Construction of Concurrent Rules with NACs

Jonas S. Bezerra
jsbezerra@inf.ufrgs.br

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, Brasil
http://www.inf.ufrgs.br

September 19, 2016

Outline of this talk

Negative Application Condition

Shifting NACs

Concurrent rules for a rule sequence

Outline of this talk

Negative Application Condition

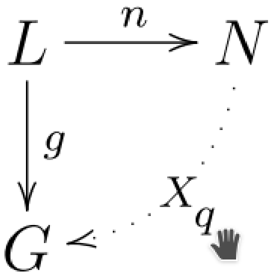
Shifting NACs

Concurrent rules for a rule sequence

Negative Application Conditions I

Definition (NAC): A *negative application condition* or $\mathrm{NAC}(n)$ on L is an arbitrary graph morphism $n : L \to N$.

Definition (NAC satisfiability): A graph morphism $g : L \to G$ satisfies $\mathrm{NAC}(n)$ on L, written $g \models \mathrm{NAC}(n)$, iff $\nexists q : N \to G$ such that q is injective and $q \circ n = g$

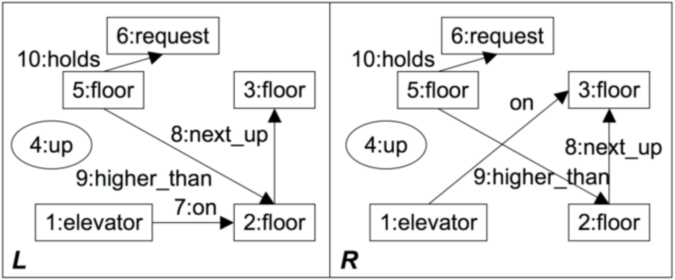


Negative Application Conditions II

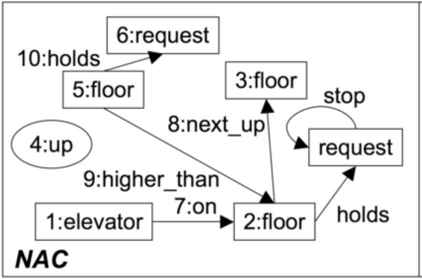
- A set of NACs on L is denoted by $NAC_L = \{NAC(n_i) | i \in I\}$. A graph morphism $g : L \to G$ satisfies NAC_L if and only if g satisfies all single NACs on L i.e. $g \models NAC(n_i) \forall i \in I$.

Negative Application Conditions III

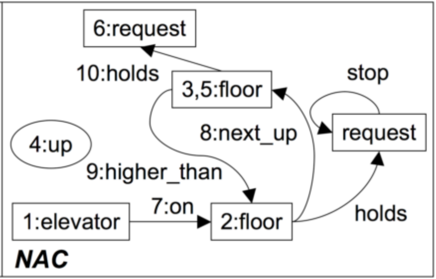
move_up



no_stop_request



no_stop_request_glue



Different NAC satisfaction and AGG approach I

- The *definition* demands the non-existing morphism to be injective.
- Another interpration of NACs the satisfaction of NACs demands the morphism $q : N \to G$ to be non-injective only on $N \backslash n(L)$. (See Remark 2.3.4)
- With this interpretation q may glue the same parts as the match is gluing.
- Without this, for each kind o potential gluing of the LHS, a corresponding NAC needs to be added.

Outline of this talk

Negative Application Condition

Shifting NACs

Concurrent rules for a rule sequence

Over a Rule I

Definition (construction of Left from right NACs): For each $NAC(n_i)$ on R with $n_i : R \to N_i$ of a rule $p = (L \leftarrow K \to R)$, the equivalent left application condition $L_p(NAC(n_i))$ is defined in the following way:

$$\begin{tikzcd}
L \arrow[d, "n_i'"'] \arrow[dr, phantom, "(2)"] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & R \arrow[d, "n_i"] \\
N_i' & Z \arrow[l] \arrow[r] & N_i
\end{tikzcd}$$

- If the pair $(K \to R, R \to N_i)$ has a pushout complement complement, we construct $(K \to Z, Z \to N_i)$ as the pushout complement (1). Then we construct pushout (2) with the morphism $n_i' : L \to N_i'$. Now we define $L_p(NAC(n_i)) = NAC(n_i')$.
- If the pair $(K \to R, R \to N_i)$ does not have a pushout complement, we define $L_p(NAC(n_i)) = \mathrm{true}$

Over a Rule II

Theorem (inverse direct transformation with NACs): For each direct transformation with NACs $G \Rightarrow H$ via a rule $p = (L \leftarrow K \rightarrow R)$with NAC_p a set of left NACs on p, there exists an inverse direct transformation with NACs $H \Rightarrow G$ via the inverse rule p^{-1} with $NAC_{p^{-1}}$

$$\begin{tikzcd}
N & & \\
L \arrow[u, "n"] \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dl, phantom, "(2)"] \arrow[dr, phantom, "(1)"] & R \arrow[d] \\
G & D \arrow[l] \arrow[r] & H
\end{tikzcd}
\qquad
\begin{tikzcd}
N' & & \\
R \arrow[u, "n'"] \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dl, phantom, "(1)"] \arrow[dr, phantom, "(2)"] & L \arrow[d] \\
H & D \arrow[l] \arrow[r] & G
\end{tikzcd}$$

Over a Rule III

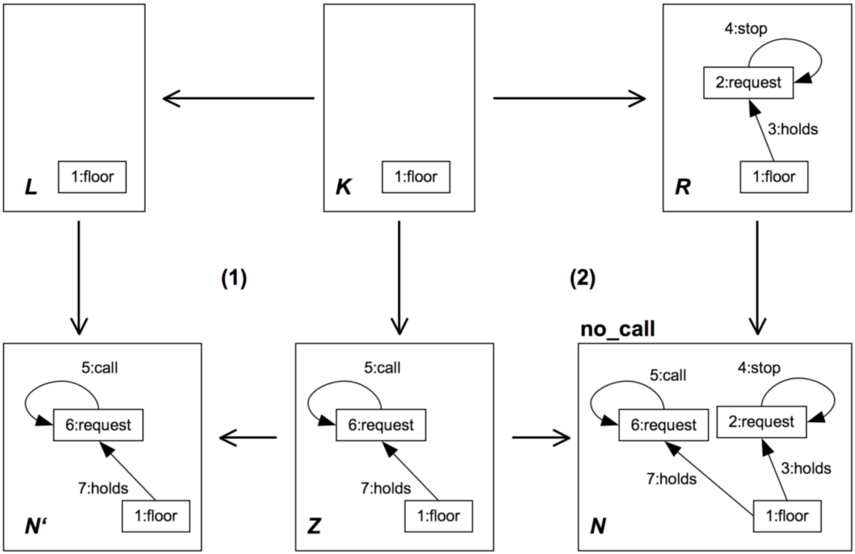


Figure: Shifting right NAC into left NAC for Elevator example

Over a Morphism I

- Consider a graph A with NACs, and some graph morphism $m : A \to B$.
- It is not enough to consider the PO of m and the single NACs on A in order to obtain the equivalent NACs on B.
- All overlaps have to be considered, where in addition graph elements not stemming from A may be glued.

Definition (construction of NACs on B from NACs on A with $m \to B$).
Consider the following diagram:

$$\begin{tikzcd}
N'_j \arrow[r, dotted, "e_{ji}"] & N_i \\
A \arrow[u, "n'_j"] \arrow[r, "m"] \arrow[ur, phantom, "(1)"] & B \arrow[u, dotted, "n_i"']
\end{tikzcd}$$

Over a Morphism II

$$\begin{tikzcd} N'_j \arrow[r, dotted, "e_{ji}"] & N_i \\ A \arrow[u, "n'_j"] \arrow[r, "m"] \arrow[ur, phantom, "(1)"] & B \arrow[u, dotted, "n_i"'] \end{tikzcd}$$

For each $NAC(n'_j)$ on A with $n'_j : A \to N'_j$ and $m : A \to B$, let

$$D_m(NAC(n'_j)) = \{NAC(n_i) | i \in I, n_i : B \to N_i\}$$

where I and n_i are constructed as follows:

- $i \in I$ iff (e_{ji}, n_i) with $e_{ji} : N'_j \to N_i$ jointly surjective
- $e_{ji} \circ n_i = n_i \circ m$
- $e_j i$ injective

Over a Morphism III

For each set of NACs $NAC_A = NAC(N_j)|j \in J$ on A the downward shift of NAC_A is then defined as:

$$D_m(NAC_A) = \cup_{j \in J} D_m(NAC(n'_j))$$

D_m is also called the *Downward shift of* NAC_A

Over a Morphism IV

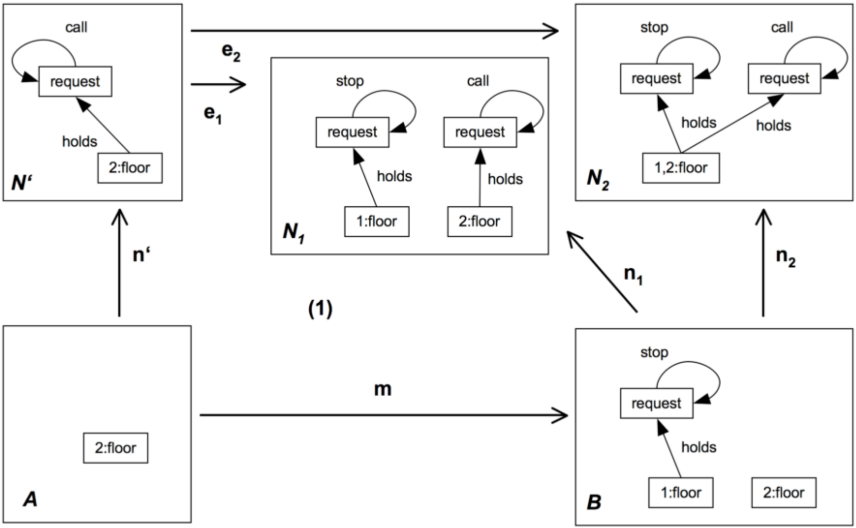


Figure: Example of shifting a NAC over a morphism

Outline of this talk

Negative Application Condition

Shifting NACs

Concurrent rules for a rule sequence

Concurrency I

Let t be a transformation via some rule sequence $p_0, \ldots, p_{n-1}, p_n$ with matches $g_0, \ldots, g_{n-1}, g_n$ and sets of NACs $NAC_{p_O}, \ldots, NAC_{p_{n-1}}, NAC_{p_n}$:

- In general there will be casual dependencies, so that is not always possible to summarize the transformation sequence via the *Parallelism Theorem*.
- It is possible to formulate a *Concurrency Theorem* expressing how to summarize such a sequence into one equivalent transformation step via a concurrent rule.

Without NACs I

Definition (concurrent rule for a rule sequence):

- $n = 0$ The *concurrent rule* p_c for p_0 is p_0 itself.
- $n \geqslant 1$ A concurrent rule p_c for the rule sequence $p_0, \ldots, p_{n-1}, p_n$is defined by $p_c = (l \circ k_c : K \to L, r \circ k_n : K \to R)$ as show in the following diagram, where
  - $p'_c : L'_c \leftarrow K'_c \to R'_c$ is a concurrent rule for the sequence $p_0, \ldots, p_{n-1}$
  - (e'_c, e_n) is jointly surjective
  - (1), (2), (3) and (4) are pushouts
  - (5) is a pullback

Without NACs II

$$
\begin{tikzcd}
L'_c \arrow[d] \arrow[dr, phantom, "(3)"] & K'_c \arrow[l] \arrow[r] \arrow[d] \arrow[drr, phantom, "(1)"] & R'_c \arrow[dr, "e'_c"] & & L_n \arrow[dl, "e_n"'] \arrow[r, phantom, "(2)"] & K_n \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(4)"] & R_n \arrow[d] \\
L & C_1 \arrow[l, "l"'] \arrow[rr] & & E \arrow[d, phantom, "(5)"] & & C_n \arrow[ll] \arrow[r, "r"] & R \\
& & & K \arrow[ull, dotted, "k_c"] \arrow[urr, dotted, "k_n"'] & & &
\end{tikzcd}
$$

Figure: Construction of a concurrent rule without NACs.

- The concurrent rule p_c may also be denoted as $p'_c *_E p_n$.

With NACs I

Definition (concurrent rule with NACs for a rule sequence):

- $n = 0$ The *concurrent rule* p_c with NACs for rule p_0 with NACs is p_0 with NACs itself.
- $n \geqslant 1$ A concurrent rule $p_c = p'_c *_E p_n$ with NACs for the rule sequence $p_0, \dots, p_{n-1}, p_n$ is defined recursively as in the definition without NACs and equals $p_c = (l \circ k_c : K \to L_c, r \circ k_n : K \to R)$

With NACs II

$$\begin{tikzcd}
N_i & & & & N_j & & \\
L'_c \arrow[u, "n_i"] \arrow[d] \arrow[dr, phantom, "(3)"] & K'_c \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & R'_c \arrow[dr, "e'_c"'] & & L_n \arrow[u, "n_j"'] \arrow[dl, "e_n"] & K_n \arrow[l] \arrow[r] \arrow[d] \arrow[dl, phantom, "(2)"] \arrow[dr, phantom, "(4)"] & R_n \arrow[d] \\
L_c & C_1 \arrow[l, "l"'] \arrow[rr] & & E \arrow[d, phantom, "(5)"] & & C_n \arrow[ll] \arrow[r, "r"] & R \\
& & & K \arrow[ull, dotted, "k_c"] \arrow[urr, dotted, "k_n"'] & & &
\end{tikzcd}$$

Figure: Construction of a concurrent rule with NACs.

with

- $NAC_{p_c} = DL_{p_c}(NAC_{L_n}) \cup D_{m_c}(NAC_{L'_c})$ where:
  - $D_{m_c}(NAC_{L'_c})$ is the Downward shift NAC over $m_c : L'_c \rightarrow L_c$
  - $DL_{p_c}(NAC_{L_n})$ with $NAC_{L_n} = \{NAC(n_j) | j \in J\}$ constructed as follows:

With NACs III

$$DL_{p_c}(NAC_{L_n}) = \cup_{j \in J} DL_{p_c}(NAC(n_j)) = \cup_{j \in J} L_p(D_{e_n}(NAC(n_j)))$$

with

- $p = L_c \leftarrow C_1 \rightarrow E$
- D_{e_n} is the Downward shift NAC over $e_n : L_n \rightarrow E$
- L_p is the left shift NAC over rule p

References

Lambers, Leen. (2010). Certifying Rule-Based Models using Graph Transformation. Elektrotechnik und Informatik der Technischen Universitat Berlin. https://depositonce.tu-berlin.de/handle/11303/2645

Construction of Concurrent Rules with NACs

Jonas S. Bezerra
jsbezerra@inf.ufrgs.br

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, Brasil
http://www.inf.ufrgs.br

September 19, 2016