0 \hookrightarrow C_1, \exists(a_1 : C_1 \hookrightarrow C_2, \mathit{true}))$ and $d = \exists(a'_1 : C_1 \hookrightarrow C_3, e)$ be given. Then,*

$$\mathrm{rep}_1(c, d) = \forall(a_0 : C_0 \hookrightarrow C_1, \exists(a'_1 : C_1 \hookrightarrow C_3, e)).$$

Using replacement at layer we now define *truncated conditions*. Intuitively, a condition is-cut off at a certain layer, by replacing the sub-condition at this layer by true or false, depending on the quantifier, the replaced sub-condition is bound by.

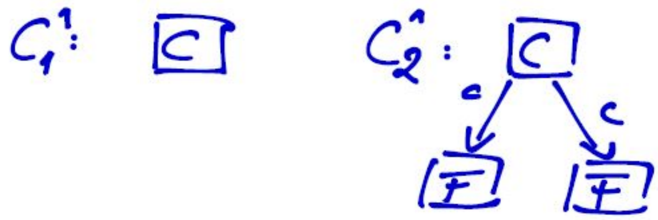
Definition 4.6 (**truncated condition**)**.** *Let c be a condition in UANF and $d = \mathrm{sub}_k(c)$ with $0 \leq k \leq \mathrm{nl}(c) - 1$. The* truncated condition at layer k *of c, denoted by* $\mathrm{cut}_k(c)$*, is defined as*

$$\mathrm{cut}_k(c) := \begin{cases} \mathrm{rep}_{k+1}(c, \mathit{true}) & \text{if } d \text{ is existentially bound, i.e. } k \text{ is odd} \\ \mathrm{rep}_{k+1}(c, \mathit{false}) & \text{if } d \text{ is universally bound, i.e. } k \text{ is even.} \end{cases}$$

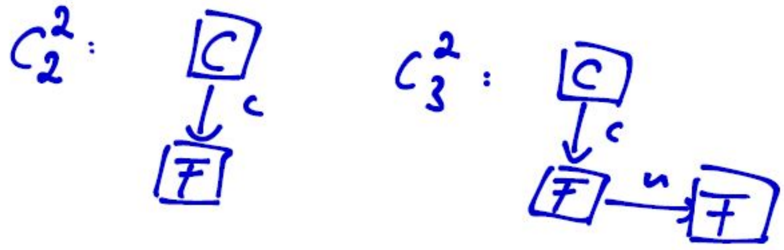
Example 4.3. *Consider constraint c_2 given in Figure 2. Then,* $\mathrm{cut}_1(c_2) = \forall C_1^1 \exists C_2^2$*.*

With these prerequisites we are now able to introduce *satisfaction at layer* which enables to check whether a condition is satisfied at a certain layer. A morphism or graph satisfies a condition or constraint, respectively, if it satisfies the truncated condition at this layer.

$$C_1 = \forall C_1^1 \exists C_2^1$$



$$C_2 = \forall C_1^1 \exists C_2^2 \forall C_3^2 \exists C_4^2$$



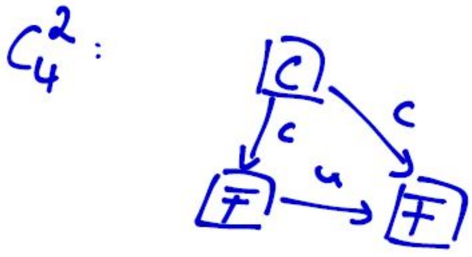


Figure 2: constraints

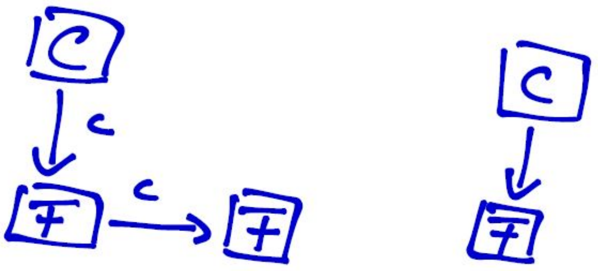


Figure 3: graph

$p \models_k c$	$p \models_{j<k} c$		$p \models_{j>k} c$		$p \models c$
	j even	j odd	j even	j odd	
k even	?	✓	✓	✓	✓
k odd	?	✓	?	?	?

Table 1: Overview of the conclusions made via satisfaction at layer with "✓" indicating that $p \models_j$ and $p \models c$, respectively, if $p \models_k$, And "?" indicating that it cannot be concluded from $p \models_k c$ whether $p \models_j c$ or $p \not\models_j c$.

Definition 4.7 (satisfaction at layer). *Let a graph G and a condition c in UANF be given. A morphism $p : C_0 \hookrightarrow G$* satisfies c at layer $0 \leq k \leq \mathrm{nl}(c) - 1$, *denoted by $p \models_k c$, if*

$$p \models \mathrm{cut}_k(c).$$

A graph G satisfies a constraint c at layer $0 \leq k \leq \mathrm{nl}(c)$, denoted by $G \models_k c$, if $q : \emptyset \hookrightarrow G$ satisfies $\mathrm{cut}_k(c)$. The biggest $0 \leq k \leq \mathrm{nl}(c)$ such that $G \models_k c$ and no $k < j \leq \mathrm{nl}(c)$ with $G \models_j c$ exists is denoted by $\mathrm{k_{max}}(c, G)$. If no such k exists, we set $\mathrm{k_{max}}(c, k) = -1$. We use the abbreviation $\mathrm{k_{max}}$ when c and G are clear from the context.

Note that, given a graph G and a constraint c, by definition and since $\mathrm{nl}(c)$ is finite, $\mathrm{k_{max}}(c, k)$ always exists. Also, if $p \models_{\mathrm{nl}(c)-1} c$ it follows immediately that $p \models c$.

Example 4.4. *Consider the graph G given in Figure 3 and the constraint c_2 given in Figure 2. This graph does not satisfy c_2, since the second occurrence of* `Class` *does not satisfy $\exists C_2^2 \forall C_3^2 \exists C_4^2$, but it satisfies $\mathrm{cut}_1(c_2)$ and therefore*

$$G \models_1 c_2 \text{ and } \mathrm{k_{max}} = 1$$

Let a graph G, a condition c and a morphism $p : C_0 \hookrightarrow G$ be given. Assume that $p \models_k c$ for any $0 \leq k < \mathrm{nl}(c)$. Then, we are able to conclude results for the satisfaction at other layers. If k is even, that means $\mathrm{sub}_k(c)$ is universally bound, we can conclude that $p \models_j c$ for all $k < j < \mathrm{nl}(c)$ and in particular $p \models c$. For any $0 \leq k < \mathrm{nl}(c)$ with $p \models_k c$ we can conclude that $p \models j$ for all odd $0 \leq j < k$. An overview of these conclusion is shown in Table 1. We show these results within the following lemmas.

We start by investigating the conclusions for satisfaction at layer $j > k$ if $p \models_k c$. Our first result shows that the replacement of the sub-condition $\mathrm{sub}_{k+1}(c)$ by any arbitrary condition over C_{k+1} leads to a condition that is satisfied by p if k is even.

Lemma 4.8. *Let a graph G, a condition c in UANF and a morphism $p : C_0 \hookrightarrow G$ with $p \models_k c$, such that $0 \leq k < \mathrm{nl}(c)$ is even, be given. Then, for any condition f over C_{k+1} it holds that*

$$p \models \mathrm{rep}_{k+1}(c, f).$$

Proof. Let $0 \leq j \leq \mathrm{nl}(c)$ be the smallest number with $\mathrm{sub}_j(c) = \forall(a_j : C_j \hookrightarrow C_{j+1}, d)$ being universally bound and $p \models_j c$. This must exist, since at least one of these exists due to the assumption. Let $q : C_j \hookrightarrow G$ be a morphism such that $q \models \forall(a_j : C_j \hookrightarrow C_{j+1}, \mathsf{false})$.

This must exist, since $p \models_j c$ and j is the smallest even number such that $p \models_j c$. Therefore, there does not exist a morphism $q' : C_{j+1} \hookrightarrow G$ with $q = q' \circ a_j$. Hence, for every condition f over C_{j+1} a morphism $q' : C_{j+1} \hookrightarrow G$ with $q \not\models f$ and $q = q' \circ a_j$ cannot exist. It follows immediately that $q \models \forall(a_j : C_j \hookrightarrow C_{j+1}, f)$ and with that $p \models \mathrm{rep}_{j+1}(c, f)$.

We can now conclude that for every even $j < k \leq \mathrm{nl}(c)$, such that $p \models_k c$, and every condition d over C_{k+1} it holds that $p \models \mathrm{rep}_{j+1}(c, f)$ with $f = \mathrm{sub}_{j+1}(\mathrm{rep}_{k+1}(c, d))$. Since $\mathrm{rep}_{j+1}(c, f) = \mathrm{rep}_{k+1}(c, d)$ it follows that $p \models \mathrm{rep}_{k+1}(c, d)$. □

As a direct consequence of the previous lemma, a morphism satisfying a condition at layer k with k being even also satisfies the condition at layer j for all $j > k$.

Lemma 4.9. *Let a graph G, a morphism $p : C_0 \hookrightarrow G$ and a condition c in UANF be given. If $0 \leq k < \mathrm{nl}(c)$ is even, i.e. $\mathrm{sub}_k(c)$ is universally bound, then for all $k < j < \mathrm{nl}(c)$ it holds that*

$$p \models_k c \implies p \models_j c.$$

Proof. Follows immediately by using lemma 4.8 and setting f equal to $\mathrm{sub}_{k+1}(\mathrm{cut}_j(c))$. □

Since a morphism p satisfies a condition c in UANF if and only if p satisfies c at layer $\mathrm{nl}(c) - 1$, because $\mathrm{cut}_{\mathrm{nl}(c)-1}(c) = c$, we can conclude the following.

Corollary 4.10. *Let a graph G, a morphism $p : C_0 \hookrightarrow G$ and a condition c in UANF be given. If $0 \leq k < \mathrm{nl}(c)$ is even it holds that*

$$p \models_k c \implies p \models c.$$

Furthermore, this allows us to make statements about the satisfaction of other conditions. Let a graph G, a morphism $p : C_0 \hookrightarrow G$ and a condition c be given such that $p \models_k c$ for an even $0 \leq k < \mathrm{nl}(c)$. Then, $p \models c$ and for every condition c' with $\mathrm{cut}_k(c) = \mathrm{cut}_k(c')$. With Lemma 4.8, it follows that $p \models c'$.

Let us now investigate the satisfaction at layer j with $j < \mathrm{k_{max}}$. If j is odd, i.e. $\mathrm{sub}_j(c)$ is existentially bound, we can conclude that $p \models_j c$ as shown in Lemma 4.11. If j is even, i.e. $\mathrm{sub}_j(c)$ is universally bound, we are only able to make statements depending on $\mathrm{k_{max}}$. If $\mathrm{k_{max}} < \mathrm{nl}(c) - 1$, it follows that $p \not\models_j c$. Otherwise $p \models c$ and therefore $\mathrm{k_{max}} = \mathrm{nl}(c) - 1$ would follow immediately with Corollary 4.10. If $\mathrm{k_{max}} = \mathrm{nl}(c) - 1$ we can only state that at least one even $j \leq \mathrm{k_{max}}$ with $p \models_j c$ exists if c ends with a condition of the form $\forall(a_k : C_k \hookrightarrow C_{k+1}, \mathsf{false})$.

Lemma 4.11. *Let a graph G, a morphism $p : C_0 \hookrightarrow G$ and a constraint c in UANF be given. Then, for all odd $0 \leq j < \mathrm{k_{max}}$, i.e. $\mathrm{sub}_j(c)$ is existentially bound, it holds that*

$$p \models_j c.$$

Proof. If an even $0 \leq j < \mathrm{k}_{\max}$ with $p \models_j c$ exists such that $\mathrm{sub}_j(c)$ is universally bound, let j' be the smallest of these. With lemma 4.8 follows that $p \models_\ell c$ for all $j' \leq \ell < \mathrm{nl}(c)$. Otherwise, if no such j' exists we set $j' = \mathrm{k}_{\max}$.

Let $\ell < j'$, such that $\mathrm{sub}_\ell(c)$ is existentially bound and let $d = \mathrm{sub}_\ell(\mathrm{cut}_{j'}(c)) = \exists(a_\ell : C_\ell \hookrightarrow C_{\ell+1}, e)$ be the sub-condition at layer ℓ of the condition up to layer j' of c. Since $\ell < j'$, a morphism $q : C_\ell \hookrightarrow G$ with $q \models d$ must exists and therefore a morphism $q' : C_{\ell+1} \hookrightarrow G$ with $q = q' \circ a_\ell$ must exists. It follows that $q \models \exists(a_\ell : C_\ell \hookrightarrow C_{\ell+1}, \mathsf{true})$ and with that $p \models_\ell c$. $\square$

Through satisfaction at layer, an increase of consistency can be detected in the following way: Let $t : G \Longrightarrow H$ be a transformation. If $\mathrm{k}_{\max}(c, G) < \mathrm{k}_{\max}(c, H)$, we consider the transformation as consistency increasing, since H satisfies more layers of the constraint than G. But, the notion of consistency increasement should also be able to detect the smallest changes, performed by a transformation, that lead to an increase of consistency, namely the inserting of a single edge or node of an existentially bound graph. To remedy this issue, we introduce *intermediate conditions*, which will be used to recognize these types of increasements by checking whether an intermediate condition not satisfied by G is satisfied by H. Obviously, an decrease of consistency can be detected in a similar manner, by checking whether an intermediate condition satisfied by G is not satisfied by H. Intuitively, given a constraint c in UANF with $\mathrm{sub}_k(c) = \exists(a_k : C_k \hookrightarrow C_{k+1}, d)$ and $0 \leq k < \mathrm{nl}(c)$, the condition $\mathrm{sub}_k(c)$ is replaced by $\exists(a_k^r : C_k \hookrightarrow C', \mathsf{true})$ with $C' \in \mathrm{IG}(C_k, C_{k+1})$.

The construction of intermediate conditions is designed to only replace graphs in existentially bound layers, since the replacement in an universally bound layer would lead to a more restrictive constraint than the original condition up to layer. That means, given the condition $c = \forall(a_0 : C_0 \hookrightarrow C_1, \mathsf{false})$, let $C' \in \mathrm{IG}(C_0, C_1)$. If the condition $c' = \forall(a_0^r : C_0 \hookrightarrow C', \mathsf{false})$ is satisfied the satisfaction of c is implied but the backwards implication does not hold.

Definition 4.12 (intermediate condition). *Let a condition c in UANF be given. Let $0 \leq k < \mathrm{nl}(c)$ such that k is odd, i.e. $\mathrm{sub}_k(c)$ is existentially bound. The* intermediate condition, *denoted by $\mathrm{IC}_k(c, C')$, of c at layer k with $C' \in \mathrm{IG}(C_k, C_{k+1})$ is defined as*

$$\mathrm{IC}_k(c, C') := \mathrm{rep}_k(c, \exists(a_k^r : C_k \hookrightarrow C', \mathit{true})).$$

Example 4.5. *Consider constraint c_1 given in figure 2. Since $C_2^2 \in \mathrm{IG}(C_1^1, C_2^1)$, we can construct a intermediate condition of c_1 at layer 1 with C_2^2 as $\mathrm{IC}_1(c_1, C_2^2) = \forall C_1^1 \exists C_2^2$. Whereas c_1 checks whether each node of type* `Class` *is connected to at least two nodes of type* `Feature`*, the intermediate condition checks whether each node of type* `Class` *is connected to at least one node of type* `Feature` *which is trivially satisfied if c_1 is satisfied.*

Given a graph G and a constraint c in UANF with $\mathrm{k}_{\max} < \mathrm{nl}(c) - 3$, note that in this case $\mathrm{sub}_{\mathrm{k}_{\max}}(c)$ has to be existentially bound, it holds that $G \not\models_{\mathrm{k}_{\max}+2} c$ and there does exist at least one graph $C' \in \mathrm{IG}(C_{\mathrm{k}_{\max}+2}, C_{\mathrm{k}_{\max}+3})$ such that $G \models \mathrm{IC}_{\mathrm{k}_{\max}+2}(c, C')$ since G always satisfies $\mathrm{IC}_{\mathrm{k}_{\max}+2}(c, C_{\mathrm{k}_{\max}+2})$.

4.3 Consistency Increasing and Maintaining Transformations and Rules

With the results above, we are now ready to define the notions of *consistency increasement* and *maintainment*, with increasement being a special case of maintainment. A transformation t is considered as consistency maintaining if the consistency is not decreased, whereas t is considered as consistency increasing if the consistency has been increased.

These notions are designed to only detect transformations that maintain (or increase) the consistency of the first two unsatisfied layer of a constraint c. That means, given a graph G and a constraint c, let $k = \mathrm{k_{max}} + 1$ and $\mathrm{sub}_k(c) := \forall(a_k : C_k \hookrightarrow C_{k+1}, \exists(a_{k+1} : C_{k+1} \hookrightarrow C_{k+2}, d))$. A transformation $t : G \Longrightarrow H$ is considered as consistency maintaining if $\mathrm{k_{max}}(c, G) \leq \mathrm{k_{max}}(c, H)$, i.e. the satisfaction up to layer is not decreased, and at least the same amount of increasing insertions or deletions have been performed than decreasing ones. An increasing deletion is the deletion of an occurrence of C_{k+1} that does not satisfy $\exists(a_{k+1} : C_{k+1} \hookrightarrow C_{k+2}, \mathsf{true})$, an increasing insertion is the insertion of elements of C_{k+2}, such that for at least one occurrence p of C_{k+1} it holds that $p \not\models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ and $\mathrm{tr}_t \circ p \models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ for a graph $C' \in \mathrm{IG}(C_{k+1}, C_{k+2})$. A decreasing insertion is the creation of an occurrence of C_{k+1} not satisfying $\exists(a_{k+1} : C_{k+1} \hookrightarrow C_{k+2}, \mathsf{true})$ and a decreasing deletion is the deletion of elements of C_{k+2} such that for an occurrence p of C_{k+1} with $p \models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ it holds that $\mathrm{tr}_t \circ p \not\models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ for a graph $C' \in \mathrm{IG}(C_{k+1}, C_{k+2})$. If $\mathrm{k_{max}}(c, G) < \mathrm{k_{max}}(c, H)$ or the number of increasing insertions and deletions is greater than the number of decreasing ones, t is considered as consistency increasing.

To evaluate this, we define the *number of violations*. Intuitively, for all occurrences p of C_{k+1} the number of graphs $C' \in \mathrm{IG}(C_{k+1}, C_{k+2})$ with $p \not\models \exists C'$ is add up and it can be determined whether more increasing or decreasing actions have been performed by a transformation.

Note, that the number of violations is defined for each layer of the constraint, but only for the first unsatisfied layer the sum is calculated as described above. For all layers k with $k \leq \mathrm{k_{max}}$ it is set to 0 and for all layers k with $k > \mathrm{k_{max}} + 1$ it is set to ∞. Through this, a transformation $t : G \Longrightarrow H$ that increases the satisfaction up to layer can easily detected since the number of violations in H at layer $c^G_{\mathrm{max}} + 1$ will be set to 0.

Definition 4.13 (number of violations). *Let a graph G and a constraint c in UANF be given. Let $e = \mathrm{sub}_{\mathrm{k_{max}}+2}(c)$. The* number of violations $\mathrm{nvc}_j(G)$ at layer $0 \leq j < \mathrm{nl}(c)$ in G *is defined as:*

$$\mathrm{nvc}_j(G) := \begin{cases} 0 & \text{if } j < \mathrm{k_{max}} + 1 \\ \sum_{C' \in \mathrm{IG}(C_{j+1}, C_{j+2})} |\{q \mid q : C_{j+1} \hookrightarrow G \wedge q \not\models \mathrm{IC}_0(e, C')\}| & \text{if } e \neq \textsf{false} \text{ and } j = \mathrm{k_{max}} + 1 \\ |\{q \mid q : C_{j+1} \hookrightarrow G\}| & \text{if } e = \textsf{false} \text{ and } j = \mathrm{k_{max}} + 1 \\ \infty & \text{if } j > \mathrm{k_{max}} + 1 \end{cases}$$

Note that the second and third case of Definition 4.13 only apply if $G \not\models c$ and $\mathrm{sub}_{\mathrm{k_{max}}}(c)$ is existentially bound. Therefore e is also existentially bound or equal to false, if c ends with $\forall(a_{\mathrm{nl}(c)-1} : C_{\mathrm{nl}(c)-1} \hookrightarrow C_{\mathrm{nl}(c)}, \mathsf{false})$. Via the number of violations, we now define *consistency maintaining* and *increasing* transformations and rules, by checking whether the number of violations has not been increased, or in case of consistency increasing, has not been decreased for any layer of the constraint.

Definition 4.14 (consistency maintaining and increasing transformations and rules). *Let a graph G, a rule ρ and a constraint c in UANF be given. A transformation $t : G \Longrightarrow_{\rho,m} H$ is called* consistency maintaining w.r.t. c*, if*

$$\mathrm{nvc}_k(H) \leq \mathrm{nvc}_k(G)$$

for all $0 \leq k < \mathrm{nl}(c)$. The transformation t is called consistency increasing w.r.t. c *this inequality is strict. A rule ρ is called* consistency maintaining/increasing w.r.t c*, if all of its transformations are.*

Note that if $G \models c$ there does not exist a consistency increasing transformation $G \Longrightarrow H$ w.r.t c, since $\mathrm{nvc}_j(G) = 0$ for all $0 \leq j < \mathrm{nl}(c)$. Also, no plain rule ρ is consistency increasing w.r.t c, since a graph G satisfying c, such that a transformation $t : G \Longrightarrow_{\rho,m} H$ exists can always be constructed. Therefore, each consistency increasing rule has to be equipped with at least one application condition.

As mentioned above, a transformation should be detected as consistency increasing if it increases the satisfaction up to layer, which is shown by the following theorem.

Theorem 4.1. *Let a graph G, a rule ρ and a constraint c in UANF with $G \not\models c$ be given. A transformation $t : G \Longrightarrow_{\rho,m} H$ is consistency increasing w.r.t. c if*

$$\mathrm{k_{max}}(c, G) < \mathrm{k_{max}}(c, H)$$

.

Proof. No $\ell > \mathrm{k_{max}}(c, G)$ with $G \models_\ell c$ exists. Hence, $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(G) > 0$ and $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(G) \neq \infty$. Since $\mathrm{k_{max}}(c, H) > \mathrm{k_{max}}(c, G)$, $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(H) = 0$ and it follows immediately that t is consistency increasing w.r.t. c. □

Since no consistency increasing transformation originating in consistent graphs exist, there do not exist infinite long sequences of consistency increasing transformations.

Theorem 4.2. *Let a constraint c in UANF be given. Every sequence of consistency increasing transformations w.r.t c is finite.*

Proof. Let G_0 be a graph and

$$G_0 \Longrightarrow_{\rho 0,m_0} G_1 \Longrightarrow_{\rho_1,m_1} G_2 \Longrightarrow_{\rho_3} \dots$$

be a sequence of consistency increasing transformations w.r.t c. We assume that $\mathrm{k_{max}}(c, G_0) < \mathrm{nl}(c)$, otherwise $\mathrm{nvc}_j(G_0) = 0$ for all $0 \leq j < \mathrm{nl}(c)$ and no consistency increasing transformation $G_0 \Longrightarrow H$ w.r.t. c exists.

We show that after at most $j := \mathrm{nvc}_{\mathrm{k}_{\max}(c,G_0)+1}(G_0)$ transformations $G_j \models_{\mathrm{k}_{\max}(c,G_0)+2} c$. Note that j has to be finite, since G_0 contains only a finite number of occurrences of C_{j+1}. Since each transformation is consistency increasing w.r.t. c it holds that $\mathrm{nvc}_{\mathrm{k}_{\max}(c,G_i)+1}(G_{i+1}) \leq \mathrm{nvc}_{\mathrm{k}_{\max}(c,G_i)+1}(G_i) - 1$ after each transformation. Therefore, after at most j transformations, $\mathrm{nvc}_{\mathrm{k}_{\max}(c,G_0)+1}(G_j) \leq \mathrm{nvc}_{\mathrm{k}_{\max}(c,G_0)+1}(G_0) - j = 0$ and $G_j \models_{\mathrm{k}_{\max}(c,G_0)+2} c$. By iteratively applying this, it follows that after a finite number of transformations a graph G_k with $G_k \models c$ has to exist. Since no consistency increasing transformation $G_k \Longrightarrow_{\rho_k,m_k} G_{k+1}$ exists, the sequence has to be finite. □

4.4 Direct Consistency Maintaining and Increasing Transformations

Let a constraint c in UANF and graphs G with $G \not\models c$ and H with $H \models c$ be given. The transformation $t : G \Longrightarrow_{\rho,\mathrm{id}_G} H$ via the rule $\rho = G \xleftarrow{l} \emptyset \xrightarrow{r} H$ is a consistency increasing transformation. Therefore, the notions of consistency increase- and maintainment, a similar example for a consistency maintaining transformation can easily be constructed, does allow insertions or deletions that are unnecessary in order to increase or maintain consistency. That is, the deletion of occurrences of existentially bound graphs, the deletions of occurrences $p : C_k \hookrightarrow G$ of universally bound graphs C_k such that $p \models \exists(a_k : C_k \hookrightarrow C_{k+1}, \mathsf{true})$ or the insertion of occurrences of universally bound graphs and the insertion of occurrences p of intermediate graphs $C' \in \mathrm{IG}(C_{k-1}, C_k)$ with C_k being existentially bound, such that each occurrence q of C_{k-1} with $q = p \circ a^r_{k-1}$ already satisfied $\exists(a^r_{k-1} : C_{k-1} \hookrightarrow C_k, \mathsf{true})$.

Direct consistency increasing and *maintaining* transformations are more restricted, in the sense that these unnecessary deletions and insertions are leading to a transformation not being direct consistency increasing or maintaining, respectively. The presence of these unnecessary actions can be checked via second-order logic formulas. Additionally, it is secured that no new violations are introduced since these can always be considered as a unnecessary insertion or deletion. With that, the removal of one violation is sufficient to state that the transformation is (direct) consistency increasing, which can also be checked via a second order logic formula. We start by introducing *direct consistency maintaining* transformations. Let $t : G \Longrightarrow H$ be a transformation.

Intuitively, formulas (4.1) and (4.2) check that no new violations of the first two unsatisfied layers are introduced, that means, every occurrence p of $C_{\mathrm{k}_{\max}(c,G)+2}$ that is not destroyed by t and satisfies $\exists C'$ still satisfies $\exists C'$ in H with the intermediate graph $C' \in \mathrm{IG}(C_{\mathrm{k}_{\max}(c,G)+2}, C_{\mathrm{k}_{\max}(c,G)+3})$. To check that p has not been destroyed by t, we use the notion of total morphisms, since it has been shown that $\mathrm{tr}_t \circ p$ is total if and only if p has not been destroyed by t [3]. Additionally, every newly inserted occurrence of $C_{\mathrm{k}_{\max}(c,G)+2}$ satisfies d with $d = \mathsf{false}$ if $\mathrm{sub}_{\mathrm{k}_{\max}(c,G)+2}(C) = \mathsf{false}$ and $d = \exists C_{\mathrm{k}_{\max}(c,G)+3}$ otherwise. This case discrimination arises as a consequence of the fact that conditions in UANF are also allows to end with a condition of the form $\forall(a : C_0 \hookrightarrow C_1, \mathsf{false})$. Formulas (4.3) and (4.4) ensure that the satisfaction at layer has not been decreased by checking that no occurrences of universally bound graphs C_j with $j < \mathrm{k}_{\max}$ have been inserted and that no occurrences of existentially bound graphs C_j with $j \leq \mathrm{k}_{\max}$

have been destroyed. Only in these cases, the satisfaction at layer can be decreased. Of course, this does not always lead to a decrease of satisfaction at layer but it can always be considered as an unnecessary insertion or deletion.

The third formula secures that at least one violation has been removed and the last formulas secures that the satisfaction up to layer is not decreased.

Definition 4.15 (**direct consistency maintaining transformations**). *Let G be a graph ρ a rule and c a constraint in UANF. If $G \models c$, a transformation $t : G \Longrightarrow_{\rho,m} H$ is called* direct consistency maintaining w.r.t. c *if $H \models c$. Otherwise, if $G \not\models c$, let $k = \mathrm{k}_{\max}(c, G) + 2$, $e = \mathrm{sub}_k(c)$ and*

$$\mathbf{G} = \begin{cases} \mathrm{IG}(C_k, C_{k+1}) & \text{if } e \neq \textsf{false} \\ \{C_k\} & \text{otherwise} \end{cases}.$$

A transformation $t : G \Longrightarrow_{\rho,m} H$ is called direct consistency maintaining w.r.t. c *if the following equations hold.*

1. *Every occurrence of C_k in G that satisfies $\mathrm{IC}_0(e, C')$ for any $C' \in \mathbf{G}$ still satisfies $\mathrm{IC}_0(e, C')$ in H.*

$$\forall p : C_k \hookrightarrow G\Big(\bigwedge_{C' \in \mathbf{G}} (p \models \mathrm{IC}_0(e, C') \wedge \mathrm{tr}_t \circ p \text{ is total}) \implies \mathrm{tr}_t \circ p \models \mathrm{IC}_0(e, C')\Big) \tag{4.1}$$

2. *Let $d = \mathrm{IC}_0(e, C_{k+1})$ if $e \neq \textsf{false}$ and $d = \textsf{false}$ otherwise. Every newly inserted occurrence of C_k satisfies d.*

$$\forall p' : C_k \hookrightarrow H\big(\neg\exists p : C_k \hookrightarrow G(p' = \mathrm{tr}_t \circ p) \implies p' \models d\big) \tag{4.2}$$

3. *No occurrence of a universally bound graph C_j with $j < \mathrm{k}_{\max}$ gets inserted.*

$$\bigwedge_{\substack{i < \mathrm{k}_{\max} \\ i \text{ even}}} \forall p : C_i \hookrightarrow H(\exists p' : C_i \hookrightarrow G(p = \mathrm{tr}_t \circ p')) \tag{4.3}$$

4. *No occurrence of an existentially bound graph C_j with $j \leq \mathrm{k}_{\max}$ gets deleted.*

$$\bigwedge_{\substack{i \leq \mathrm{k}_{\max} \\ i \text{ odd}}} \forall p : C_i \hookrightarrow G(\mathrm{tr}_t \circ p \text{ is total}) \tag{4.4}$$

Before we continue with the definition of direct consistency increasing, let us first show that each direct consistency maintaining transformation is indeed consistency maintaining. For this, we start by showing that the satisfaction of formulas (4.3) and (4.4) guarantee that the satisfaction at layer has not been decreased.

Lemma 4.16. *Let a transformation $t : G \Longrightarrow H$ and a constraint c in UANF be given, such that (4.3) and (4.4) are satisfied. Then,*

$$H \models_{\mathrm{k}_{\max}(c,G)} c.$$

Proof. Assume that $H \not\models_{\mathrm{k}_{\max}(c,G)} c$. Then, either a new occurrence of an universally bound graph C_k with $k < \mathrm{k}_{\max}(c, G)$ has been inserted or an occurrence of an existentially bound graph C_k with $k \leq \mathrm{k}_{\max}(c, G)$ has been destroyed. Therefore, the following holds:

$$\exists p : C_i \hookrightarrow H(\neg\exists p' : C_i \hookrightarrow G(p = \mathrm{tr}_t \circ p')) \vee \exists p : C_j \hookrightarrow G(\mathrm{tr}_t \circ p \textit{ is not total})$$

with $i, j \leq \mathrm{k}_{\max}(c, G)$, i being even and j being odd, i.e. C_i is universally and C_j is existentially bound. It follows immediately that either (4.3) or (4.4) is not satisfied. This is a contradiction. $\square$

With this, we will now show that direct consistency maintaining transformation is also consistency maintaining.

Theorem 4.3. *Let a graph G, a constraint c in UANF, a rule ρ and a direct consistency maintaining transformation $t : G \Longrightarrow_{\rho,m} H$ w.r.t. c be given. Then, t is also a consistency maintaining transformation.*

Proof. With Lemma 4.16 follows that $\mathrm{k}_{\max}(c, G) \leq \mathrm{k}_{\max}(c, H)$ and it also holds that $\mathrm{nvc}_{\mathrm{k}_{\max}(c,G)+1}(H) \neq \infty$. It remains to show that $\mathrm{nvc}_k(H) \leq \mathrm{nvc}_k(H)$ for all $0 \leq k < \mathrm{nl}(c)$. In particular, we only need to show that $\mathrm{nvc}_{\mathrm{k}_{\max}(c,G)+1}(H) \leq \mathrm{nvc}_{\mathrm{k}_{\max}(c,G)+1}(G)$ since for all $0 \leq j < \mathrm{k}_{\max}(c, G) + 1$ it holds that $\mathrm{nvc}_j(H) = \mathrm{nvc}_j(G) = 0$. Also, since $\mathrm{nvc}_j(G) = \infty$ for all $\mathrm{k}_{\max}(c, G) + 1 < j < \mathrm{nl}(c)$ it follows that $\mathrm{nvc}_j(H) \leq \mathrm{nvc}_j(G)$.

Let $k = \mathrm{k}_{\max}(c, G) + 1$ and $d = \mathrm{sub}_{k+1}(c)$. We show that (4.1) and (4.2) imply that $\mathrm{nvc}_k(H) \leq \mathrm{nvc}_k(G)$. Assume that $\mathrm{nvc}_k(H) > \mathrm{nvc}_k(G)$.

Therefore, a morphism $p : C_{k+1} \hookrightarrow H$ with $p \not\models \mathrm{IC}_0(d, C')$ for any $C' \in \mathrm{IG}(C_{k+1}, C_{k+2})$ exists, such that either 1 or 2 is satisfied. Note that this is only the case if $d \neq \mathsf{false}$. Otherwise, a morphism p satisfying 2 must exist.

1. There does exist a morphism $q' : C_k \hookrightarrow G$ with $q' \models \mathrm{IC}_0(d, C')$ and $p = \mathrm{tr}_t \circ q'$.

2. There does not exist a morphism $q : C_k \hookrightarrow G$ with $p = \mathrm{tr}_t \circ q$.

This is a contradiction, if 1 is satisfied, q' does not satisfy equation (4.1) and if (2) is satisfied q does not satisfy equation (4.2) since q only satisfies $\mathrm{IC}_0(d, C_{k+2})$ if q satisfies $\mathrm{IC}_0(d, C')$ for all $C' \in \mathrm{IG}(C_{k+1}, C_{k+2})$. It follows that

$$\mathrm{nvc}_k(H) \leq \mathrm{nvc}_k(G).$$

Therefore, t is a consistency maintaining transformation.

$\square$

Let us now introduce the notion of *direct consistency increasing* transformations. Similar to the definition of consistency maintaining and increasing transformation, again this notion is based on the notion of direct consistency maintaining transformations, in the sense that a direct consistency increasing transformation is also a direct consistency maintaining one. Since a direct consistency maintaining transformation t does not insert any new violations it is sufficient that t deletes at least one violation to state that t is direct consistency increasing. For this, (4.5) checks that either an occurrence p of $C_{\mathrm{k_{max}}+2}$ has been deleted or $\mathrm{tr}_t \circ p \models \exists C'$ for an intermediate graph $C' \in \mathrm{IG}(C_{\mathrm{k_{max}}+2}, C_{\mathrm{k_{max}}+3})$ such that $p \not\models \exists C'$.

Definition 4.17 (direct consistency increasing). *Let a graph G, a rule ρ and a constraint c in UANF with $G \not\models c$ be given. Let $e = \mathrm{sub}_{\mathrm{k_{max}}(c,G)+2}(c)$ and*

$$\mathbf{G} = \begin{cases} \mathrm{IG}(C_{\mathrm{k_{max}}(c,G)+2}, C_{\mathrm{k_{max}}(c,G)+3}) & \text{if } e \neq \mathsf{false} \\ \{C_k\} & \text{otherwise.} \end{cases}$$

A transformation $t : G \Longrightarrow_{\rho,m} H$ is called direct consistency increasing w.r.t. c *if it is direct consistency maintaining w.r.t. c and the following holds:*

$$\exists p : C_k \hookrightarrow G \Big(\bigvee_{C' \in \mathbf{G}} (p \not\models \mathrm{IC}_0(e, C') \wedge (\mathrm{tr}_t \circ p \textit{ is not total } \vee \mathrm{tr}_t \circ p \models \mathrm{IC}_0(e, C'))) \Big) \quad (4.5)$$

Note that (4.3) and (4.4) not only secure that the satisfaction up to layer does not decrease, as shown in Lemma 4.16, but also prevent further unnecessary insertions and deletions, since the insertion of a universally and the deletion of a existentially bound graph will never lead to an increase of consistency.

Now, we will show the already indicated relationship between direct consistency increasing and consistency increasing, namely that a direct consistency increasing transformation is also consistency increasing. Counterexamples, that the inversion of the implication does not hold can easily be constructed, showing that these notions are not identical but related.

Theorem 4.4. *Let a graph G, a constraint c in UANF with $G \not\models c$, a rule ρ and a direct consistency increasing transformation $t : G \Longrightarrow_{\rho,m} H$ w.r.t. c be given. Then, t is also a consistency increasing transformation.*

Proof. With Theorem 4.3 follows that t is a consistency maintaining transformation. Therefore, it is sufficient to show that $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(H) < \mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(G)$. Let $k = \mathrm{k_{max}}(c, G) + 2$ and $d = \mathrm{sub}_k(c)$ with $d \neq \mathsf{false}$.

Since (4.5) is satisfied, a morphism $p : C_k \hookrightarrow G$ with $p \not\models \mathrm{IC}_0(d, C')$, such that either $\mathrm{tr} \circ p$ is total and $\mathrm{tr}_t \circ p \models \mathrm{IC}_0(d, C')$ or $\mathrm{tr} \circ p$ is not total exists, for a graph $C' \in \mathrm{IG}(C_k, C_{k+1})$. In both cases the following holds:

$$\begin{gathered} p \in \{q \mid q : C_k \hookrightarrow G \wedge q \not\models \mathrm{IC}_0(d, C')\} \wedge \\ \mathrm{tr} \circ p \notin \{q \mid q : C_k \hookrightarrow H \wedge q \not\models \mathrm{IC}_0(d, C')\} \end{gathered}$$

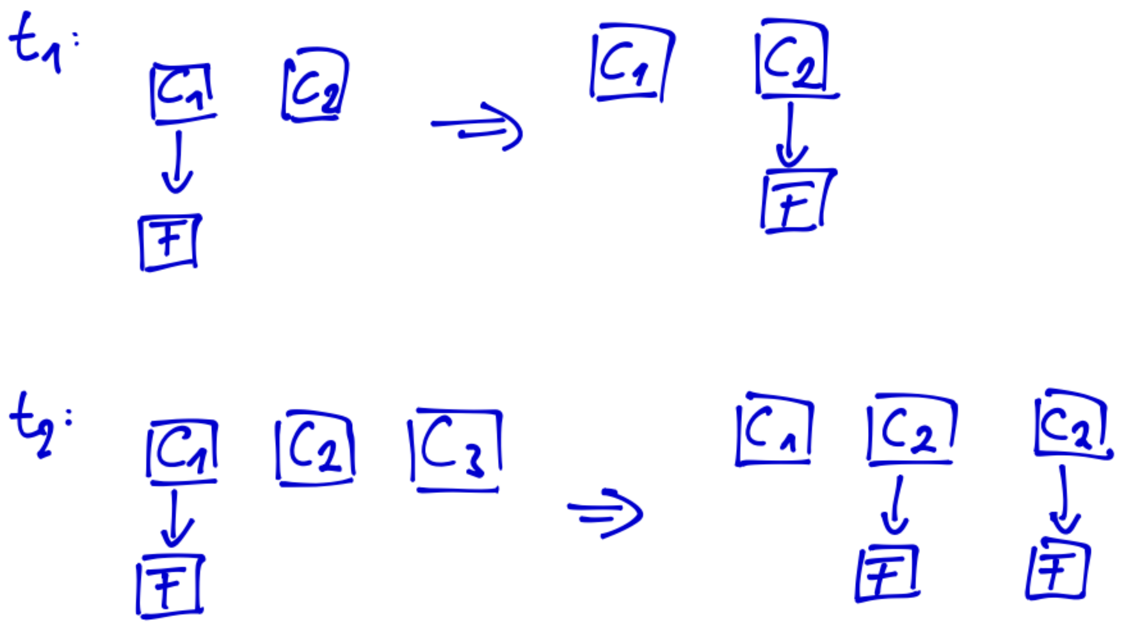


Figure 4: example

Since t is direct consistency maintaining it follows that

$$|\{q \mid q : C_k \hookrightarrow G \wedge q \not\models \mathrm{IC}_0(d, C)\}| \leq |\{q \mid q : C_k \hookrightarrow H \wedge q \not\models \mathrm{IC}_0(d, C)\}|.$$

for all $C \in \mathrm{IG}(C_k, C_{k+1})$. Additionally, this inequality is strictly satisfied if $C = C'$. Therefore, it follows that $\mathrm{nvc}_k(G) < \mathrm{nvc}_k(G)$ and t is a consistency increasing transformation.

If $d = \mathsf{false}$ it holds that

$$|\{q \mid q : C_k \hookrightarrow G\}| \leq |\{q \mid q : C_k \hookrightarrow H\}|$$

since t is a direct consistency maintaining transformation and it can be in shown a similar manner as above that (4.5) implies

$$|\{q \mid q : C_k \hookrightarrow G\}| < |\{q \mid q : C_k \hookrightarrow H\}|.$$

In total it follows that t is a consistency increasing transformation. □

Example 4.6. *Consider the transformations t_1 and t_2 given in Figure 4 and constraint c_1 given in Figure 2. Then, t_1 is a consistency maintaining transformation since the number of violations in both graphs is equal to 2. But, t_1 is not a direct consistency maintaining transformation since one occurrence of a node of type* `Class` *satisfying $\exists C_2^2$ in the first but not in the second graph of the transformation exists. Therefore (4.1) is not satisfied.*

$$\begin{tikzcd}
 & c\text{-guaranteeing} \arrow[dl, Rightarrow] \arrow[d, Rightarrow] & \\
c\text{-preserving} & \text{maintaining w.r.t. } c \arrow[l, Rightarrow] & \text{increasing w.r.t. } c \arrow[l, Rightarrow] \\
 & \text{direct maintaining w.r.t. } c \arrow[u, Rightarrow] & \text{direct increasing w.r.t. } c \arrow[l, Rightarrow] \arrow[u, Rightarrow]
\end{tikzcd}$$

Figure 5: Relations of consistency notions.

The transformation t_2 is consistency increasing w.r.t. c_1 since the number of violations is equal to 4 in the first and equal to 2 in the second graph. But, t_2 is not a direct consistency increasing transformation since (4.1) is not satisfied.

4.5 Comparison with other concepts of Consistency

In this chapter, the notions of (direct) consistency increase- and maintainment are compared to the already known concepts of consistency guaranteeing, consistency preserving [2], (direct) consistency increasing and sustaining [3] in order to reveal relationships between them and ensure that (direct) consistency increase- and maintainment are indeed a new notions of consistency. These relations are displayed in Figure 5.

First, we compare (direct) consistency increase- and maintainment with the notions of consistency-guaranteeing and -preserving. We start by showing that consistency maintaining implies preserving but the backwards implication does not hold.

Lemma 4.18. *Let a constraint c in UANF, graphs G and H and a transformation $t : G \Longrightarrow H$ be given. Then,*

$$\begin{aligned}
&t \text{ is maintaining w.r.t. } c \Longrightarrow t \text{ is } c\text{-preserving} \qquad \text{and}\\
&t\ c\text{-preserving} \quad\not\Longrightarrow t \text{ is maintaining w.r.t. } c
\end{aligned}$$

Proof. 1. Let t be a consistency maintaining transformation w.r.t. c. If $G \not\models c$, t is a c-preserving transformation. If $G \models c$, it holds that $\text{nvc}_j(G) = 0$ for all $0 \leq j < \text{nl}(c)$. Since t is consistency maintaining it follows that $\text{nvc}_j(H) = 0$ for all $0 \leq j < \text{nl}(c)$ and therefore $H \models c$. It follows that t is a c-preserving transformation.

2. Consider graphs C_1^1, C_2^2 and constraint c_1 given in Figure 2. Then, the transformation $t : C_2^2 \Longrightarrow C_1^1$ is c-preserving, since $C_2^2 \not\models c_1$, but not consistency maintaining w.r.t. c since $\text{nvc}_0(C_2^2) = 2$ and $\text{nvc}_0(C_1^1) = 5$.

□

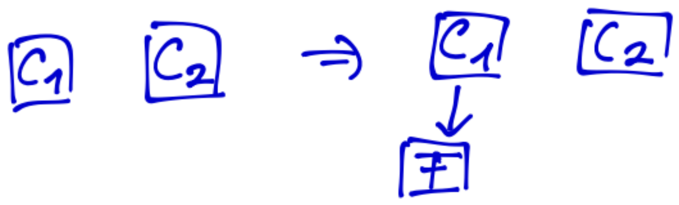


Figure 6: example

Obviously, guaranteeing implies consistency maintaining, since this property is embedded in the definition of consistency maintainment. The inversion of this implication does not holds, since maintaining is a way stricter notion, in the sense, that the number of removed violations has to be greater or equal than the number of introduced violations. For guaranteeing transformations this is not the case, an arbitrary number of violations can be inserted, as long as the derived graph satisfies the constraint and therefore guaranteeing does not imply direct increasing, since a direct increasing transformations is not allowed to introduce any new violations.

Lemma 4.19. *Let a constraint c in UANF, graphs G and H and a transformation $t : G \Longrightarrow H$ be given. Then,*

$$\begin{array}{lll} t \text{ is } c\text{-guaranteeing} & \Longrightarrow & t \text{ is maintaining w.r.t } c \quad \text{and} \\ t \text{ is } c\text{-guaranteeing} & \not\Longrightarrow & t \text{ is direct maintaining w.r.t } c \text{ and} \\ t \text{ is maintaining w.r.t. } c & \not\Longrightarrow & t \text{ is } c\text{-guaranteeing} \end{array}$$

Proof. 1. Let t be a c-guaranteeing transformation, then $H \models c$. It holds that $\text{nvc}_j(H) = 0$ for all $0 \leq j < \text{nl}(c)$ and since the number of violations cannot be negative, t is a maintaining transformation w.r.t. c.

2. Let t be a c-guaranteeing transformation. Then, t is also a c-preserving transformation and since each direct maintaining transformation is also a maintaining one the statement follows directly with Lemma 4.18.

3. Consider the transformation $t : G \Longrightarrow H$ shown in Figure 6 and constraint c_1 shown in Figure 2. Then, t is a maintaining transformation w.r.t c, in particular, c is a consistency increasing transformation w.r.t. c since $\text{nvc}_0(G) = 10$ and $\text{nvc}_0(H) = 7$. But, t is not c-guaranteeing since both occurrences of nodes of type `Class` do not satisfy $\exists C_2^1$.

□

Let $t : G \Longrightarrow H$ be a c-guaranteeing transformation. If $G \models c$, the transformation is, by definition, not consistency increasing. If $G \not\models c$, t is always also a consistency increasing transformation w.r.t. c.

Lemma 4.20. *Let graphs G, H, a constraint c with $G \not\models c$ and a transformation $t : G \Longrightarrow H$ be given. Then*

$$t \textit{ is c-guaranteeing} \quad \Longrightarrow \quad t \textit{ is increasing w.r.t } c.$$

Proof. Let t be a c-guaranteeing transformation, then $H \models c$. Since $G \not\models c$, it holds that $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(G) > 0$ and $\mathrm{nvc}_{\mathrm{k_{max}}(c,G)+1}(H) = 0$. Therefore, t is consistency increasing. □

The definition of consistency improvement only differs from guaranteeing if the corresponding constraint is universally bound and these notions are identical for existentially bound constraint. Therefore, with Lemmas 4.19 and 4.20, we can state the following.

Corollary 4.21. *Let an existentially bound constraint c in ANF and a transformation $t : G \Longrightarrow H$ be given. Then,*

$$\begin{aligned} &t \textit{ is consistency improving w.r.t } c && \Longrightarrow \; t \textit{ is consistency maintaining w.r.t } c \quad \textit{and} \\ &t \textit{ is consistency maintaining w.r.t } c && \not\Longrightarrow \; t \textit{ is consistency improving w.r.t } c. \end{aligned}$$

If $G \not\models c$, we can also state that

$$t \textit{ is consistency improving w.r.t } c \quad \Longrightarrow \quad t \textit{ is consistency increasing w.r.t } c$$

The notions of increae- and improvement are equivalent for universally bound constraints with nesting level 1. Note that, with corollary 4.21, this equivalence does not hold for existentially bound constraints with nesting level 1.

Lemma 4.22. *Let a universally bound constraint c in UANF with $\mathrm{nl}(c) = 1$, a graph G with $G \not\models c$ and a transformation $t : G \Longrightarrow H$ be given. Then,*

$$t \textit{ is consistency improving w.r.t } c \quad \Longleftrightarrow \quad t \textit{ is consistency increasing w.r.t } c$$

Proof. Let $c = \forall(a : \emptyset \hookrightarrow C, \mathsf{false})$. Since $\mathrm{sub}_1(c) = \mathsf{false}$, $\mathrm{nvc}_0(G)$ is the number of occurrences of C in G. This is exactly the definition of the number of violations for consistency improving transformations and the statement follows immediately. □

For universally bound constraints c with $\mathrm{nl}(c) \geq 2$, the notions of (direct) consistency increase- and maintainment are not related to (direct) consistency improve- and sustainment. By definition, (direct) consistency improvement implies (direct) consistency sustainment [3]. Therefore, it is sufficient to show that direct improvement does not imply maintainment and that direct increasement does not imply consistency sustainment.

Lemma 4.23. *Let a universally bound constraint c in UANF with $\mathrm{nl}(c) \geq 2$, a graph G with $G \not\models c$ and a transformation $t : G \Longrightarrow H$ be given. Then,*

$$\begin{aligned} &t \textit{ is direct consistency improving w.r.t } c && \not\Longrightarrow \; t \textit{ is consistency maintaining w.r.t } c \quad \textit{and} \\ &t \textit{ is direct increasing w.r.t } c && \not\Longrightarrow \; t \textit{ is consistency sustaining w.r.t } c \end{aligned}$$

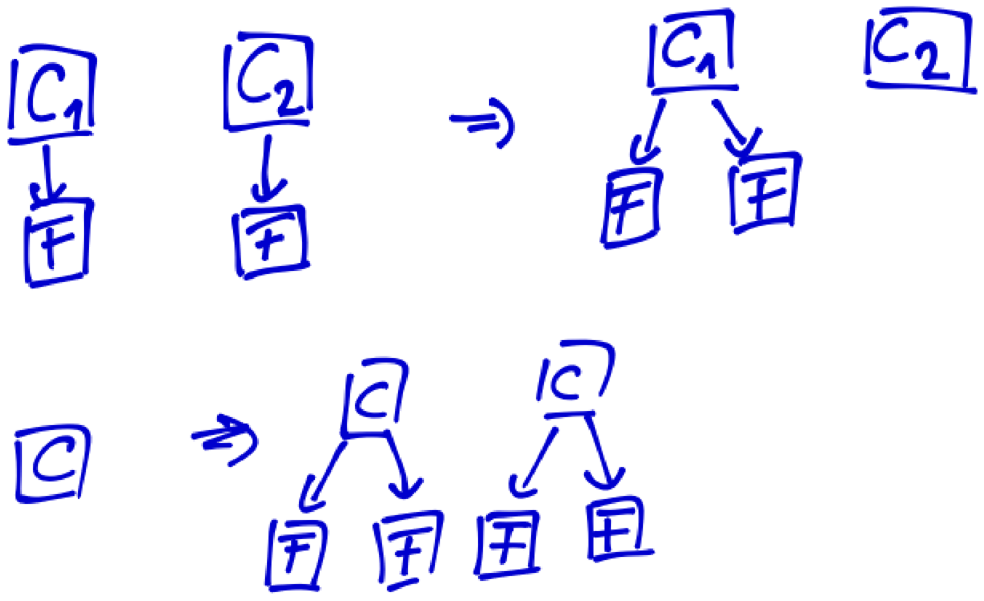


Figure 7: example

Proof. 1. Consider transformation t_1 given in Figure 7 and constraint c_1 given in Figure 2. The transformation t_1 is direct consistency improving but not maintaining since $\text{nvc}_0(G) = 4$ and $\text{nvc}_0(H) = 5$.

2. Consider the constraint $c = \forall(C_1^1, \exists(C_2^1, \forall(C_4^2, d)))$ with d being an existentially bound constraint in ANF with $d \neq \mathsf{false}$ composed of the graphs given in Figure 2 and transformation t_2 given in Figure 7. Then, t is direct consistency increasing, (4.1), (4.2), (4.3) and (4.4) are trivially satisfied and (4.5) is satisfied since one occurrence of C_1^1 that did not satisfy $\exists C_2^1$ in G satisfies $\exists C_2^1$ in H, but not consistency sustaining since the number of occurrences of C_1^1 that not satisfying $\exists(C_2^1, \forall(C_4^2, d))$ in H is greater than the number of occurrences of C_1^1 in G not satisfying $\exists(C_2^1, \forall(C_4^2, d))$.

□

5 Application Conditions

To guarantee that each transformation t is (direct) consistency increasing or maintaining w.r.t to a constraint c, we present applications conditions ensuring this property. Given a constraint c, this application conditions are designed to only consider graph of c up to a certain layer. In particular, this is useful to reduce the restrictiveness of these application conditions, since all graphs C_j of c with $j > \mathrm{k}_{\max} + 2$ do not affect whether an transformation is considered as consistency maintaining or increasing. Additionally, in the case of consistency increasing application conditions, this design is necessary

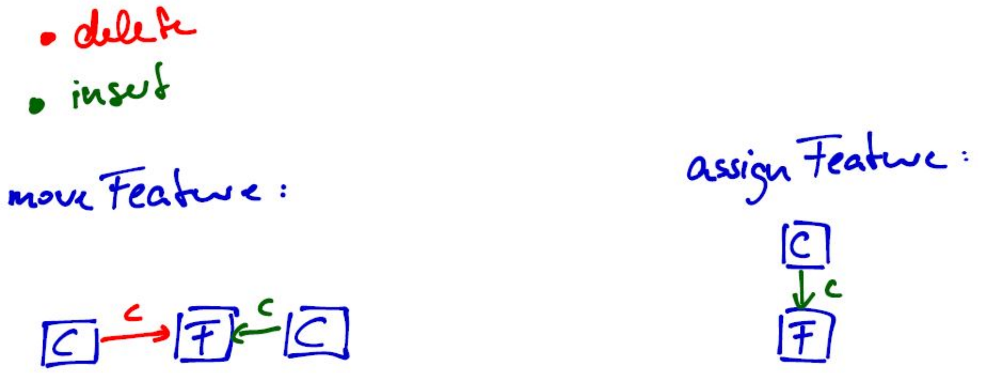
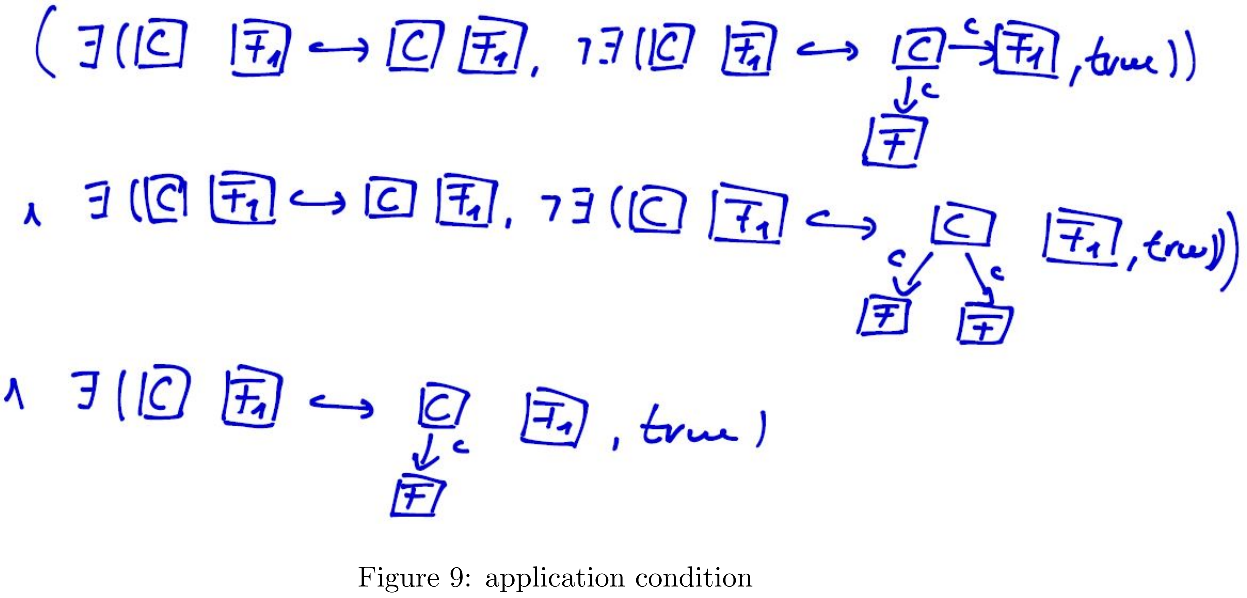


Figure 8: rules



(∃(C F1 ↪ C F1, ¬∃(C F1 ↪ C →c F1, true))
c
F
∧ ∃(C F1 ↪ C F1, ¬∃(C F1 ↪ C F1, true))
c c
F F
∧ ∃(C F1 ↪ C F1, true)
c
F

Figure 9: application condition

since it has to be ensured that violations at layer $\mathrm{k}_{\max}+1$ will be removed. Therefore, let us introduce a weaker notion of consistency increasing and maintaining rules, namely *consistency increasing and maintaining rules at layer*. As the name suggests, given a constraint c, a rule is consistency increasing or maintaining at layer $0 \leq k < \mathrm{nl}(c)$ if all of its applications at graphs G, with $\mathrm{k}_{\max} = k$, are consistency increasing or maintaining w.r.t. c, respectively.

Definition 5.1 (consistency increasing and maintaining rule at layer). *Let a constraint c be given. A rule ρ is called* consistency maintaining at layer $0 \leq k < \mathrm{nl}(c)$ w.r.t. c *if all transformations $t : G \Longrightarrow_\rho H$, with $\mathrm{k}_{\max}(G, c) = k$, are consistency maintaining w.r.t. c. Additionally, ρ is called* consistency increasing at layer $0 \leq k < \mathrm{nl}(c)$ w.r.t. c *if all transformations $t : G \Longrightarrow_\rho H$, with $\mathrm{k}_{\max}(G, c) = k$, are consistency increasing w.r.t. c.*

Note, that a consistency maintaining rule at layer $\mathrm{nl}(c)-1$ w.r.t. c is also a consistency maintaining or increasing rule w.r.t. c. But, a consistency increasing rule at layer $\mathrm{nl}(c)-1$ must not necessarily be a consistency increasing rule w.r.t. c.

5.1 General Application Conditions

We start by introducing consistency maintaining application conditions, i.e. a rule equipped with this application condition will only be applicable if the corresponding transformation is consistency maintaining. In particular, we will show that this transformation is even direct consistency maintaining.

The maintaining application condition consists of the three parts $\mathrm{ned}_k(\rho')$, $\mathrm{nui}_k(\rho')$ and $\mathrm{nds}_k(\rho')$ that are connected via the boolean $\wedge$−operator with ρ' being a plain rule and $0 \leq k < \mathrm{nl}(c)$ for a given constraint c. As already discussed, the satisfaction at layer is decreased if and only if occurrences of existentially bound graphs C_j have been deleted or occurrences of universally bound graphs C_j have been inserted, with $0 \leq j \leq k$. To check that no existentially bound graph C_j, $0 \leq j \leq k$, will be deleted, we use the condition constructed by $\mathrm{ned}_k(\rho')$ that check that no overlap P of the left-hand-side of ρ' with C_j, such that the application of ρ' at P leads to a deletion of C_j, exists in the originating graph. To ensure that no universally bound graph C_j, $0 \leq j \leq k$, will be inserted, the condition constructed by $\mathrm{nui}_k(\rho')$ checks that no overlap P of the right-hand-side of ρ' and C_j, such that the application of ρ'^{-1} at P would destroy the occurrence of C_j, exists in the derived graph. Note, that a condition constructed in this way is a right application condition. Therefore, we use the shift over rule operator to construct an equivalent left application condition. To verify that the number of violations is not decreased, we use the condition constructed by $\mathrm{nds}_k(\rho')$, that checks, in the same manner as $\mathrm{nui}_k(\rho')$, that no occurrences of graphs $C' \in \mathrm{IG}(C_{k+2}, C_{k+3})$ will be deleted if $k < \mathrm{nl}(c) - 2$. Otherwise, this condition is set to `true`.

Definition 5.2 (consistency maintaining application condition). *Let a rule $\rho = (\mathrm{ac}, \rho')$ with $\rho' = L \hookleftarrow K \hookrightarrow R$ and a constraint c in UANF be given. The* maintaining application condition *of c for ρ at layer $0 \leq k < \mathrm{nl}(c)$ is defined as* $\mathrm{ac} \wedge \mathrm{main}_k(\rho')$ *with*

$$\mathrm{main}_k(\rho') := \mathrm{ned}_k(\rho') \wedge \mathrm{nui}_k(\rho') \wedge \mathrm{nds}_k(\rho')$$

and

1. *Let E be the set of all existentially bound graphs graphs C_j with $j \leq k+1$ and $\mathbf{P}_{C_j}$ be the set all overlaps P' of L and C_j with $i_L^{P'}(L \setminus K) \cap i_{C_j}^{P'}(C_j) \neq \emptyset$:*

$$\mathrm{ned}_k(\rho') := \bigwedge_{C \in E} \bigwedge_{P' \in \mathbf{P}_{C_j}} \neg\exists(i_L^{P'} : L \hookrightarrow P', \mathit{true})$$

2. *Let U be the set of all universally bound graphs C_j with $j \leq k+2$, and $\mathbf{P}_{C_j}$ be the set of all overlaps P' of R and C_j with $i_R^{P'}(R \setminus K) \cap i_{C_j}^{P'}(C_j) \neq \emptyset$:*

$$\mathrm{nui}_k(\rho') := \bigwedge_{C \in U} \bigwedge_{P' \in \mathbf{P}_{C_j}} \mathrm{Left}(\neg\exists(i_R^{P'} : R \hookrightarrow P', \mathit{true}), \rho')$$

3. *If $k > \mathrm{nl}(c) - 3$ or $\mathrm{sub}_k(c)$ is universally bound, $\mathrm{nds}_k(\rho') = \mathit{true}$. Otherwise, Let E be the set of all overlaps of L and C' with $i_L^{P'}(L \setminus K) \cap i_{C'}^{P'}(C') \neq \emptyset$ for all $C' \in \mathrm{IG}(C_{k+2}, C_{k+3})$:*

$$\mathrm{nds}_k(\rho') := \bigwedge_{P \in E} \neg\exists(i_L^P : L \hookrightarrow P', \mathit{true})$$

Example 5.1.

Let us now show that each rule equipped with the according application condition is a consistency maintaining rule at layer.

Theorem 5.1. *Let a constraint c in UANF be given. Each rule $\rho = (\mathrm{ac}', \rho')$ with $\mathrm{ac}' = \mathrm{ac} \wedge \mathrm{main}_k(\rho')$ and $0 \leq k < \mathrm{nl}(c)$ is a consistency maintaining rule at layer k w.r.t. c.*

Proof. Let a graph G, such that $\mathrm{k}_{\max} = k$, and a transformation $t : G \Longrightarrow_\rho H$ be given. It is sufficient to show that t is a direct consistency maintaining transformation. If $k < \mathrm{nl}(c) - 1$, it follows that $G \not\models c$ and that k is odd. We show that t satisfies (4.1), (4.2), (4.3) and (4.4).

1. Assume that (4.1) does not hold. Then, $e = \mathrm{sub}_{k+2}(c) \neq \mathsf{false}$ and a morphism $p : C_{k+2} \hookrightarrow G$ exists, such that $p \models \mathrm{IC}_0(e, C')$, $\mathrm{tr}_t \circ p$ is total and $\mathrm{tr}_t \circ p \not\models \mathrm{IC}_0(e, C')$ for a graph $C' \in \mathrm{IG}(C_{k+2}, C_{k+3})$. Therefore, an overlap P of L and C' such that $i_{C_{k+2}}^P \models \exists(a_{k+2}^r : C_{k+2} \hookrightarrow C', \mathsf{true})$ with $i_L^P(L \setminus K) \cap i_{C'}^P(C' \setminus C_{k+2}) \neq \emptyset$ must exist and $m \models \exists(i_L^P : L \hookrightarrow P, \mathsf{true})$ holds. Thus, $\mathrm{nds}_k(\rho')$ and consequently also $\mathrm{main}_k(\rho')$ cannot be satisfied.

2. Assume that (4.2) does not hold and let
$$d := \begin{cases} \mathrm{IC}_0(\mathrm{sub}_{k+2}(c), C_{k+3}) & \text{if } \mathrm{sub}_{k+2}(c) \neq \mathsf{false} \\ \mathsf{false} & \text{otherwise.} \end{cases}$$
Then, a morphism $p' : C_{k+2} \hookrightarrow H$ with $p' \not\models d$ exists, such that no morphism $p : C_{k+2} \hookrightarrow G$ with $\mathrm{tr}_t \circ p = p'$ exists. Therefore, an overlap P of R and C_{k+2} with $i_R^P(R \setminus K) \cap i_{C_{k+2}}^P(C_{k+2}) \neq \emptyset$ exists, such that $m \models \mathrm{Left}(\exists(i_R^P : R \hookrightarrow P, \mathsf{true}), \rho')$. Hence, m does not satisfy $\mathrm{nui}_k(\rho')$.

3. Assume that (4.3) does not hold. Then, a morphism $p : C_j \hookrightarrow H$ with C_j being universally bound and $j < k$ exists, such that no morphism $p' : C_j \hookrightarrow G$ with $\mathrm{tr}_t \circ p' = p$ exists. Then, an overlap P of C_j and R with $i_R^P(R \setminus K) \cap i_{C_j}^P(C_j) \neq \emptyset$ exists, such that $m \models \mathrm{Left}(\exists(i_R^P : R \hookrightarrow P, \mathsf{true}), \rho)$. Hence, $m \not\models \mathrm{nui}_k(\rho')$.

4. Assume that (4.4) does not hold. Then, a morphism $p : C_j \hookrightarrow G$ with C_j being existentially bound and $j \leq k$ exists, such that $\mathrm{tr}_t \circ p$ is not total. Then, an overlap P of C_j and L with $i_L^P(L \setminus K) \cap i_{C_j}^P(C_j) \neq \emptyset$ exists, such that $m \models \exists(i_L^P : L \hookrightarrow P, \mathsf{true})$. Hence, $m \not\models \mathrm{ned}_k(\rho')$.

If $k = \mathrm{nl}(c) - 1$, it follows immediately that $G \models c$. Therefore, we need to show that $H \models c$. It follows that $\mathrm{nds}_k(\rho') = \mathsf{true}$. Assume that $H \not\models c$. Therefore, either an occurrence $p : C_j \hookrightarrow H$ of an universally bound graph C_j exists such that no $q : C_j \hookrightarrow G$ with $p = \mathrm{tr}_t \circ q$ exists or an occurrence $p' : C_{j'} \hookrightarrow G$ of an existentially bound graph $C_{j'}$ exists, such that $\mathrm{tr}_t \circ p'$ is not total. If the first case applies, an overlap P of C_j and R with $i_{C_j}^P(C_j) \cap i_R^P(R \setminus K) \neq \emptyset$ exists, such that $m \models \mathrm{Left}(\neg\exists(i_R^P : R \hookrightarrow P, \mathsf{true}), \rho')$ and therefore $m \not\models \mathrm{nui}_k(\rho')$. If the second case applies, an overlap P of $C_{j'}$ and L with $i_{C_{j'}}^P(C_{j'}) \cap i_L^P(L \setminus K) \neq \emptyset$ exists, such that $m \models \neg\exists(i_L^P L \hookrightarrow P, \mathsf{true})$ and therefore $m \not\models \mathrm{ned}_k(\rho')$. By contradiction follows that $H \models c$.

In total follows that ρ is a consistency maintaining rule at layer k w.r.t. c. □

For an application condition, such that a rule equipped with it is consistency increasing at layer, we have additionally to ensure that at least one violation will be removed. To check this via an application condition, it is necessary (a) to check that an occurrence p of the universally bound graph $C_{\mathrm{k_{max}}+2}$ exists, such that p and the match m do overlap, i.e $p(C_{\mathrm{k_{max}}+2}) \cap m(L) \neq \emptyset$, and, if the sub-condition at layer $\mathrm{k_{max}}+2$ is not equal to false, (b) that p does not satisfy $c' := \exists C_{\mathrm{k_{max}}+3}$. Only in this case, it is possible that the transformation does remove a violation. To ensure that p does not satisfy c', the non-existence of all possible overlaps P of L and $C_{\mathrm{k_{max}}+3}$ such that $p \models c'$ has to be checked. For this, we introduce *extended overlaps*. Intuitively, given an overlap C of L and $C_{\mathrm{k_{max}}+2}$ with $p : C_{\mathrm{k_{max}}+2} \hookrightarrow C$, the overlap is extended with elements of $C_{\mathrm{k_{max}}+3}$ such that $p \models C_{\mathrm{k_{max}}+3}$.

Definition 5.3 (**extended overlaps**). *Let G, C_0 and C_1 with $i_{C_0}^{C_1} : C_0 \hookrightarrow C_1$ be graphs. Let P be an overlap of C_0 and G with the inclusion $i_{C_0}^{P} : C_0 \hookrightarrow P$. The set of* extended overlaps of P with $i_{C_0}^{C_1}$, *denoted by* $\mathrm{eol}(C, i_{C_0}^{C_1})$, *is the set of all overlaps Q of G and C_1, such that an injective morphism $i_P^Q : P \hookrightarrow Q$ with $i_P^Q \circ i_{C_0}^P \models \exists(i_{C_0}^{C_1} : C_0 \hookrightarrow C_1, \textsf{true})$ exists.*

Via the notion of extended overlaps we are now able to check that a violation exists, to decide whether a transformation is able to remove a violation at all. It remains to check whether such a violation will be removed.

In the definition below, $\mathrm{exv}(\cdot,\cdot)$ and $\mathrm{remv}(\cdot,\cdot)$ ensure that a violation will be removed, with $\mathrm{exv}(\cdot,\cdot)$ ensuring that a violation is present and $\mathrm{remv}(\cdot,\cdot)$ ensuring that this violation will be removed. Note, that the construction of these is divided in two cases. Firstly, either $k \leq \mathrm{nl}(c) - 3$ and secondly, $k = \mathrm{nl}(c) - 2$ and the constraint ends with a condition of the form $\forall(a : C_0 \hookrightarrow C_1, \textsf{false})$.

For $\mathrm{exv}(\cdot,\cdot)$, the first case will be checked via extended overlaps as already described above. In the second case, it is sufficient to check whether an occurrence p of $C_{\mathrm{nl}(c)}$ with $m(L) \cap p(C_{\mathrm{nl}(c)}) \neq \emptyset$ exists.

For $\mathrm{remv}(\cdot,\cdot)$, in the first case, a violation can be removed by either deleting an occurrence p of C_{k+2} or inserting elements of C_{k+3}, such that $p \not\models \exists C'$ and $\mathrm{tr}_t \circ p \models \exists C'$ for a graph $C' \in \mathrm{IG}(C_k k+2, C_{k+3})$. In the second case, a violation can only be removed by deleting an occurrence p of C_1. This will only occur if $m(L \setminus K) \cap p(C_1) \neq \emptyset$.

Definition 5.4 (**consistency increasing application condition**). *Let a rule $\rho = (\mathrm{ac}, \rho'$ with $\rho' = L \hookleftarrow K \hookrightarrow R$ and a constraint c in UANF be given. Let $0 \leq k < \mathrm{nl}(c)$ be even, i.e. $\mathrm{sub}_k(c)$ is universally bound, and $C \in \mathrm{IG}(C_{k+1}, C_{k+2})$ if $\mathrm{sub}_{k+1}(c) \neq \textsf{false}$ and $C = C_{k+1}$ otherwise. The* increasing application condition of c for ρ at layer k with C *is defined as*

$$\mathrm{incr}_k(C, \rho) := \mathrm{ac} \wedge \mathrm{main}_{k-1}(\rho) \wedge \big(\bigvee_{P \in \mathrm{ol}(L, C_{k+1})} \mathrm{exv}(P, C) \wedge \mathrm{remv}(P, C) \big) \tag{5.1}$$

with

1. *Let $a^r : C_{k+1} \hookrightarrow C$ be the restricted morphism of a_{k+1} and i_L^P and i_P^Q the inclusions of L in P and P in Q, respectively:*

$$\mathrm{exv}(P, C') := \begin{cases} \exists(i_L^P : L \hookrightarrow P, \textsf{true}) & \text{if } \mathrm{sub}_{k+1}(c) = \textsf{false} \\ \bigwedge_{Q \in \mathrm{eol}(P, a^r)} \exists(i_L^P : L \hookrightarrow P, \neg\exists(i_P^Q : P \hookrightarrow Q, \textsf{true})) & \text{otherwise} \end{cases}$$

2. *If $i_L^P(L \setminus K) \cap i_{C_{k+1}}^P(C_{k+1}) \neq \emptyset$, we set*

$$\mathrm{remv}(P, C') := \textsf{true}$$

Otherwise, let P' be the graph derived by the transformation $P \Longrightarrow_{\rho,m} P'$. Then, P' is an overlap of R and C_{k+1}. If this transformation does not exist, we set $\mathrm{remv}(P, C') := \textsf{false}$. *Let $a^r : C_{k+1} \hookrightarrow C$ be the restricted morphism of a_{k+1}, then*

$$\operatorname{remv}(P, C') := \begin{cases} \mathsf{false} & \textit{if } \operatorname{sub}_{k+2}(c) = \mathsf{false} \\ \bigvee_{Q \in \operatorname{eol}(P', a^r)} \operatorname{Left}(\exists(i_R^Q : R \hookrightarrow Q, \mathsf{true}), \rho) & \textit{otherwise.} \end{cases}$$

Note that $\operatorname{ap}_k(C')$, for any $C' \in \operatorname{IG}(C_{k+1}, C_{k+2})$, will only be evaluated with $\mathtt{true}$ if an occurrence p of C_{k+1} with $p \not\models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ and $\operatorname{tr}_t \circ p \models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$ exists. For any smaller improvements, i.e a similar improvement for a subgraph $C'' \in \operatorname{IG}(C_{k+1}, C_{k+2})$ of C', $\operatorname{ap}(k, C')$ would be evaluated with false. For any bigger improvements, i.e the same improvement for a supergraph $C'' \in \operatorname{IG}(C_{k+1}, C_{k+2})$ of C', $\operatorname{ap}_k(C')$ would also be evaluated with false, if $p \models \exists(a^r_{k+1} : C_{k+1} \hookrightarrow C', \mathsf{true})$. In both cases, the application condition would prohibit the transformation, even if it would be consistency increasing. To resolve this problem, multiple application conditions could be combined by

$$\bigvee_{C' \in \operatorname{IG}(C_{k+1}, C_{k+2})} \operatorname{ap}_k(C').$$

This application condition will be evaluated with $\mathtt{true}$ if the cases described above do appear, with the drawback that this leads to a huge condition, even if duplicate conditions are removed. At least all duplicates of ned(), nui() and nds() can be removed, since they are identical for each $\operatorname{ap}(k, C')$ and only need to be constructed once.

In general, these application conditions are a trade-off between conditions-size and restrictiveness. They are very restrictive, since they do not allow any deletions of occurrences of existentially bound and insertions of universally bound graphs. For example, any of these application conditions with the rule `moveFeature` and constraint c_1 will be equivalent to false; nds() will always be evaluated with false since `moveFeature` does remove elements of the existentially bound graph C_2^1. A change of the conditions constructed by nds() such that it is checked whether two nodes of the type `Feature` are connected to a node `Class` will yield application conditions that are satisfiable with `moveFeature`, but for a similar rule moving two nodes of type `Feature`, this newly constructed nds() would still be evaluated with false. Therefore, this only leads to a slight decrease of restrictiveness.

The conditions constructed by ned() and nui() could be changed in a similar fashion. For ned() and the universally bound graph C_j, by checking whether there does exist an additional occurrence p of C_{j+1} such that $p \models \operatorname{sub}_{j+2}(\operatorname{cut}_{\mathrm{k_{max}}}(c))$ and for nui(), by checking whether an introduced occurrence p of C_j does satisfy $\operatorname{sub}_{j+1}(\operatorname{cut}_{\mathrm{k_{max}}}(c))$. The construction of these is similar to the construction of consistency guaranteeing application conditions as introduced by Habel and Pennemann [2], which is known to construct huge application conditions. Also, they do get more and more restrictive for increasing k, since the number of conditions constructed by ned() and nui() also increases.

For constraints of the form $c := \forall C_0 \exists C_1$ it can be shown that $\operatorname{ap}_0(C_2)$ is not only a direct consistency increasing, but also a direct consistency improving application condition.

Example 5.2. *Consider the rule* `assignFeature` *of figure 8 and constraint c_1 of figure 2. The application condition of c_1 at layer 1 with C_2^1 for* `assignFeature` *constructed by definition 5.4 is shown in figure 9. The first two rows are conditions constructed by* $\mathrm{exv}(\cdot,\cdot)$ *and the third row is the condition constructed by* $\mathrm{remv}(\cdot,\cdot)$. *Note that* $\mathrm{ned}()$, $\mathrm{nui}()$ *and* $nwo()$ *did not construct any conditions since* `assignFeature` *does not create elements of C_1^1 and does not delete elements of C_2^1.*

Additionally, this application condition is also a consistency improving application condition w.r.t c_2.

Let us now show that the construction above generates consistency increasing application conditions.

Theorem 5.2. *Let a constraint c in UANF be given. Each rule $\rho = (\mathrm{ac}', \rho')$ with $\mathrm{ac}' = \mathrm{ac} \wedge \mathrm{incr}_k(C, \rho)$ and $0 \leq k < \mathrm{nl}(c)$ being even is a consistency increasing rule at layer $k-1$ w.r.t. c.*

Proof. Let a transformation $t : G \Longrightarrow_\rho H$ with $\mathrm{k_{max}}(c, G) = k-1$ be given. Since $\mathrm{main}_k(\rho)$ is contained in $\mathrm{incr}_k(C, \rho)$, t is a consistency maintaining transformation at layer k with Theorem 5.1. It remains to show that t satisfies (4.5).

Let $\mathrm{sub}_k(c) = \forall(a_k : C_k \hookrightarrow C_{k+1}, e)$ be the sub-condition of c at layer k. Let

$$\mathbf{G} := \begin{cases} \mathrm{IG}(C_{k+1}, C_{k+2}) & \text{if } e \neq \mathsf{false} \\ \{C_k\} & \text{otherwise.} \end{cases}$$

Assume that (4.5) does not hold. Then, no morphism $p : C_{k+1} \hookrightarrow G$ with $p \not\models \mathrm{IC}_0(e, C')$, such that $\mathrm{tr}_t \circ p$ is not total or $\mathrm{tr}_t \circ p \models \mathrm{IC}_0(e, C')$ exists, for any $C' \in \mathbf{G}$. Then, no overlap P of L and C_{k+1} with $i_L^P(L \setminus K) \cap i_{C_{k+1}}^P(C_{k+1}) \neq \emptyset$ exists, such that $m \models \mathrm{exv}(P, C)$.

Let P be an overlap of C_{k+1} and L with $i_L^P(L \setminus K) \cap i_{C_{k+1}}^P(C_{k+1}) = \emptyset$. If $e = \mathsf{false}$, then $\mathrm{remv}(P, C) = \mathsf{false}$. Otherwise, if $m \models \mathrm{exv}(P, C) \wedge \mathrm{remv}(P, C)$, it holds that $i_{C_{k+1}}^P \not\models \mathrm{IC}_0(e, C)$ and a graph $Q \in \mathrm{eol}(P', a^r)$ exists, such that $m \models \mathrm{Left}(\exists(i_R^Q : R \hookrightarrow Q, \mathsf{true}), \rho)$. In this case, $\mathrm{tr}_t \circ i_{C_{k+1}}^P \models \mathrm{IC}_0(e, C)$ follows and (4.5) is satisfied.

In total, $m \not\models \mathrm{exv}(P, C) \wedge \mathrm{remv}(P, C)$ follows for all $P \in \mathrm{ol}(L, C_{k+1})$ and therefore $m \not\models \mathrm{ap}_k(C)$.

In total follows, if $\models \mathrm{ap}_k(C)$, then t is a direct consistency increasing transformation. □

5.2 Basic Increasing and Maintaining Rules

5.3 Application Conditions for Basic Rules

5.4 Conflicts within and between conditions

Definition 5.5 (Conflicts within condition). *Let a condition c in UANF be given. Then, an existentially bound graph C_k has a* conflict *with an universally bound graph C_j*

if a transformation $t : G \Longrightarrow_\rho H$ with $\rho = C_{k-1} \overset{\mathrm{id}}{\hookleftarrow} C_{k-1} \xhookrightarrow{a_{k-1}} C_k$ exists such that the following holds

$$\exists p : C_j \hookrightarrow H(\neg\exists p' : C_j \hookrightarrow G(\mathrm{tr}_t \circ p' = p)).$$

An universally bound graph C_k is has a conflict *with an existentially bound graph C_j if a transformation $t : G \Longrightarrow_\rho H$ with $\rho = C_k \xhookleftarrow{i_{C'}} C' \overset{\mathrm{id}}{\hookrightarrow} C'$ and $C' \in \mathrm{IG}(C_{k-1}, C_k)$ exists such that the following holds.*

$$\exists p : C_j \hookrightarrow G(\mathrm{tr}_t \circ p \textit{ is not total}).$$

A condition c in UANF is called conflict free *if no graph C_k in c has a conflict with a graph C_j with $0 \leq j < k \leq \mathrm{nl}(c)$. A graph C_k has a* transitive conflict *with $C_{j'}$ a graph C_j exists such that C_k has a conflict with C_j and C_j has a (transitive) conflict with $C_{j'}$. A condition c does contain a* circular conflict *if a graph C_k exists such that C_k has a transitive conflict with itself. Otherwise, c is called* circular conflict free.

Lemma 5.6. *Let a constraint c in UANF be given.*

1. *Let C_k be an existentially bound graph of c. Then, C_k has a conflict with C_j if and only if an overlap P of C_k and C_j exists such that*

$$i^P_{C_k}(C_k \setminus C_{k-1}) \cap i^P_{C_j}(C_j) \neq \emptyset$$

   and the rule $\rho = C_k \overset{l}{\hookleftarrow} C_{k-1} \overset{r}{\hookrightarrow} C_{k-1}$ is applicable at match $i^P_{C_k}$.

2. *Let C_k be an universally bound graph of c. Then C_k has a conflict with C_j if and only if an overlap P of C_k and C_j exists such that*

$$i^P_{C_k}(C_k \setminus C_{k-1}) \cap i^P_{C_j}(C_j \setminus C_{j-1}) \neq \emptyset$$

   and each rule $\rho = C_k \xhookleftarrow{i_{C'}} C' \overset{\mathrm{id}}{\hookrightarrow} C'$ with $C' \in \mathrm{IG}(C_{k-1}, C_k)$ is applicable at match $i^P_{C_k}$.

Proof. Let a condition c in UANF be given.

1. "$\Longrightarrow$": Let C_k be an existentially bound graph that has a conflict with an universally bound graph C_j. Then, there does exists a transformation $t : G \Longrightarrow_\rho H$ with $\rho = C_{k-1} \overset{l}{\hookleftarrow} C_{k-1} \overset{r}{\hookrightarrow} C_k$ such that a new occurrence p of C_j has been inserted. Since only elements of $C_k \setminus C_{k-1}$ have been inserted it holds that $p(C_j) \cap n(C_k \setminus C_{k-1})$ with n being the comatch of t. The graph $p(C_j) \cup n(C_k)$ is the searched for overlap and the rule $\rho^{-1} = C_k \overset{l}{\hookleftarrow} C_k \overset{r}{\hookrightarrow} C_{k-1}$ has to be applicable at n.
   "$\Longleftarrow$": Let C_k be an existentially and C_j an universally bound graph such that an overlap P of C_k and C_j with $i^P_{C_k}(C_k \setminus C_{k-1}) \cap i^P_{C_j}(C_j) \neq \emptyset$ exists such that the rule $\rho = C_k \overset{l}{\hookleftarrow} C_k \overset{r}{\hookrightarrow} C_{k-1}$ is applicable at match $i^P_{C_k}$. Then, the inverse transformation of $t : P \Longrightarrow_{\rho, i^P_{C_k}} H$ is the searched for transformation and C_k has a conflict with C_j.

2. "$\Longrightarrow$": Let C_k be an universally bound graph that has a conflict with an existentially bound graph C_j. Then, a transformation $t : G \Longrightarrow_\rho H$ with $\rho = C_k \overset{l}{\hookleftarrow} C' \overset{r}{\hookrightarrow} C'$ and C' being a subgraph of C_k exists such that a occurrence p of C_j has been deleted. Then, the graph $p(C_j) \cup m(C_k)$ is the searched for overlap and $i^P_{C_k}(C_k \setminus C_{k-1}) \cap i^P_{C_j}(C_j \setminus C_{j-1}) \neq \emptyset$ has to hold since ρ only deletes elements of $C_k \setminus C_{k-1}$.
   "$\Longleftarrow$": Let C_k be universally and C_j existentially bound such that an overlap P of C_k and C_j with $i^P_{C_k}(C_k \setminus C_{k-1}) \cap i^P_{C_j}(C_j \setminus C_{j-1}) \neq \emptyset$ exists such that each rule $\rho = C_k \overset{i_{C'}}{\hookleftarrow} C' \overset{\text{id}}{\hookrightarrow} C'$ with $C' \in \text{IG}(C_{k-1}, C_k)$ is applicable at match $i^P_{C_k}$. Then, the inverse transformation of $t : P \Longrightarrow_{\rho, i^P_{C_k}} H$ is the searched for transformations and C_k has a conflict with C_j.

□

Definition 5.7 (**Conflicts between conditions**). *Let conditions c_1 and c_2 be given. Then, c_1 has a conflict with c_2 if graphs C_k and C_j of c_1 and $c_2$exist such that C_k has a conflict with C_j or vice versa. A set of conditions is called* conflict free *if no conditions c_1 and c_2 exist such that c_1 has a conflict with c_2 or vice versa. A condition c_1 has a* transitive conflict with *c_2 if a condition c' exists such that c_1 has a conflict with c' and c' has a (transitive) conflict with c_2. A set of conditions does contain a* circular conflict *if a condition c has a transitive conflict with itself. Otherwise, the set is called* circular conflict free.

Definition 5.8 (**conflict free graphs**). *Let a graph G and a constraint c in UANF be given. Then, G is called* conflict free *w.r.t c if for each transformation $t : G \Longrightarrow_{\rho,m} H$ with*

$$\rho = C_{\text{k}_\text{max}} \overset{\text{id}}{\hookleftarrow} C_{\text{k}_\text{max}} \overset{i_{C_{\text{k}_\text{max}}}}{\hookrightarrow} C'$$

and $C' \in \text{IG}(C_{\text{k}_\text{max}+1}, C_{\text{k}_\text{max}+2})$ or

$$\rho = C_{\text{k}_\text{max}+1} \overset{1_{C'}}{\hookleftarrow} C' \overset{\text{id}}{\hookrightarrow} C'$$

and $C' \in \text{IG}(C_{\text{k}_\text{max}}, C_{\text{k}_\text{max}+1})$ and (4.3) and (4.4) hold.

Lemma 5.9. *Let a graph G and a constraint c in UANF be given. Let P be the set of all Then, G is conflict free w.r.t c if and only if either c is conflict free or*

$$G \models \bigwedge_{P \in P} \forall(a : \emptyset \hookrightarrow P, \mathit{false})$$

Proof. □

5.5 Basic increasing rules

As described above, the application conditions for general rules are very restrictive. This is because the application condition does not allow a transformation to delete occurrences of existentially bound or insert occurrences of universally bound graphs even if this would not lead to a decrease of satisfaction up to layer. Now, we will consider a special set of rules called *basic increasing rules.* The main idea is that these rules are not able to delete occurrences of existentially or insert occurrences of universally bound graphs and additionally are able to increase consistency. That means, given a basic increasing rule ρ, there does exist a transformation $t : G \Longrightarrow_\rho H$ such that t is a consistency increasing transformation w.r.t to a constraint c in UANF. For *basic increasing rules*, consistency increasing application conditions can be constructed that are less restrictive and smaller in size compared to application conditions for general rules.

First, we define *non-consistency decreasing rules at layer* and *non-consistency decreasing rules up to layer* where a non-consistency decreasing rule at layer k is not able to delete occurrences of C_{k+1}, if $\text{sub}_k(c)$ is existentially bound and not able to create occurrences of C_{k+1} if $\text{sub}_k(c)$ is universally bound. This can be checked via second order formulas by, in the first case, checking whether all occurrences of C_{k+1} still exist in H and, in the second case, checking whether all occurrences of C_{k+1} in H already existed in G. The notion of non-consistency decreasing rules up to layer k is a abbreviation, meaning that a rule ρ is non-consistency decreasing up to layer k if it is non-consistency decreasing at layer j for all $0 \leq j \leq k$.

Definition 5.10 (non-consistency decreasing rule). *Let a plain rule* $\rho = L \overset{l}{\hookleftarrow} K \overset{r}{\hookrightarrow} R$ *and a constraint* c *in UANF be given. Then,* ρ *is called a* non-consistency decreasing rule w.r.t c at layer k *if*

1. *If* $\text{sub}_k(c)$ *is universally bound and for all transformations* $t : G \Longrightarrow_\rho H$ *it holds that*
$$\forall p : C_{k+1} \hookrightarrow H(\exists p' : C_{k+1} \hookrightarrow G(\text{tr}_t \circ p' = p)). \tag{5.2}$$

2. *If* $\text{sub}_k(c)$ *is existentially bound for all transformations* $t : G \Longrightarrow_\rho H$ *it holds that*
$$\forall p : C_{k+1} \hookrightarrow G(\text{tr}_t \circ p \textit{ is total}). \tag{5.3}$$

The rule ρ *is called* non-consistency decreasing w.r.t c up to layer k *if* ρ *is non-consistency decreasing w.r.t* c *at layer* j *for all* $0 \leq j \leq k$ *and it holds that*

$$\bigwedge_{C' \in \text{IG}(C_k, C_{k+1})} \forall p : C' \hookrightarrow G(\text{tr}_t \circ p \textit{ is total}) \tag{5.4}$$

if scondkc is existentially bound.

Given a graph G and a constraint c, we will show that non-decreasing rules w.r.t c up to layer $\text{k}_{\max}$ cannot decrease the satisfaction up to layer.

Lemma 5.11. *Let a graph G, a constraint c in UANF and a non-consistency decreasing rule w.r.t c up to layer $\mathrm{k}_{\max}$ be given. Then, for each transformation $t : G \Longrightarrow_\rho H$ it holds that*

$$\mathrm{k}_{\max}(c, G) \leq \mathrm{k}_{\max}(c, H).$$

Proof. Assume that $\mathrm{k}_{\max}(c, G) > \mathrm{k}_{\max}(c, H)$. Then, either 1. or 2. applies.

1. There does exists an occurrence $p : C_j \hookrightarrow H$ with $0 \leq j \leq \mathrm{k}_{\max}(c, G)$ of a universally bound graph C_j such that there does not exist a occurrence $q : C_j \hookrightarrow G$ of C_j with $p = \mathrm{tr}_t \circ q$. Then ρ is not a non-consistency decreasing rule up to layer $\mathrm{k}_{\max}(c, G)$ since (5.2) is not satisfied.

2. There does exists an occurrence $p : C_j \hookrightarrow G$ with $0 \leq j \leq \mathrm{k}_{\max}(c, G)$ of an existentially bound graph C_j such that $\mathrm{tr} \circ p$ is not total. Then, ρ is not a non-consistency decreasing rule up to layer $\mathrm{k}_{\max}(c, G)$ since (5.3) is not satisfied.

□

Since it is very time consuming to check whether a rule ρ is non-consistency decreasing based on the definition above, we present a characterisation for non-consistency decreasing rules which only relies on ρ itself. First, let us assume that ρ is able to create occurrences of a universally bound graph C_j. This is possible if (a) ρ does insert an edge of $C_j \setminus C_{j-1}$ which connects already existing nodes of C_j, since it is unclear whether this would create a new occurrence of C_j. Or (b) if ρ does insert a node v of C_j such that all edges $e \in E_{C_j}$ with $\mathrm{src}(e) = v$ or $\mathrm{tar}(e) = v$ are also inserted. If at least one of these edges is not inserted, it is guaranteed that this insertion does not create an occurrence of C_j since v is only connected to edges that have also been inserted by ρ.

Second, let us assume that ρ is able to delete occurrences of a existentially bound graph C_j. This is possible if (a) ρ does delete an edge of $C_j \setminus C_{j-1}$ or (b) ρ does delete a node v of $C_j \setminus C_{j-1}$ such that all edges $e \in E_{C_j}$ with $\mathrm{src}(e) = v$ or $\mathrm{tar}(e) = v$ are also deleted. If ρ deletes a node c of $C_j \setminus C_{j-1}$ without all its connected edges in C_j there does not exist a transformation such that ρ deletes an occurrence of C_j by deleting this node since the dangling edge condition is not satisfied.

Then, we are able to determine whether a rule ρ is non-consistency decreasing by checking whether ρ does not satisfy the just mentioned properties.

Lemma 5.12. *Let a plain rule $\rho = L \overset{l}{\hookleftarrow} K \overset{r}{\hookrightarrow} R$ and a constraint c in EANF be given. Then, ρ is a non-consistency decreasing rule at layer k w.r.t c if either 1. or 2. applies.*

1. *If C_k is universally bound, let I be the set of all isolated nodes of C_k, P be the set of all partial morphisms $p : C_k \hookrightarrow R$,*

$$E = \bigcup_{p \in P} p(E_{C_k}) \cap (E_R \setminus E_K) \text{ and } V = \bigcup_{p \in P} p(V_{C_k}) \cap (V_R \setminus V_K).$$

Then, ρ is called a non-consistency decreasing rule at layer k w.r.t c *if* $I \cap V = \emptyset$ *and*

$$\begin{aligned}&\big(\neg\exists e \in E : \exists v, v' \in V_K : \mathrm{src}(e) = r(v) \wedge \mathrm{tar}(e) = r(v')\big) \qquad \wedge\\ &\big(\forall v \in V : \exists e \in E_{C_k} : \exists p \in P : \mathrm{src}(e) = p^{-1}(v) \vee \mathrm{tar}(e) = p^{-1}(v)\\ &\wedge \neg\exists e' \in E : \exists p \in P : p^{-1}(e') = e\big)\end{aligned}$$

holds.

2. *If C_k is existentially bound, let I be the set of all isolated nodes of C_k, P be the set of al morphisms $p : C_k \hookrightarrow L$,*

$$E = \bigcup_{p \in P} p(E_{C_k}) \cap (E_L \setminus E_K) \text{ and } V = \bigcup_{p \in P} p(V_{C_k}) \cap (V_L \setminus V_K).$$

Let I be the set of all isolated nodes of C_k. Then, ρ is called a non-consistency decreasing rule at layer k w.r.t c *if* $I \cap V = \emptyset$ *and*

$$\begin{aligned}&\big(\neg\exists e \in E_{C_k} \setminus E_{C_{k-1}} : \exists p \in P : \exists e' \in E : p(e) = e'\big) \qquad \wedge\\ &\big(\forall v \in V : \exists e \in E_{C_k} : \exists p \in P : \mathrm{src}(e) = p^{-1}(v) \vee \mathrm{tar}(e) = p^{-1}(v)\\ &\wedge \neg\exists e' \in E : \exists p \in P : p^{-1}(e') = e\big)\end{aligned}$$

Proof. □

Now we are ready to define *basic increasing rules at layer k* with the set of basic increasing rules being a subset of the set of non-decreasing rules. With the property that each basic increasing rule ρ is also a non-decreasing rule up to layer k it is satisfied that the satisfaction up to layer is not decreased. Additionally, if $\mathrm{sub}_k(c)$ is existentially bound it has to be checked that the consistency for an occurrence of C_k is not decreased and that ρ removes an violation. This can be done by either deleting an element of $C_{k+1} \setminus C_k$ if $\mathrm{sub}_k(c)$ is universally bound or by inserting elements of C_{k+1} such that for one occurrence p of C_k the consistency in increased if $\mathrm{sub}_k(c)$ is existentially bound. Additionally, the left-hand side of each basic increasing rule has to contain C_{k+1} if $\mathrm{sub}_k(c)$ is universally bound such that either C_{k+1} gets deleted and the left-hand side has to contain C_k if $\mathrm{sub}_k(c)$ is existentially bound such that the consistency of C_k is increased. This property yields the advantage that application conditions for basic increasing rules are less complex, smaller in size and during a repair process matches of these rules are easier to handle.

This, on first sight seems like a restriction of the set of basic increasing rules but the context of each rule ρ that does satisfy all properties of a basic increasing rule excluding that C_k or C_{k+1} is a subgraph of the left-hand side can be expanded such that ρ is a basic increasing rule and the semantic of ρ is not increased. Later on, a method to derive this rules will be presented.

Basic increasing rules at layer k are called *deleting basic increasing rules* if $\mathrm{sub}_k(c)$ is universally bound and *inserting basic increasing rules* if $\mathrm{sub}_k(c)$ is existentially bound.

For deleting basic increasing rules we introduce the restriction that these rules are only allowed to delete edges but no nodes of C_{k+1} since otherwise it is not possible, given a rule set and a constraint to decide whether this rules set is able to repair an arbitrary graph based only on deleting basic increasing rules. For example, consider a rule that deletes a node of C_{k+1}. Then, it is unknown whether this node is connected by edges not belonging to C_{k+1} and it is unclear whether all occurrence of C_{k+1} could be destroyed by ρ since the dangling edge condition could be unsatisfied.

Definition 5.13 (**basic increasing rule**). *Let a constraint c in UANF and a plain rule $\rho = L \overset{l}{\hookleftarrow} K \overset{r}{\hookrightarrow} R$ be given. Then, ρ is called* basic increasing w.r.t c at layer k *if*

1. *If* $\mathrm{sub}_k(c)$ *is universally bound: ρ is a non-consistency decreasing rule up to layer $k+1$ and there does exist a morphism $i : C_{k+1} \hookrightarrow L$ such that*

$$\exists e \in E_{C_{k+1}} \setminus E_{C_k} \big(\neg \exists e' \in E_K (i(e) = l(e'))\big) \wedge \\ \forall v \in V_{C_{k+1}} \setminus V_{C_k} \big(\exists v' \in V_K (i(c) = l(v')\big)$$

   holds. Then, ρ is called a deleting basic increasing rule.

2. *If* $\mathrm{sub}_k(c)$ *is existentially bound: ρ is a non-consistency decreasing rule up to layer k and there does exist a morphism $i : C_k \hookrightarrow L$ such that*

$$i \not\models \exists(a_k^r : C_k \hookrightarrow C', \mathsf{true}) \wedge r \circ l^{-1} \circ i \models \exists(a_k^r : C_k \hookrightarrow C', \mathsf{true})$$

   for any $C' \in \mathrm{IG}(C_k, C_{k+1})$. Then, ρ is called a inserting basic increasing rule with *C'.*

Again, 5.13 relies on every transformation of a rule ρ. Therefore, we present an alternative method to determine whether ρ satisfies 5.13 or not by checking that ρ does not delete any edges or nodes of $C_{k+1} \setminus C_k$.

Lemma 5.14. *let the sets E and V be constructed as introduced in lemma 5.12 it holds that*

$$\forall v \in V : \exists v' \in V_K : l(v') = v \wedge \\ \forall e \in E : \exists e' \in E_K : l(e') = e$$

Then, ρ is called an inserting basic increasing rule *and i is called the increasing morphism of ρ.*

Proof. □

Lemma 5.15. *Let a constraint c in UANF, a basic increasing rule ρ at layer* $\mathrm{k_{max}} + 1$ *and a transformation $t : G \Longrightarrow_\rho H$ be given. Then,*

$$t \textit{ is consistency increasing w.r.t } c \iff t \textit{ is direct consistency increasing w.r.t } c$$

Proof. The "$\Longleftarrow$" side of the equivalence holds with lemma 4.4. We need to show that the "$\Longrightarrow$" side of the equivalence holds. Let $t : G \Longrightarrow_\rho H$ be a consistency increasing transformation and ρ a basic increasing rule. Then, t satisfies (5.13) and the satisfaction of (4.1) follows immediately. Since ρ is also a non-consistency decreasing rule up to layer $\mathrm{k_{max}}$, t satisfies (5.2) and (5.3), then the satisfaction (4.2), (4.3) and (4.4) also follows immediately.

1. If ρ is a deleting basic increasing rule. Since t is a consistency increasing rule there has to exist a morphism $p : C_{\mathrm{k_{max}}+2} \hookrightarrow G$ such that $\mathrm{tr}_t \circ p$ is not total and with that (4.5) is satisfied.

2. If ρ is a inserting basic increasing rule there exists an occurrence $p : C_{\mathrm{k_{max}}+2} \hookrightarrow G$ and a graph $C \in \mathrm{IG}(C_{\mathrm{k_{max}}+2}, C_{\mathrm{k_{max}}+3})$ such that $p \not\models \mathrm{IC}_{\mathrm{k_{max}}+2}(c, C)$ and $\mathrm{tr}_t \circ p \not\models \mathrm{IC}_{\mathrm{k_{max}}+2}(c, C)$. Therefore (4.5) is satisfied.

In total follows that t is a direct consistency increasing transformation w.r.t c. □

For basic increasing rules at layer $j < \mathrm{k_{max}}+1$ this equality does not hold since these rules are allowed to destroy occurrences of universally bound graphs $C_{j'}$ or to create occurrences of existentially bound graphs $C_{j'}$ with $j < j' < \mathrm{k_{max}}+1$. With this results follow that if $t : G \Longrightarrow_\rho H$ is a c-guaranteeing transformation and ρ is a basic increasing rule at layer $\mathrm{k_{max}}+1$, t is also a direct increasing rule w.r.t c.

Definition 5.16 (application conditions for basic increasing rules). *Let a constraint c in UANF and a basic increasing rule $\rho = L \overset{l}{\hookleftarrow} K \overset{r}{\hookrightarrow} R$ w.r.t c at layer k be given. We define the application condition for ρ as:*

1. *If ρ is a deleting increasing rule:*

$$\mathrm{basic}_j(\rho) := \begin{cases} \bigvee_{P \in \mathrm{ol}(L, C_k)} \bigwedge_{P' \in \mathrm{eol}(P, a_k)} \exists(i_L^P : L \hookrightarrow P, \neg\exists(i_P^{P'} : P \hookrightarrow P', \mathit{true})) & \text{if } j = k \\ \mathit{false} & \text{if } j \neq k. \end{cases}$$

2. *If ρ is an inserting increasing rule. Let $a^p : C_k \hookrightarrow C$ be a partial morphism of a_k , $e : C_k \hookrightarrow L$ be the increasing morphism of c and Q be set of all overlaps P of L and C such that $i_L^P \circ e \models \exists(a^p : C_k \hookrightarrow C, \mathit{true})$:*

$$\mathrm{basic}_j(\rho) := \begin{cases} \bigvee_{P \in Q} \exists(i_L^P : L \hookrightarrow P, \mathit{true}) & , \text{if } j = k \\ \mathit{false} & , \text{if } j \neq k \end{cases}$$

Theorem 5.3. *Let a graph G, a constraint c and a basic increasing rule ρ at layer $\mathrm{k_{max}} < k \leq \mathrm{k_{max}}+2$ be given. Then, each transformation*

$$t : G \Longrightarrow_{\rho', m} H$$

with the rule $\rho' = (\rho, \mathrm{basic}_k(\rho))$ is a direct consistency increasing transformation.

Proof. Since ρ is a basic increasing rule it is also a non-consistency decreasing rule up to layer $\mathrm{k_{max}}+2$. Therefore, (4.1) is trivially satisfied since ρ satisfies (5.4), also (4.2) is satisfied since ρ satisfies (5.2) also (4.3) and (4.4) are satisfied because ρ satisfies (5.2) and (5.3) for all $0 \leq j \leq k+2$. It remains to show that (4.5) is satisfied.

1. Let ρ be a deleting basic increasing rule. Then, $k = \mathrm{k_{max}}+1$ and $\mathrm{basic}_k(\rho) = \bigvee_{P \in \mathrm{ol}(L, C_k)} \bigwedge_{P' \in \mathrm{eol}(P, a_k)} \exists(i_L^P : L \hookrightarrow P, \neg\exists(i_P^{P'} : P \hookrightarrow P', \mathsf{true}))$. If $m \models \mathrm{basic}_j(\rho)$ there does exist a occurrence $p : C_k \hookrightarrow G$ with $p \not\models \exists(a_{k+1} : C_k \hookrightarrow C_{k+1}, \mathsf{true})$ and $p(C_k \setminus C_{k-1}) \cap m(L \setminus K) \neq \emptyset$. If follows that $\mathrm{tr}_t \circ p$ is not total and (4.5) is satisfied.

2. Let ρ be a inserting basic increasing rule with $C \in \mathrm{IG}(C_k, C_{k+1})$. Then, $k = \mathrm{k_{max}}+1$ and $\mathrm{basic}_j(\rho) = \bigvee_{P \in Q} \exists(i_L^P : L \hookrightarrow P, \mathsf{true})$. If $m \models \mathrm{basic}_j(\rho)$, $m \circ e \not\models \exists(a_k^r : C_k \hookrightarrow C, \mathsf{true})$ with e being the increasing morphism of ρ. Therefore $\mathrm{tr}_t \circ m \circ i \models \exists(a_k^r : C_k \hookrightarrow C, \mathsf{true})$ and (4.5) is satisfied.

In total follows that t is a direct consistency increasing transformation. □

Theorem 5.4. *Let a constraint c in EANF and a set of rules $\mathcal{R}$ be given. Then, $\mathcal{R}$ is a repairing set of c at layer $k \leq \mathrm{nl}(c)$ if either 1 or 2 applies.*

1. *For any universally bound graph C_j at layer $j \leq k$ of c, $(\rho, \mathrm{basic}(j, C_{j+1})) \in \mathcal{R}$ and ρ is a deleting potentially minimal improving rule at layer j with C_{j+1}, such that ρ only deletes edges of C_j.*

2. *A decomposition $\mathbf{P}$ of C_k with C_{k-1} exists, such that for each $P \in \mathbf{P}$ a rule $(\rho, \mathrm{basic}(k, P)) \in \mathcal{R}$ exists, such that ρ is an inserting basic improving rule at layer k with P.*

3. *There does exist a sequence of graphs*

$$C_k = C'_0 \hookrightarrow C'_1 \hookrightarrow \ldots \hookrightarrow C'_n = C_{k+1}$$

such that a inserting basic increasing rule $\rho_i = L_i \overset{l_i}{\hookleftarrow} K_i \overset{r_i}{\hookrightarrow} R_i$ at layer k with C_i exists such that $L_i \hookrightarrow R_{i-1}$ and $e_i = i_{C_{i-1}}^{C_i} \circ e_{i-1}$ with e_i being the increasing morphism of ρ_i for all $i \in \{1, \ldots n\}$.

Proof. Let a constraint c in EANF, a rule set $\mathcal{R}$ and a graph G with $k = \mathrm{c_{max}}$ and $\mathrm{c_{max}} < \mathrm{nl}(c)$ be given. We show that a sequence $G = C'_0 \Rightarrow \ldots \Rightarrow C'_n = H$ with rules of $\mathcal{R}$ exists, such that $H \models_{k+2} c$ if 1. or 2. of theorem 5.4 is satisfied.

1. Assume that 1. of theorem 5.4 holds. Let $(\rho, \mathrm{basic}(j, C_{j+1})) \in \mathcal{R}$, such that $\rho = L \overset{l}{\hookleftarrow} K \overset{r}{\hookrightarrow} R$ is a deleting potentially minimal improving rule at layer $j \leq k$ with C_{j+1} and C_j is a universally bound graph of c. Then, $\mathrm{basic}(j, C_{j+1}) = \exists\big(b : L \hookrightarrow C_j, \neg\exists(a_j : C_j \hookrightarrow C_{j+1}, \mathsf{true})\big)$. Let $q : C_j \hookrightarrow G$ be a morphism such that $q \not\models \exists(a_j : C_j \hookrightarrow C_{j+1}, \mathsf{true})$. Since $L \subseteq C_j$, we can construct a morphism $m_1 : L \hookrightarrow G$ with $m_1 = q \circ b$ and therefore $m_1 \models \mathrm{basic}(j, C_{j+1})$. Since r only

deletes edges, a transformation $t : G = G_0 \Rightarrow_{r,m_1} G_1$ exists and $\mathrm{tr}_t \circ p$ is not total. Because r does not insert any elements of C_j:

$$|\{q : C_k \hookrightarrow G_0 \mid q \not\models d\}| < |\{q : C_k \hookrightarrow G_1 \mid q \not\models d\}|$$

with $d = \exists(a_j : C_j \hookrightarrow C_{j+1}, \mathsf{true})$. By iteratively applying this construction, we can generate a finite sequence of transformations

$$G = G_0 \Rightarrow_{r,m_1} G_1 \Rightarrow_{r,m_2} \ldots \Rightarrow_{r,m_n} G_n = H$$

such that $|\{q : C_k \hookrightarrow G_n \mid q \not\models d\}| = 0$ and therefore $H \models_j c$. With lemma 4.9, $H \models_{k+2} c$ and $H \models c$ follows.

2. Assume that 2. of theorem 5.4 holds. Let $\rho_0 = (\rho, \mathrm{basic}(k, P)) \in \mathcal{R}$, such that ρ is an inserting basic improving rule of c at layer k with $P \in \mathbf{P}$. Then, $\mathrm{basic}(k, P) = \neg\exists(b : L \hookrightarrow P, \mathsf{true})$. Let $q_0 : C_k \hookrightarrow G$ be a morphism, such that $q_0 \not\models \exists(a'_k : C_k \hookrightarrow P, \mathsf{true})$ with a'_k being a partial morphism of a_k. Since $L = C_k$, we set $m_0 : C_k \hookrightarrow G$ with $m_0 = q_0$. It follows that $m_0 \models \neg\exists(a'_k : C_k \hookrightarrow P, \mathsf{true}) = \mathrm{basic}(k, P)$. Because r does not delete any elements, a transformation $t_0 : G \Longrightarrow_{r_0,m_0} G_1$ exists and $\mathrm{tr}_t \circ q \models \exists(a'_k : C_k \hookrightarrow P, \mathsf{true})$. We set $q_1 = \mathrm{tr}_{t_0} \circ q_0$ and apply the same method to q_1.

   By iteratively applying this, we can construct a finite sequence of transformations

$$G \Longrightarrow_{r_0,m_0} G_0 \Longrightarrow_{r_1,m_1} \ldots \Longrightarrow_{r_n,m_n} G_n$$

   such that $m_i = \mathrm{tr}_{t_{i-1}} \circ \ldots \circ \mathrm{tr}_{t_0} \circ m_0$ and $q \models \exists(b_i : C_k \hookrightarrow P_i, \mathsf{true})$ for all $P_i \in \mathbf{P}$ with $q = \mathrm{tr}_{t_n} \circ q_n$. Let $p_i : P_i \hookrightarrow G_n$ be the morphism, such that $q = p_i \circ b_i$.

   Now, we can construct a morphism $p : C_{k+1} \hookrightarrow G$ with

$$p(e) := \begin{cases} p_1(e) & \text{,if } e \in P_1 \\ & \vdots \\ p_j(e) & \text{,if } e \in P_j. \end{cases}$$

   Let $e \in C_k$, because $q(e) = p_i \circ b_i(e)$ and $q(e) = p_\ell \circ b_\ell(e)$ and b_i and b_ℓ are both partial morphisms of a_k, it follows that $b_i(e) = b_\ell(e)$ and therfore $p_i(e) = p_\ell(e)$. Because $(P_i \cap P_\ell) \setminus C_k = \emptyset$ for all $i \neq \ell$, p is a morphism and by the definition of p it follows that $q = p \circ a_k$ and therefore $q \models \exists(a_k : C_k \hookrightarrow C_{k+1}, \mathsf{true})$.

   By iteratively applying this whole construction to all occurrences of C_k that do not satisfy $\exists(a_k : C_k \hookrightarrow C_{k+1}, \mathsf{true})$ the derived graph graph H does not contain any occurrences of C_k not satisfying $\exists(a_k : C_k \hookrightarrow C_{k+1}, \mathsf{true})$ and therefore $H \models_{k+2} c$.

□

Corollary 5.17. *If a set of rule $\mathcal{R}$ is a repairing set of c at layer $k \leq \mathrm{nl}(c)$ and 1. of theorem 5.4 applies, then $\mathcal{R}$ is a repairing set of c at layer j for all $k \leq j \leq \mathrm{nl}(c)$.*

Lemma 5.18. *Let a graph G_0, a constraint c in EANF and a repairing set $\mathcal{R}$ at layer $c^{G_0}_{\max}+1$ be given, such that each rule in $\mathcal{R}$ is a basic increasing. Then, for every sequence*

$$G_0 \Longrightarrow_{\rho_0,m_0} G_1 \Longrightarrow_{\rho,m_0} \dots \Longrightarrow_{\rho_n,m_n} G_n$$

such that $\rho_i = (\rho, \mathrm{basic}(c^{G_0}_{\max}+1, P)$ with $\rho \in \mathcal{R}$ being a consistency increasing rule at layer $c^{G_0}_{\max}+1$ with P, it holds that

5.6 Repairing rule Sets

Definition 5.19 (repairing rule set). *Let a constraint c in UANF and a set of rules $\mathcal{R}$ be given. Then, $\mathcal{R}$ is called a* repairing rule set for c at layer k *if for all graphs G with $\mathrm{k}_{\max} = k$ a sequence*

$$G = G_0 \Longrightarrow_{\rho_0} \dots \Longrightarrow_{\rho_{n-1}} G_n = H$$

exists, such that $\mathrm{k}_{\max}(c, H) \geq \mathrm{k}_{\max}(c, G) + 2$ and $\rho_j \in \mathcal{R}$ for all $0 \leq j \leq n-1$.

Corollary 5.20. *Let a constraint c in UANF and a rule set $\mathcal{R}$ be given. If either (a) $\mathcal{R}$ is a repairing rule set at layer k for all odd $0 \leq k \leq \mathrm{nl}(c)$ or (b) $\mathcal{R}$ is a repairing rule set at layer k with k being even and $\mathcal{R}$ is also a repairing rule set for all odd $0 \leq j < k$ then for all graphs G a sequence*

$$G = G_0 \Longrightarrow_{\rho_0} \dots \Longrightarrow_{\rho_{n-1}} G_n$$

with $G_n \models c$ and $\rho_j \in \mathcal{R}$ for all $0 \leq j \leq n-1$ exists.

5.6.1 Construction of repairing sets

Definition 5.21 (decomposition of a graph). *Let graphs $G_0 \hookrightarrow G_1$ be given.*

A decomposition of G_1 with G_0 is a minimal set

$$\mathbf{D} \subseteq \{D_v \mid v \in V_{C_1 \setminus C_0}\}$$

of subgraphs of G_1, such that every element of G_1 is contained in at least one $D \in \mathbf{D}$ and every D_v is constructed in the following way: $G_0 \hookrightarrow D_v$, $v \in D_v$ and for all nodes $v' \in D \setminus C_0$ D contains all edges $e \in E_{G_1}$ and all nodes $u \in V_{G_1}$ such that either $\mathrm{tar}(e) = v' \wedge \mathrm{src}(e) = u$ or $\mathrm{src}(e) = u \wedge \mathrm{tar}(e) = v'$.

Lemma 5.22. *Let graphs $G_0 \hookrightarrow G_1$ and a decomposition $\mathbf{D}$ of G_1 with G_0 be given. Then, for each pair $D, D' \in \mathbf{D}$ with $D \neq D'$ the following holds:*

$$(D \setminus C_k) \cap (D' \setminus C_k) = \emptyset$$

Proof. Assume that $(D \setminus C_k) \cap (D' \setminus C_k) \neq \emptyset$, therefore a node $v \in G_1 \setminus G_0$ with $v \in D \cap D'$ exists. By the construction of D and D' it follows that $D = D_v$ and $D' = D_v$ and therefore $D = D'$. This is a contradiction. □

Lemma 5.23. *Let graphs $G_0 \hookrightarrow G_1$ and a decomposition $\mathbf{D}$ of G_1 with G_0 be given. Then,*

$$G_1 = \bigcup_{P \in \mathbf{P}} P.$$

Proof. Let $H := \bigcup_{P \in \mathbf{P}} P$. Firstly, we show that $H \subseteq G_1$. Since every $P \in \mathbf{P}$ is a subgraph of G_1 it follows that $V_H \subseteq V_{G_1}$ and $E_H \subseteq E_{G_1}$

Secondly, we show that $G_1 \subseteq H$. Let $u \in V_{G_1}$ be a node, if $u \in V_{G_0}$, then u is contained in every $P \in \mathbf{P}$ and therefore $u \in V_H$. Otherwise, if $u \notin V_{G_0}$, then u has to be, by the definition of $\mathbf{P}$, contained in at least one $P \in \mathbf{P}$ and $V_{G_1} \subseteq V_H$ follows. Let $e \in E_{G_1}$ be an edge. If $e \in E_{G_0}$, then e is contained in every $P \in \mathbf{P}$ and $e \in E_H$. Otherwise, if $e \notin E_{G_0}$, by the definition of $\mathbf{P}$, e has to be contained in at least one $P \in \mathbf{P}$. It follows that $e \in E_H$ and with that $E_{G_1} \subseteq E_H$. □

Now we are ready to provide the construction for

Construction 5.24. *Let a constraint c in UANF be given. For each existentially bound subcondition $\mathrm{sub}_k(c)$ we construct a graph decomposition $\mathbf{D_k}$ as described in definition 5.21. Then, for each element $D \in \mathbf{D_k}$ a rule*

$$\rho = C_k \xhookleftarrow{\mathrm{id}} C_k \xhookrightarrow{i^D_{C_k}} D$$

is derived.

Additionally, for each universally bound subcondition $\mathrm{sub}_k(c)$ and for each edge $e \in E_{C_{k+1} \setminus C_k}$ a rule

$$\rho = C_{k+1} \xhookleftarrow{i^{C_{k+1}}_D} D \xhookrightarrow{\mathrm{id}} D$$

with $D = C_{k+1} \setminus \{e\}$ is derived.

Example 5.3.

6 Rule-based Graph Repair

6.1 Rule based Graph repair for conflict free constraints

Theorem 6.1. *Let a rule set $\mathcal{R}$ and a conflict free constraint c in UANF be given. Then, $\mathcal{R}$ is a repairing set of c at layer $k \leq \mathrm{nl}(c)$ if a sequence*

$$C_k \Longrightarrow_{\rho_0, m_0} \dots \Longrightarrow_{\rho_n, m_n} H$$

of consistency increasing or non-consistency decreasing transformations $t_0, \dots, t_n$ with $\rho_0, \dots, \rho_n \in \mathcal{R}$ exists such that

$$\forall v \in V_{C_k} (\exists v' \in V_H (\mathrm{tr}_{t_n} \circ \dots \circ \mathrm{tr}_{t_0} \circ \mathrm{id}_{C_k}(v) = v'))$$

and

Algorithm 1: Repair for conflict free constraints and repairing set

```
Data: A graph $G$, a conflict free constraint in UANF and a repairing set $\mathcal{R}$ for
      each layer $\mathrm{k}_{\max} < k\,\mathrm{nl}(c)$
Result: A graph $H$ with $H \models c$
1 for $i = \mathrm{k}_{\max} + 1$ to $\mathrm{nl}(c)$ do
2     while $G \not\models_i c$ do
3         Choose one occurrence $p : C_i \hookrightarrow G$ uniformly at random ;
4         apply a repairing sequence $G \Longrightarrow \ldots \Longrightarrow H$;
5     end
6 end
```

1. *If* $\mathrm{sub}_k(c)$ *is universally bound:*

$$\mathrm{tr}_{t_n} \circ \ldots \circ \mathrm{tr}_{t_0} \textit{ is not total}$$

2. *If* $\mathrm{sub}_k(c)$ *is existentially bound:*

$$\mathrm{tr}\, t_n \circ \ldots \mathrm{tr}\, t_0 \circ \mathrm{id}_{C_k} \models \mathrm{sub}_k(c)\mathrm{cut}_{k+1}(c)$$

Proof. □

Lemma 6.1. *The set rule set constructed as described in construction 5.24 is a repairing set.*

Proof. □

Theorem 6.2. *Algorithm 1 is correct and does terminate.*

Proof. □

6.2 Rule based Graph repair for non-conflict free constraints

Theorem 6.3. *Algorithm 2 terminates and returns an consistent graph.*

Proof. □

6.3 Rule based repair for non-repairing sets

7 Conclusion

Algorithm 2: Repair for non conflict free constraints and repairing set

```
Data: A graph $G$, a non-conflict free constraint in UANF and a repairing set $\mathcal{R}$
      for each layer $\mathrm{k}_{\max} < k\,\mathrm{nl}(c)$
Result: A graph $H$ with $H \models c$
1  for $i = \mathrm{k}_{\max} + 1$ to $\mathrm{nl}(c)$ do
2      while $G \not\models_i c$ do
3          Choose one occurrence $p : C_i \hookrightarrow G$ uniformly at random ;
4          apply a repairing sequence $G \Longrightarrow \ldots \Longrightarrow H$;
5          while $G \not\models_{\mathrm{k}_{\max}}$ do
6              Choose an occurrence $p : C_j \hookrightarrow G$ uniformly at random;
7              apply the repairing sequence.
8          end
9      end
10 end
```

Algorithm 3: Repair for conflict free constraints and repairing set

```
Data: A graph $G$, a conflict free constraint in UANF and a repairing set $\mathcal{R}$ for
      each layer $\mathrm{k}_{\max} < k\,\mathrm{nl}(c)$
Result: A graph $H$ with $H \models c$
1  $\mathcal{G} \leftarrow \{G\}$ for $i = 0$ to $m$ do
2      foreach $G \in \mathcal{G}$ do
3          foreach $t : G \Longrightarrow H$ do
4              if $H \models c$ then
5                  return $H$
6              end
7              if t is non-consistency decreasing or consistency increasing then
8                  $\mathcal{G} \leftarrow \mathcal{G} \cup \{H\}$;
9              end
10         end
11     end
12 end
```

References

[1] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. Fundamentals of algebraic graph transformation. *Monographs in theoretical computer science. An EATCS series. Springer*, 2006.

[2] A. Habel and K.-H. Pennemann. Correctness of high-level transformation systems relative to nested conditions. *Mathematical Structures in Computer Science*, 19(2):245–296, 2009.

[3] J. Kosiol, D. Strüber, G. Taentzer, and S. Zschaler. Sustaining and improving graduated graph consistency: A static analysis of graph transformations. *Science of Computer Programming*, 214:102729, 2022.

[4] K.-H. Pennemann. *Development of Correct Graph Transformation Systems*. PhD thesis, Department für Informatik, Universität Oldenburg, Oldenburg, 2009.

[5] D. Plump. Confluence of graph transformation revisited. In *Processes, Terms and Cycles: Steps on the Road to Infinity*, pages 280–308. Springer, 2005.

[6] C. Sandmann and A. Habel. Rule-based graph repair. *arXiv preprint arXiv:1912.09610*, 2019.