Seminar

Graph-Grammatiken

Lehrstuhl für Informatik III
RWTH Aachen
Sommersemester 2004

Visuelles Design verteilter Systeme

Saskia Dedenbach **233859**

Inhaltsverzeichnis

1 Motivation

Das Prinzip verteilter Systeme beinhaltet, dass ein System in mehrere Module unterteilt ist, in denen zum Beispiel verschiedene Daten gespeichert und Prozesse durchgeführt werden. Die einzelnen Module sind miteinander vernetzt, damit ein Austausch von Informationen stattfinden kann. In Abbildung 1 ist ein Beispiel für ein verteiltes System zu sehen, das die Vorgänge bei der Verwaltung von Kunden in einer Bank beschreibt. Dabei bildet sowohl die Bank als auch jeder Kunde ein einzelnes Modul. Die Kanten zwischen diesen Modulen repräsentieren die Pfade, über die Daten zwischen der Bank und dem Kunden ausgetauscht werden.

Im Laufe dieser Arbeit wird ein mathematischer Ansatz vorgestellt, der das Prinzip verteilter Graphen, das in Kapitel 2 eingeführt wird, verwendet, um verteilte Systeme zu beschreiben. Zur Darstellung der Vorgänge in einem verteilten System wird darauf aufbauend eine Notation zur Transformation verteilter Graphen hergeleitet. Zwei Komponenten dieser Notation, die in Kapitel 3 und Kapitel 4 vorgestellt werden, sind die Konstruktion eines einfachen Pushouts und die Anwendung von verteilten Regeln auf verteilte Graphen. Die vollständige Notation zur Transformation verteilter Graphen wird in Kapitel 5 definiert.

2 Verteilte Graphen

Das Hauptmerkmal eines verteilten Graphen in diesem Ansatz ist die Verteilung des Graphen auf zwei Ebenen. Die übergeordnete Ebene besteht aus einem Netzwerkgraph. Jeder Knoten dieses Netzwerkgraphen beinhaltet einen Graphen der unteren Ebene, der als lokaler Graph bezeichnet wird (siehe Abbildung 2). Im Folgenden wird eine Definition eines verteilten Graphen mit Hilfe eines attributierten Graphen eingeführt. Attributierte Graphen sind Graphen, deren Knoten Attribute, wie zum Beispiel Strings, enthalten können. Die Definition liefert eine mathematische Beschreibung der Hierarchie eines verteilten Graphen.

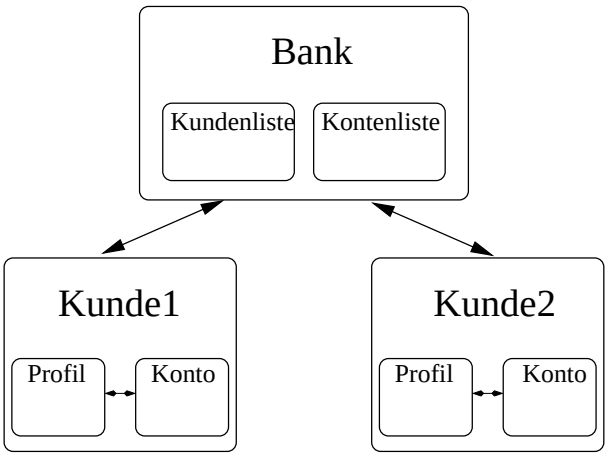

Abbildung 1: Beispiel eines verteilten Systems

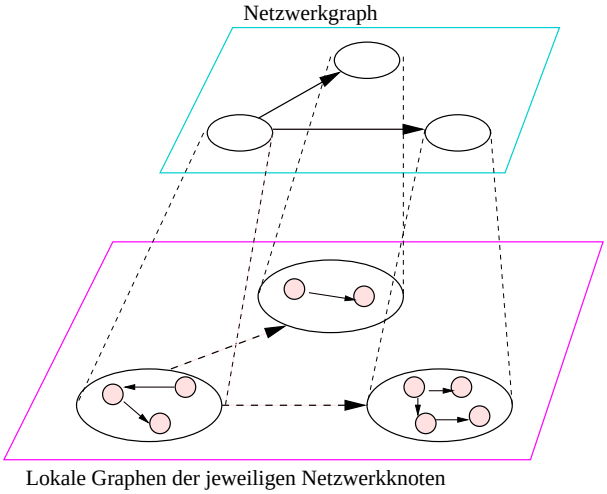

Abbildung 2: Aufbau verteilter Graph

Des Weiteren wird durch die Einführung von Graphmorphismen, Graphregeln und des verteilten Kontextgraphen ein doppelter Pushout hergeleitet. Ein doppelter Pushout beschreibt eine Notation zur Transformationen von verteilten Graphen gemäß einer verteilten Graphregel.

2.1 Markierte Graphen

Für die Definition des verteilten Graphen wird der Begriff des markierten Graphen benötigt, der wie folgt definiert wird.

Definition 2.1.1 (L-Markierter Graph)
Sei G_V eine Menge von Knoten und G_E eine Menge von Kanten, für die die Funktionen s, t : $G_E \to G_V$ so definiert sind, dass für alle $e \in G_E$ $s(e)$ der Quellknoten (source) und $t(e)$ der Zielknoten (target) der Kante e ist. Des Weiteren seien $l_V : G_V \to L$ und $l_E : G_E \to L$ zwei Markierungs-Abbildungen für eine Menge L von Bezeichnern. Mit $l_V(v)$ bzw. mit $l_E(e)$ wird ein Knoten v bzw. eine Kante e mit einem Bezeichner aus L markiert, wie z.B. einem Namen [HGGT].

$$\text{Der Graph } G = (G_V, G_E, s, t, l_V, l_E) \text{ ist ein L-markierter Graph.}$$

Ein L-markierter Graphmorphismus f zwischen zwei L-markierten Graphen G und H ist eine strukturerhaltende Abbildung $f : G \to H$, bei der die Bezeichner der Elemente des Graphen G erhalten bleiben. Somit gilt [HGGT]:

$$l_V^H(f_V(n)) = l_V^G(n) \text{ für alle Knoten } n \in G_V$$

und

$$l_E^H(f_V(e)) = l_E^G(n) \text{ für alle Kanten } e \in G_E.$$

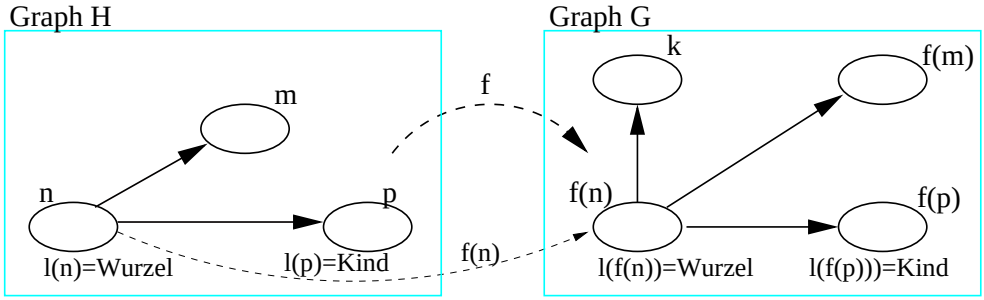

Abbildung 3: Beispiel eines gelabelten Graphen

Beispiel 2.1.2 (L-Markierter Graph)
Ein Beispiel eines markierten Graphen ist in Abbildung 3 zu sehen. Die Markierungsmenge ist in diesem Fall $L = \{Wurzel, Kind\}$. Der Knoten n erhält daraus den Bezeichner $l(n) = Wurzel$. Die Funktion f ist ein Graphmorphismus, denn f erhält die Struktur eines Graphen. In Abbildung 3 wird der Knoten $n \in H$ mit dem Bezeichner $l(n) = Wurzel$ auf den Knoten $f(n) \in G$ abgebildet. Dabei ist $l(n)$ nicht verändert worden, denn auch $f(n)$ hat den Bezeichner $Wurzel$. Ist dies für alle Knoten $i \in H$ und ihre Bilder $f(i) \in G$ der Fall, dann ist f ein L-markierter Graphmorphismus.

2.2 Verteilte Graphen

Mit Hilfe der eingeführten markierten Graphen und einer Menge von attributierten Graphen kann ein verteilter Graph wie folgt definiert werden.

Definition 2.2.1 (Verteilter Graph)
Sei A ein L-markierter Graph und $\tilde{A} : A \rightarrow \mathbf{AGr}$ eine Abbildung, bei der für alle $i \in V$ gilt $i \mapsto A_i, A_i\rangle$. Dabei ist A_i ein attributierter Graph. Dann ist das Tupel [HGGT]

$$\hat{A} = \langle A, \tilde{A} \rangle \text{ ein verteilter Graph.}$$

Das bedeutet, dass bei einem verteilten Graphen jedem Knoten $i \in A$ ein Graph $\tilde{A}(i)$ zugeordnet wird. A wird als der Netzwerkgraph und $\tilde{A}(i)$ als der lokale Graph des Netzwerkknoten i bezeichnet. Die Signatur eines lokalen attributierten Graphen $\tilde{A}(i)$ ist eine attributierte Graphsignatur AGS_i. Sie kann sich von der anderer lokaler Graphen in $\hat{A}$ unterscheiden.

Einer Netzwerkkante $e \in A$ mit Quellknoten i und Zielknoten j wird durch $\tilde{A}$ ein attributierter Graphmorphismus $h_e^A : \tilde{A}(i) \rightarrow \tilde{A}(j)$ zugeordnet. Dieser Graphmorphismus bildet den lokalen Graphen des Knoten i auf den des Knoten j ab. Die Klasse $\mathbf{Distr}(\mathbf{AGr})$ ist die Klasse aller verteilten Graphen $\hat{A} = \langle A, \tilde{A} \rangle$ über $\mathbf{AGr}$.

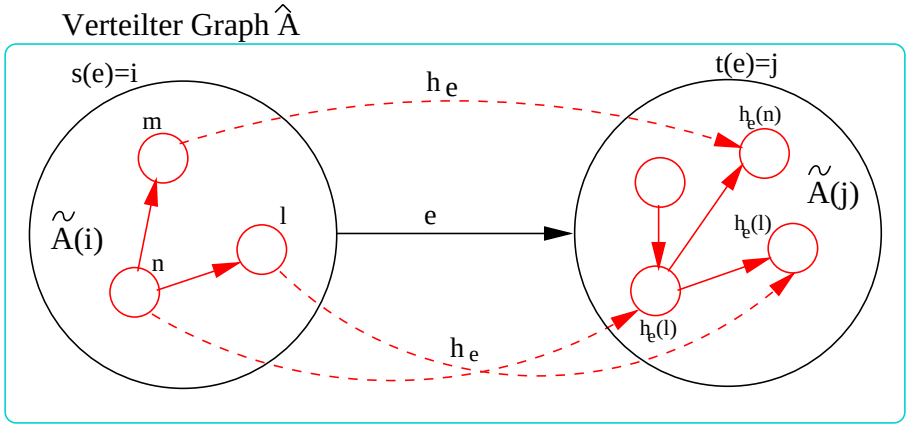

Abbildung 4: Beispiel eines verteilten Graphen

Beispiel 2.2.2 (Verteilter Graph)

Abbildung 4 zeigt ein Beispiel für einen kleinen verteilten Graphen in der Notation, die im Folgenden weiter verwendet wird. Dabei stellen die beiden großen schwarzen Kreise die Netzwerkknoten i und j dar. Die roten Graphen, die sich innerhalb dieser Knoten befinden, sind die zugehörigen lokalen Graphen $\tilde{A}(i)$ und $\tilde{A}(j)$. Bei der Funktion h_e, hier mit rot gestrichelten Pfeilen dargestellt, handelt es sich um den zuvor definierten Graphmorphismus der Netzwerkkante e. Dieser lokale Morphismus bildet den lokalen Graphen von $s(e) = i$ auf den von $t(e) = j$ ab.

Definition 2.2.3 (Verteilter Morphismus)

Ein verteilter Morphismus $\hat{f}$ ist ein Morphismus über der Klasse **Distr**(**AGr**) der verteilten Graphen. Er besteht aus einem Tupel [HGGT]

$$\hat{f} = \langle f, \tilde{f} \rangle : \langle A, \tilde{A} \rangle \to \langle B, \tilde{B} \rangle$$

von Funktionen, wobei $f : A \to B$ ein L-markierter Morphismus für den Netzwerkgraphen und $\tilde{f} : \tilde{A} \to \tilde{B} \circ f$ eine Menge von attributierten Morphismen auf lokaler Ebene ist. Für jeden Netzwerkknoten $i \in A$ enthält $\tilde{f}$ einen Morphismus $\tilde{f}_i : \tilde{A}(i) \to \tilde{B}(f(i))$. Die Elemente $\tilde{f}_i$ von $\tilde{f}$ sind lokale Graphmorphismen, die den lokalen Graphen eines Knoten $i \in A_V$ auf den von $f(i) \in B_V$ abbilden.

Bemerkung 2.2.4 (Schreibweise)

Für alle Knoten $i \in A_V$ sei $f(i)$ der Netzwerkknoten in B_V, auf den i durch f abgebildet wird. Dann wird mit f_i statt $\tilde{f}_i$ der lokale Morphismus zwischen dem lokalen Graphen $A_i = \tilde{A}(i)$ und dem lokalen Graphen $B_{f(i)}$ bezeichnet. Zu beachten ist, dass f_i nicht $f(i)$ entspricht. $f(i)$ ist die Abbildung des Netzwerknoten $i \in A_V$ und f_i bildet den lokalen Graphen $\tilde{A}(i)$ von i ab.

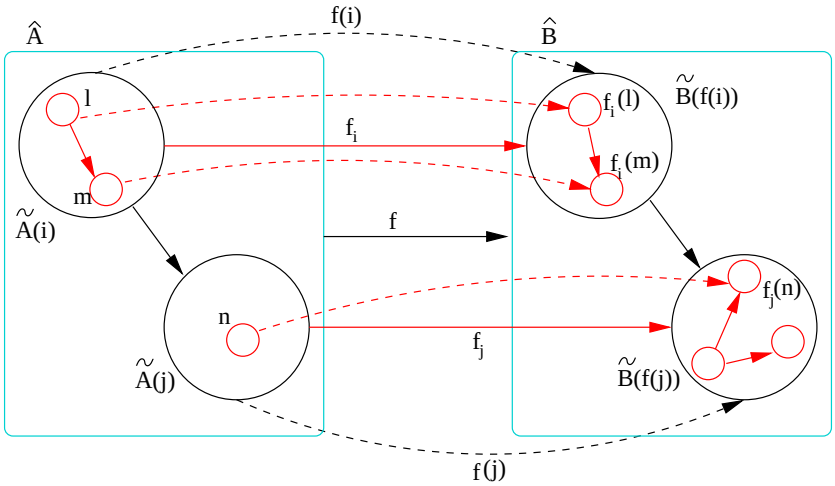

Abbildung 5: Beispiel eines verteilten Morphismus

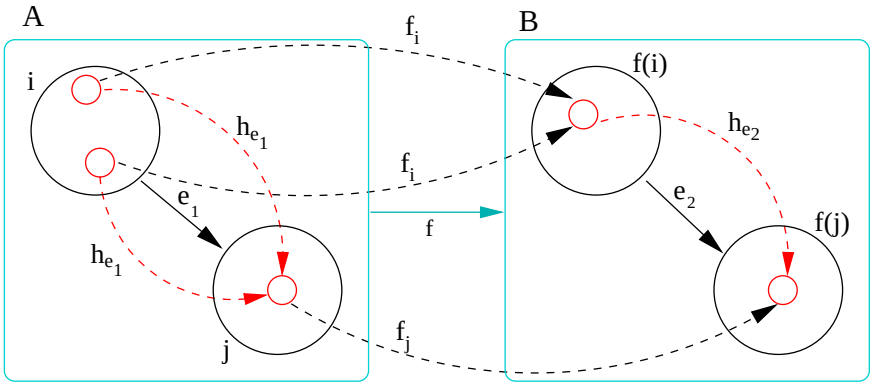

Abbildung 6: Bedingung für lokale Morphismen

Beispiel 2.2.5 (Verteilter Morphismus)

Abbildung 5 zeigt einen verteilten Morphismus zwischen den beiden Graphen $\hat{A}$ und $\hat{B}$. Die schwarzen Pfeile stehen für die Abbildung f der Elemente des Netzwerkgraphen, bei der mit $f(i)$ und $f(j)$ die Netzwerkknoten von A entsprechend der schwarz gestrichelten Pfeile nach B abgebildet werden. Die Funktionen f_i und f_j, hier in rot, stehen für die Morphismen, die die Elemente der lokalen Graphen von $i, j \in A_V$ auf die der lokalen Graphen von $f(i)$ und $f(j)$ in B abbilden (rot gestrichelte Pfeile).

Bemerkung 2.2.6 (Eigenschaft lokaler Morphismen)

Seien $h_e^A : \tilde{A} \to \tilde{A}$ und $h_e^B : \tilde{B} \to \tilde{B}$ attributierte Morphismen zwischen zwei lokalen Graphen von A bzw. B. Dann gilt für alle Kanten $e : i \to j \in A_E$ mit $i, j \in A_V$ [**?**]:

$$f_j \circ h_e^A = h_{f(e)}^B \circ f_i$$

Das bedeutet, dass der resultierende lokale Graph $\tilde{B}(f(j))$ (vgl. Abbildung 6) derselbe ist, wenn zuerst der Morphismus h_e innerhalb eines Graphen und dann

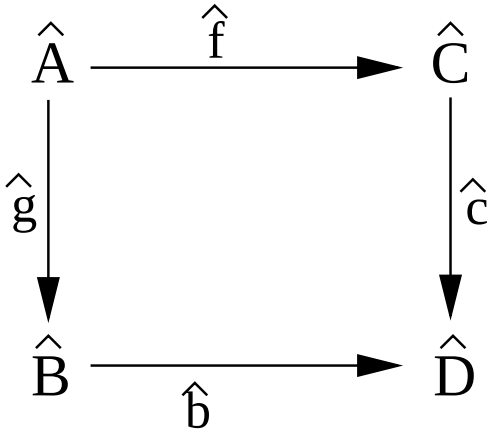

Abbildung 7: Herleitung eines einfachen verteilten Pushouts

der Morphismus f zwischen den verteilten Graphen $\tilde{A}$ und $\tilde{B}$ angewandt wird, oder umgekehrt. Die roten Kanten in Abbildung 6 beschreiben die Morphismen h_e, zwischen den lokalen Graphen innerhalb eines verteilten Graphen. Die schwarzen Kanten stellen die Morphismen f_x zwischen den lokalen Graphen zweier verschiedener verteilter Graphen $\hat{A}$ und $\hat{B}$ dar.

Bemerkung 2.2.7 (Netzwerk-Injektivität)

Ein verteilter Morphismus $\hat{f} = \langle f, \tilde{f} \rangle : \hat{A} \to \hat{B}$ wird netzwerk-injektiv (kurz: n-injektiv) genannt, wenn der Netzwerkmorphismus f injektiv ist. Das heißt, dass es keine zwei Netzwerkknoten oder -kanten $i, j \in A$ gibt, die auf das gleiche Bild $f(i) = f(j) \in B$ abgebildet werden. Normale Injektivität liegt bei $\hat{f}$ vor, falls für alle Netzwerkknoten $i \in A_V$ gilt, dass f_i injektiv ist [HGGT].

3 Einfacher verteilter Pushout

Das Ziel der nächsten beiden Kapitel ist es, eine Notation für die Transformation verteilter Graphen zu beschreiben. Der vorgestellte Ansatz, beruht auf dem Prinzip eines doppelten Pushouts. Ein doppelter Pushout verteilter Graphen setzt sich zusammen aus einer verteilten Regel, einem verteilten Vorkommen, einem verteilten Kontextgraphen und einem einfachen verteilten Pushout. Um den doppelten Pushout zu konstruieren, müssen diese Elemente für verteilte Graphen definiert werden. Begonnen wird hier mit dem einfachen Pushout.

Ein einfacher verteilter Pushout besteht aus einem verteilten Graphen $\hat{D}$. Dieser ergibt sich aus der Vereinigung zweier verteilter n-injektiven Morphismen $\hat{f}$ und $\hat{g}$, die auf den gleichen Graphen $\hat{A}$ angewandt werden. Das Prinzip der Herleitung von $\hat{D}$ ist in Abbildung 7 zu sehen. Die dort aufgeführten Graphen $\hat{B}$ und $\hat{C}$ ergeben sich aus der Anwedung von $\hat{g}$ bzw. $\hat{f}$ auf den Graphen $\hat{A}$.

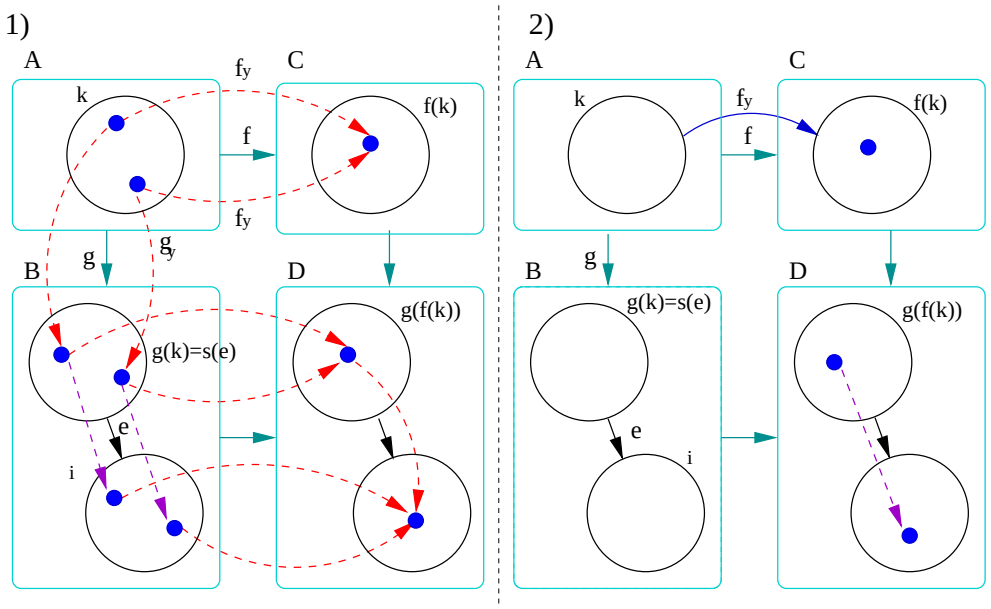

Abbildung 8: Beispiele, die die Lokalitätsbedingung nicht erfüllen

3.1 Lokalitätsbedingungen

Im Folgenden wird die genaue Konstruktion eines verteilten Pushouts über der Klasse **Distr**(**AGr**) erläutert. Dabei soll komponentenweise vorgegangen werden. Komponentenweise heißt in diesem Fall, dass in einem ersten Schritt der Netzwerkgraph D von $\hat{D}$ konstruiert wird. Im zweiten Schritt werden die lokalen Graphen der Netzwerkknoten von D hergeleitet. Damit die Konstruktion komponentenweise erfolgen kann, muss sichergestellt werden, dass die Lokalitätsbedingungen (*locality conditions*) für die Morphismen $\hat{f}$ und $\hat{g}$ erfüllt werden. Diese sind wie folgt definiert:

Definition 3.1.1 (Lokalitätsbedingungen)

Seien $\hat{f} : \hat{A} \to \hat{C}$ und $\hat{g} : \hat{A} \to \hat{B}$ zwei n-injektive verteilte Morphismen. $\hat{f}$ und $\hat{g}$ erfüllen die Lokalitätsbedingungen, falls [HGGT]:

1. für alle $e \in C_E - f_E(A_E)$ und für alle $y \in A_V$ gilt: wenn $f(y) = s(e)$, dann ist g_y bijektiv.
2. für alle $e \in B_E - g_E(A_E)$ und für alle $y \in A_V$ gilt: wenn $g(y) = s(e)$, dann ist f_y bijektiv.

Die erste Bedingung sagt aus, dass der lokale Morphismus $g_y : \tilde{A(y)} \to \tilde{B}(g(y))$ eines Netzwerkknotens $y \in A_V$ bijektiv sein muss, falls y die Wurzel einer in C_E neu hinzugekommenen Kante ist. Das heißt, dass sie kein Urbild in A_E hat. Die zweite Bedingung beschreibt den umgekehrten Fall. Hier muss f_y bijektiv sein, falls für einen Knoten $y \in A$ eine Kante $e : g(y) \to z$ nur in B existiert.

Beispiel 3.1.2 (Lokalitätsbedingung)
Um die Bedingungen zu verdeutlichen, sind in Abbildung 8 zwei Beispiele für Graphmorphismen gegeben, die die Lokalitätsbedingungen nicht erfüllen. In beiden Fällen existiert eine neue Netzwerkkante e in B, deren Wurzelknoten ein Urbild k im Graphen A hat. Daher muss der lokale Morphismus f_k bijektiv sein, um die Lokalitätsbedingung zu erfüllen. Im ersten Bild bildet f_k zwei Knoten des lokalen Graphen von k auf das gleiche Bild in $f(k)$ ab. Dies bedeutet, dass f_k nicht injektiv und somit auch nicht bijektiv ist.

In der zweiten Abbildung des Beispiels 8 existiert im lokalen Graphen von $f(k) \in C$ ein Knoten auf den kein lokaler Knoten aus k abgebildet wird. Deswegen ist f_k nicht surjektiv, dass heißt, dass auch hier keine Bijektivität vorliegt. Damit sind in beiden Fällen die Lokalitätsbedingungen verletzt. In diesem Bild sind auch die Konsequenzen der Bedingungen zu sehen. Werden sie von zwei verteilten Morphismen $\hat{f}$ und $\hat{g}$ nicht erfüllt, so können ungewollte Seiteneffekte auftreten. Ist der Morphismus f_k nicht bijektiv, so bedeutet dies, dass f_k im lokalen Graphen von k eine Transformation, wie das Löschen oder das Hinzufügen eines Knoten, vornimmt. Bei der Konstruktion von $\hat{D}$ führt dies dazu, dass in dem lokalen Graphen des Zielknoten der neuen Kante $e : g(k) \to i \in B$ die gleiche Transformation bei der Abbildung in $\hat{D}$ stattfindet, wie im Quellknoten. Für das erste Beispiel der Abbildung 8 bedeutet das, dass bei der Abbildung von $\hat{B}$ nach $\hat{D}$ das Löschen eines Knoten des lokalen Graphen von $g(k)$ dazu führt, dass auch ein lokaler Knoten des Netzwerkknoten i gelöscht wird. Gleiches gilt für das Hinzufügen eines lokalen Knotens, wie es im zweiten Bild zu sehen ist

3.2 Komponentenweise Pushout-Konstruktion

Bei Erfüllung der Lokalitätsbedingungen kann für einen verteilten Graphen $\hat{A}$ und zwei n-injektive verteilte Morphismen ein Pushout komponentenweise konstruiert werden. Dabei wird folgendermaßen vorgegangen: Als erstes wird aus A durch f und g der Pushout D des Netzwerkgraphen erstellt (siehe Schritt 1 des Algorithmus aus 3.2)). Dann wird für jeden Knoten $i \in D_V$ ein Pushout anhand der lokalen Graphmorphismen f_i und g_i über $\mathbf{AGr}(\mathbf{AGS}^{\mathbf{A}}_{\mathbf{i}})$ konstruiert. Wenn ein Netzwerkknoten j aus A nicht in D existiert, wird er durch f und g gelöscht. In diesem Fall wird der lokale Graph von j nicht bei der Pushout-Konstruktion beachtet.

Die lokalen Graphmorphismen h^D_e, die den Kanten des Pushout-Graphen entsprechen, müssen den Eigenschaften der lokalen Pushouts angepasst werden. Wie in Abbildung 9 zu sehen ist, bildet h^A_e einen Quellknoten und einen Zielknoten ab. Da der lokale Graph des Knoten $p \in D$ einen Knoten hat, der weder ein Quell- noch ein Zielknoten ist, ist die Abbildung h^A_e für diesen lokalen Knoten nicht definiert. Das bedeutet, dass die Morphismen h^D_e nicht ohne weiteres aus den anderen Graphen, zum Beispiel $\hat{A}$, übernommen werden können, da sich die lokalen Graphen der Quell- und Zielknoten der Kante e in D von denen der entsprechenden Kante aus A unterscheiden können.

Dies tritt auf, wenn in $s(e)$ oder $t(e)$ Veränderungen durch die lokalen Morphismen von $\hat{f}$ und $\hat{g}$ stattfinden. Daher müssen auch die lokalen Morphimen h^D_e für den einfachen Pushout konstruiert werden. Das genaue Vorgehen beim Herleiten aller Elemente des Pushouts wird im folgenden Algorithmus erläutert.

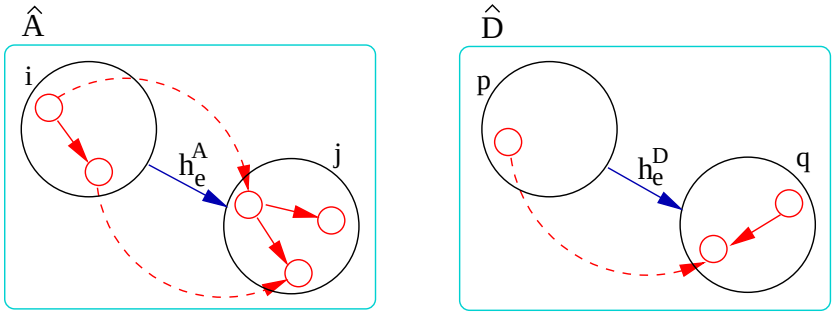

Abbildung 9: Beispiele lokaler Morphismen

Algorithmus 3.2.1 (Komponentenweisen Pushout-Konstruktion)

Seien $\hat{f} : \hat{A} \to \hat{C}$ und $\hat{g} : \hat{A} \to \hat{B}$ zwei n-injektive verteilte Morphismen, die die Lokalitätsbedingungen erfüllen. Der Pushout von $\hat{f}$ und $\hat{g}$ wird dann wie folgt definiert [HGGT]:

1. Konstruiere den Netzwerkgraphen D des Pushout von f und g für den Netzwerkgraphen von $\hat{A}$. Dafür werden die Knoten und Kanten aus A, die nicht durch f und g gelöscht werden in D übernommen. Die Knoten und Kanten, die durch f bzw. g in C bzw. B neu hinzukommen sind, werden an diesen Graphen D angehängt. Die Pushoutmorphimen $b : B \to D$ und $c : C \to D$ sind Inklusionen, das heißt, dass B und C jeweils einem Teilgraph von D entsprechen, auf den sie durch b bzw.c abgebildet werden.

2. Es existiert für alle $x \in D_V$ höchstens ein Knoten $y \in A_V$ mit $c_V \circ f_V(y) = x$, weil f und g injektiv sind. Daher kann mit P_x der Pushout des lokalen Graphen von y bezüglich f_y und g_y eindeutig bezeichnet werden. Des Weiteren ergeben die Inklusionen $c_{f(y)} : C_{f(y)} \to D_x$ und $d_{g(y)} : B_{g(y)} \to D_x$ die lokalen Pushoutmorphismen. Die lokalen Pushouts der Knoten des Netzwerk-Pushouts werden mit $\tilde{D} : D \to AGr$ wie folgt definiert. Für alle $x \in D_V$ gilt:

   $\tilde{\mathbf{D}}(\mathbf{x}) =$

   - $\mathbf{P_x}$, falls es einen Knoten $y \in A_V$ gibt, der auf $x \in D_V$ abgebildet wird. P_x ist der lokale Pushout der beiden Morphismen f_y und g_y über dem lokalen Graphen von $y \in A$. Die Konstruktion von P_x erfolgt auf die gleiche Weise, wie sie in Punkt 1 für den Netzwerkgraphen geschildert ist.
   - $\tilde{\mathbf{C}}(\mathbf{z})$, falls ein Netzwerkknoten $z \in C_V$ existiert, der kein Urbild in A hat und für den gilt: $c(z) = x$. $\tilde{C}(z)$ ist der lokaler Graph des Knoten z in C.
   - $\tilde{\mathbf{B}}(\mathbf{z})$, falls ein Netzwerkknoten $z \in B_V$ existiert, der kein Urbild in A hat und für den gilt: $b(z) = x$. $\tilde{B}(z)$ ist der lokaler Graph des Knoten z in B.

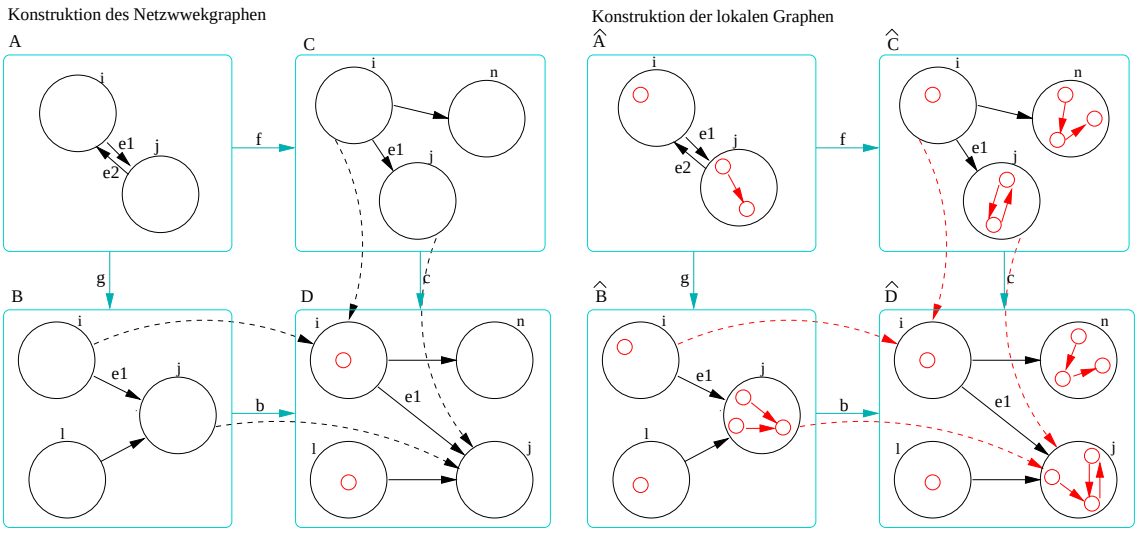

Abbildung 10: Pushout des Graphen $\hat{A}$ für die Morphismen $\hat{f}$ und $\hat{g}$

Die lokalen Morphismen für die Netzwerkkanten x aus D werden durch Zurückführung auf den Morphismus des Urbilds e von x hergeleitet. Dies geschieht wie folgt: zuerst wird der Quellknoten von x auf den Quellknoten von e abgebildet. Dann wird der lokale Morphismus der Kante e durchgeführt und der daraus resultierende Zielgraph auf den lokalen Graphen des Zielknoten von x in $\hat{D}$ abgebildet. Aus der Konkatenation dieser drei Morphismen ergibt sich der lokale Morphismus h_x^D.

Die Funktion $\tilde{D}(x)$, die den Netzwerkkanten x des Pushouts einen lokalen Morphismus h_x zuordnet, ist wie folgt definiert:

$\mathbf{\tilde{D}(x)} =$

- $\mathbf{f_{t(x)} \circ c_{f(t(x))} \circ h_e^A \circ f_{s(e)}^{-1} \circ c_{f(s(e))}^{-1}}$ (siehe Abbildung 11.2) , falls ein Kante $e \in A_E$ existiert, die auf x abgebildet wird und, falls die Morphismen folgende Bedingung erfüllen: $b_{g(t(e))} \circ h_{g(e)}^B \circ g_{s(e)} = c_{f(t(e))} \circ h_{f(e)}^C \circ f_{s(e)}$ (siehe Abbildung 11.1). Dies ist aus der Tatsache hergeleitet, dass gilt: $h_x^D(s(x)) = t(x) \iff c(f(h_e^A(s(e)))) = c(f(t(e)))$.
- $\mathbf{c_{t(e)} \circ h_e^C \circ c_{s(e)}^{-1}}$ (siehe Abbildung 11.3), falls es eine Kante $e \in C_E$ gibt, die kein Urbild in A_E hat. Dies folgt aus: $h_x^D(s(x)) = t(x) \iff c(h_e^C(s(e))) = c(t(e))$.
- $\mathbf{b_{t(e)} \circ h_e^B \circ b_{s(e)}^{-1}}$ (siehe Abbildung 11.4), falls es eine Kante $e \in B_E$ gibt, die kein Urbild in A_E hat. Dies folgt aus: $h_x^D(s(x)) = t(x) \iff b(h_e^B(s(e))) = b(t(e))$.

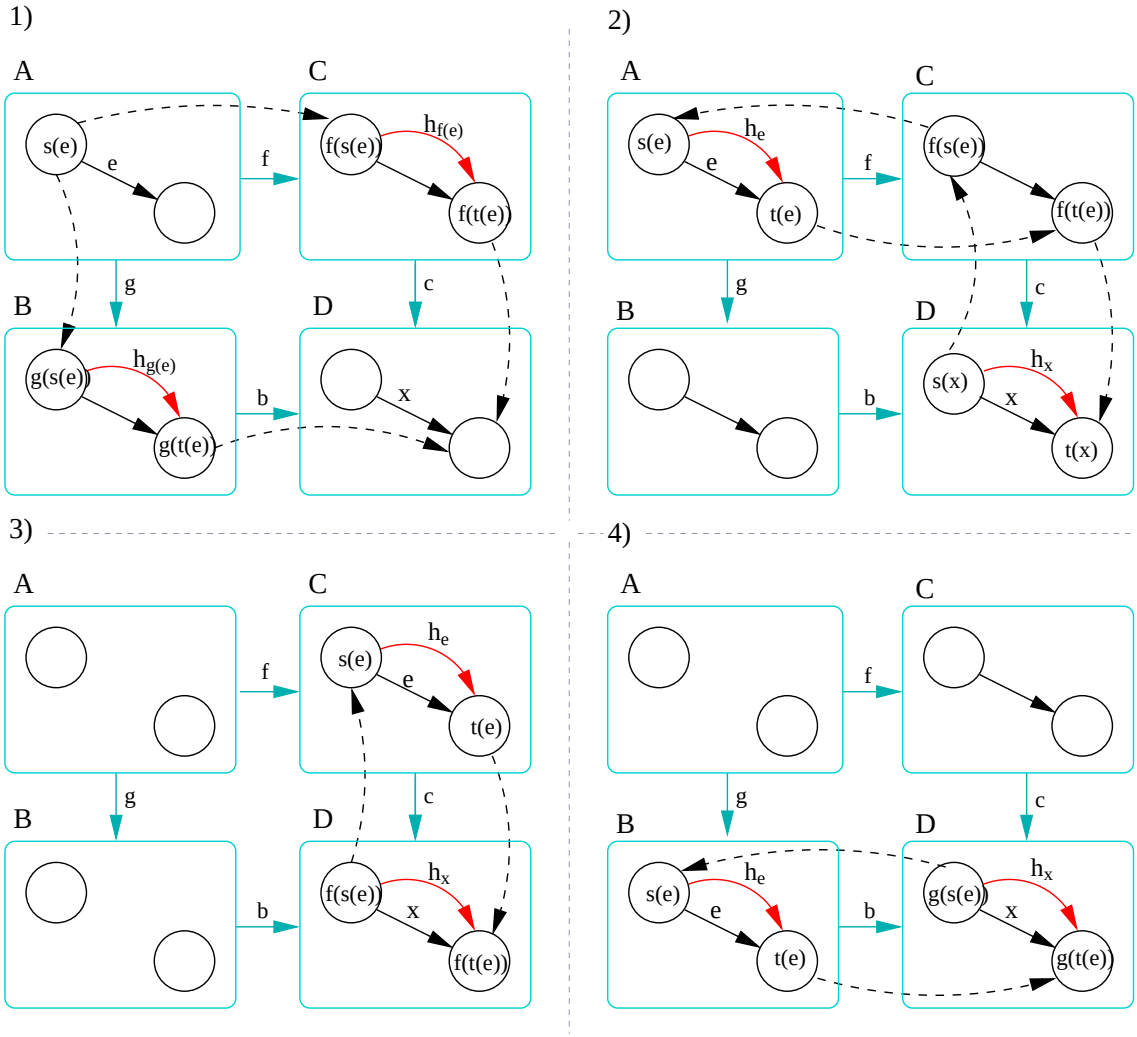

Abbildung 11: Konstruktion der Kanten und Morphismen eines Pushouts

Beispiel 3.2.2 (Komponentenweise Pushout-Konstruktion)
In Abbildung 10 ist der Pushout der beiden verteilten Graphmorphismen $\hat{f}$ und $\hat{g}$ dargestellt. Bevor mit der Konstruktion dieses Pushouts begonnen werden kann, muss die Erfüllung der Lokalitätsbedingungen überprüft werden. In diesem Beispiel existiert kein Netzwerkknoten in A der Quelle einer durch f oder g neu hinzugekommenen Kante ist. Somit gibt es keine Anforderungen an $\hat{f}$ und $\hat{g}$.

Da die Lokalitätsbedingungen erfüllt sind, kann mit der Konstruktion des Netzwerkgraphen D des Pushouts begonnen werden (siehe linke Abbildung 10). In diesem Fall besteht D aus den beiden Knoten i, j und der Kante $e1$, da diese nicht von f und g gelöscht wurden. Die Kante $e2$ ist nicht in D, denn sie existiert werder in B noch in C. Die Knoten $l \in B$ und $n \in C$, die durch g bzw. f hinzugefügt wurden, werden in D übernommen. Das Gleiche gilt für die neuen Kanten aus B und C.

Danach werden die lokalen Graphen des Pushouts ermittelt, wie in der rechten Abbildung des Beispiels 10 zu erkennen ist. Für die Netzwerkknoten $l \in B$ und $n \in C$ gilt, dass sie kein Urbild in A haben. Daher werden ihre lokalen Graphen in

D übernommen. Für die Knoten $x, y \in D$ existieren Urbilder i, j in A. In diesem Fall muss für den lokalen Graphen von x bzw. y jeweils der Pushout von f_i, g_i für den lokalen Graphen von i bzw. von f_j, g_j für den Graphen von j gebildet werden. Bei dem Netzwerkknoten x ist der lokale Graph konstant geblieben, da $\tilde{A}(i)$ durch keinen der Morphismen verändert wurde. $\tilde{D}(y)$ muss aus den lokalen Graphen von $f(j) \in C$ und $g(j) \in B$ konstruiert werden.

Die Morphismen h_x^D zwischen den lokalen Graphen werden durch die Konkatenation der Morphismen hergeleitet, vergleichbar zu Abbildung 11. Wie im zweiten, dritten und vierten Bild zu erkennen ist, treten bei dieser Konkatenation drei verschiedene Fälle auf. Diese ergeben sich daraus, dass das Urbild e der Kante $x \in D$ entweder in C und nicht in B (11.3), oder in B und nicht in C (11.4) oder in allen drei Graphen (A, B und C) liegen kann (11.2). Der Ansatz zur Herleitung von h_x^D ist in allen Fällen der gleiche und besteht aus drei Abbildungen. Die Erste bildet den lokalen Graph des Quellkonten der Kante $x \in D_E$ auf den Graphen des Knoten $s(e)$ ab, wobei die Kante e das Urbild von x ist. Der Morphismus h_e, der $s(e)$ auf den lokalen Graphen des Zielknoten $t(e)$ abbildet, entspricht der zweiten Abbildung. Die Dritte bildet diesen lokalen Zielgraphen zurück nach $\tilde{D}$ auf den lokalen Graphen von $t(x)$ ab. Die Morphismen h_x^D sind für alle Kanten $x \in D$ jeweils gleich der Konkatenation dieser drei Abbildungen.

4 Anwendung verteilter Regeln

Die wichtigste Komponente des doppelten Pushouts, der im Laufe dieses Kapitels hergeleitet wird, ist die verteilte attributierte Regel. Eine Regel definiert die Transformation, die an einem Graphen durchgeführt werden soll und bestimmt somit das Resultat des doppelten Pushouts.

4.1 Verteilte Regeln

Eine verteilte Regel besteht aus einer Netzwerkregel und einer Menge von einfachen attributierten Regeln, den lokalen Regeln. Bei ihrer Anwendung auf einen konkreten verteilten Graphen werden zunächst die Transformationen entsprechend der Netzwerkregel auf der Netzwerkebene ausgeführt. Auf den daraus resultierenden Netzwerkgraphen werden dann die lokalen Regeln angewandt. Mögliche Transformationen auf beiden Ebenen sind das Löschen, das Hinzufügen und das einfache Erhalten eines Knoten oder einer Kante.

Definition 4.1.1 (Verteilte attributierte Regel)

Seien $\hat{l}$ und $\hat{r}$ zwei n-injektive verteilte Morphismen über $\hat{I}$, so dass $\tilde{p} = \{p_i = (L_{l(i)} \xleftarrow{l_i} I_i \xrightarrow{r_i} R_{r(i)}) : i \in I_V\}$ eine Menge attributierter Regeln ist. Für jeden Netzwerkknoten $i \in I$ existiert eine lokale Regel p_i in $\tilde{p}$. Die Regel $p = (L \xleftarrow{l} I \xrightarrow{r} R)$ ist die Netzwerkregel. Das Tupel

$$\hat{p} = \langle p, \tilde{p} \rangle = \hat{L} \xleftarrow{\hat{l}} \hat{I} \xrightarrow{\hat{r}} \hat{R}$$

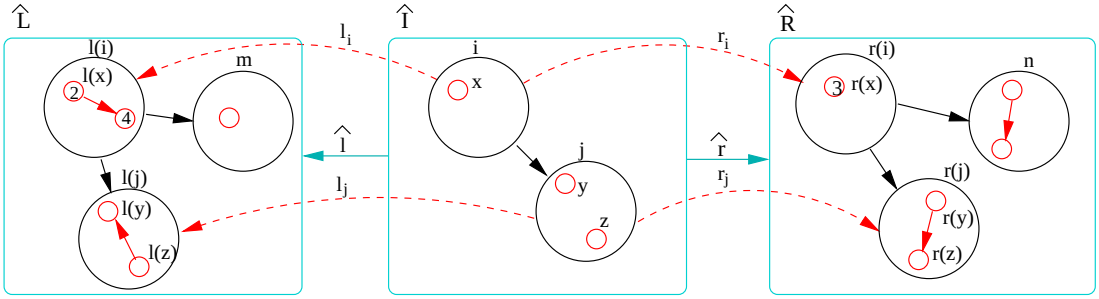

Abbildung 12: Beispiel einer verteilten Regel

bildet eine verteilte attributierte Regel [HGGT]. Zu beachten ist, dass nur für die Netzwerkknoten, die sich im Graphen $\hat{I}$ der verteilten Regel befinden, lokale Regeln existieren. Der Graph $\hat{I}$ besteht aus den Elementen des Graphen $\hat{L}$, die bei Anwendung der Regel konstant bleiben. Ein Netzwerkknoten, der in $\hat{L}$ vorkommt, aber nicht in $\hat{I}$, wird durch die Regel gelöscht. Im Gegensatz dazu wird ein Knoten, der nicht in $\hat{I}$ ist, aber in der rechten Regelseite $\hat{R}$, dem Graphen neu hinzugefügt. Das Gleiche gilt für die Netzwerkkanten. Wichtig beim Löschen oder Hinzufüg en eines Netzwerkknoten j ist, dass gleichzeitig auch der lokale Graph von j gelöscht oder hinzugefügt wird, da für ihn keine lokale Regel existiert. Bei Netzwerkkanten e muss der lokale Morphismus h_e gelöscht oder hinzugefügt werden.

Beispiel 4.1.2 (Verteilte Regel)
Abbildung 12 zeigt ein Beispiel einer verteilten Regel. Diese würde in einem passenden Graphen, auf den sie angewendet werden kann, bewirken, dass ein Netzwerkknoten, hier der Knoten m, gelöscht wird. Statt dessen kommt der Netzwerkknoten n hinzu. Wie in diesem Bild zu erkennen ist, enthält der Graph $\hat{I}$ genau die Elemente des Graphen $\hat{L}$, die durch die Transformation unverändert bleiben.

Die lokalen Regeln, hier durch rote Pfeile gekennzeichnet, werden nur auf die lokalen Graphen der Netzwerkknoten k, die erhalten bleiben, angewandt (hier die Knoten i und j). Das heißt auf die $k \in L$, die auch im Graphen von I vorhanden sind. Dabei bewirkt zum Beispiel die Regel p_i für den lokalen Graphen von i, dass eine Kante und ihr Zielknoten gelöscht werden, während p_j nur die Richtung der Kante zwischen den beiden lokalen Knoten umdreht.

Eine Auswirkung sowohl der Netzwerkregel als auch der lokalen Regeln kann außerdem sein, dass Knoten mit Bezeichnern versehen werden, bzw. umbenannt werden. Das Gleiche gilt für die Zuweisung von Attributen, zum Beispiel Zahlenwerte oder Variablen, zu den Knoten.

4.2 Verteilte Klebe-Bedingung und Vorkommen

Um die verteilte Regel $\hat{p}$ auf einen verteilten Graphen $\hat{A}$ anwenden zu können, wird in $\hat{A}$ ein Vorkommen $\hat{m} : \hat{L} \to \hat{G}$ der linken Regelseite benötigt. Dieses Vorkommen muss die verteilte Klebe-Bedingung (*gluing condition*) erfüllen, damit die Regel angewandt werden kann. Diese verteilte Klebe-Bedingung stellt sicher, dass der

Teilgraph, der durch die Regel verändert wird, wieder an die Kanten des restlichen Graphen, den die Regel nicht betrifft, "angeklebt" werden kann. Ein weiterer Grund für die Klebe-Bedingung ist, dass sie die Existenz des verteilten Kontextgraphen $\hat{C}$ der Regel $\hat{p}$ und des Vorkommens $\hat{m} : \hat{L} \to \hat{G}$ garantiert. Dieser Kontextgraph ist eine weitere Komponente des doppelten Pushouts. Er besteht aus den Elementen des Graphen $\hat{G}$, die entweder nicht von der Regel $\hat{p}$ betroffen sind oder sich im Graphen $\hat{I}$ befinden. Das heißt, dass der verteilte Kontextgraph der Teilgraph von $\hat{G}$ ist, der sowohl vor als auch nach der Anwendung der verteilten Regel existiert und das $\hat{I}$ eine Teilmenge von $\hat{G}$ ist.

Die verteilte Klebe-Bedingung besagt, dass für ein Vorkommen $\hat{m}$ die einfachen attributierten Klebe-Bedingungen für die Netzwerkregel und für alle lokalen Regeln erfüllt sein müssen. Diese bestehen aus der Hängenden Bedingung und der Identitäts-Bedinung. Außerdem müssen die Verbindungsbedingung und die Netzwerkbedingung beachtet werden. Die Verbindungsbedingung sichert die Konsistenz der lokalen Morphismen h_e für alle Kanten $e \in G - m(L - I)$. Das bedeutet, dass ein Morphismus h_e, der nicht durch die verteilte Regel $\hat{p}$ gelöscht wurde, nach der Transformation immer noch auf den eventuell veränderten lokalen Graphen von $s(e)$ anwendbar ist.

Bei der Netzwerkbedingung kommt es darauf an, dass auch der lokale Graph eines Netzwerkknoten n entfernt wird, falls die Regel n löscht. Außerdem muss der lokale Zielgraph einer Kante e in sich vollständig definiert sein, falls e gelöscht wird. Dies bedeutet, dass die Attribute der Elemente des lokalen Graphen von $t(e)$ nicht durch den Graphmorphismus h_e importiert werden dürfen, denn beim Wegfallen dieser Kante wäre der Graph sonst unvollständig. Ein Beispiel für eine solche Situation ist ein lokaler Knoten $i \in \tilde{A}(t(e))$, dessen Attribut $i.x = j.x + k.x$ ist. Die Attribute $j.x$ und $k.x$ werden durch h_e aus dem lokalen Graphen von $s(e)$ auf $i.x$ abgebildet und definieren so das Attribut von i. Wird die Kante e gelöscht, fällt auch der Morphismus h_e weg und i ist unvollständig.

Es ist wichtig, dass die verteilte Klebe-Bedingung für ein verteilten Morphismus $\hat{m} : \hat{L} \to \hat{G}$ und eine verteilte Regel $\hat{p}$ erfüllt ist, bevor $\hat{p}$ angewendet wird, damit die Konsistenz des Graphen gewährleistet ist.

Definition 4.2.1 (Verteilte Klebe-Bedingung und Vorkommen)

Gegeben sei die verteilte Regel $\hat{p} = \hat{L} \xleftarrow{\hat{l}} \hat{I} \xrightarrow{\hat{r}} \hat{R}$. Außerdem sei die Abbildung $\hat{m} : \hat{L} \to \hat{G}$ ein n-injektiver verteilter Morphismus. $\hat{m}$ ist ein verteiltes Vorkommen, falls es die folgenden Bedingungen erfüllt [HGGT]:

1. Netzwerkebene:

   $m : L \to G$ erfüllt die einfache Klebe-Bedingung für die Netzwerkregel $p = (L \leftarrow I \rightarrow R)$.

2. Lokale Ebene:

   $m_{l(i)} : L_{l(i)} \to G_{m(l(i))}$ erfüllt die einfache Klebe-Bedingung für die lokale Regel $p_i = (L_{l(x)} \leftarrow^{l_x} I_x \rightarrow^{r_x} R_{r(x)})$ für alle $i \in I_V$. Beachte dass $m_{l(i)}$ nicht das Gleiche ist, wie $m(l(i))$. Ersteres bezeichnet den lokalen Morphismus von i, während mit $m(l(i))$ die Abbildung des Netzwerkknoten gemeint ist.

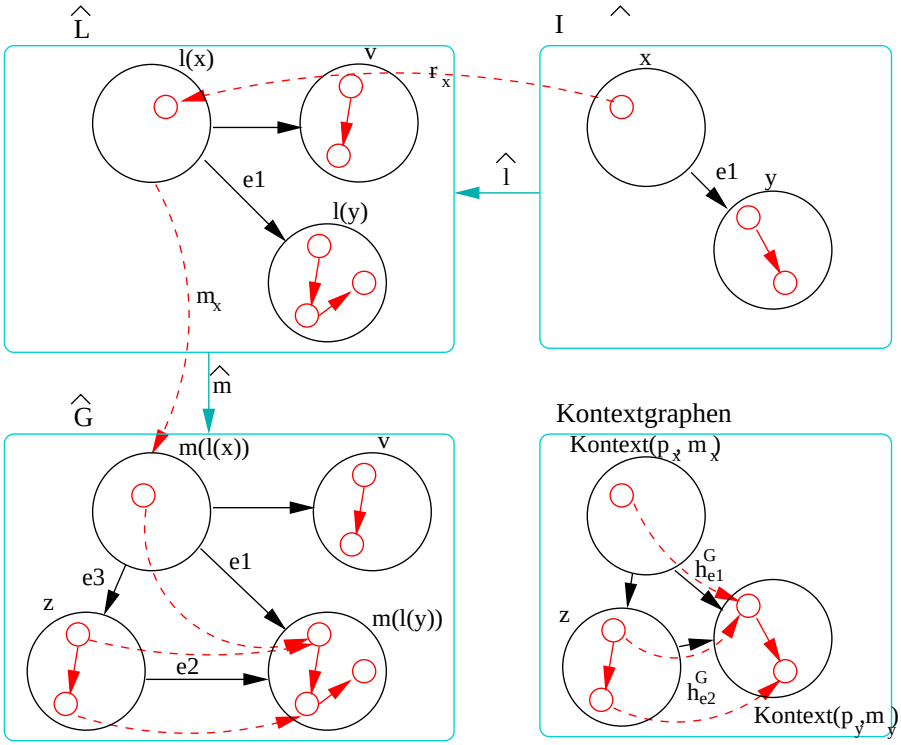

Abbildung 13: Beispiel für die Erfüllung der Verbindungsbedingung

3. Verbindungsbedingungen:

   Sei $Kontext(\hat{p}, \hat{m})$ der Graph der Elemente aus G, die nicht durch die Regel p verändert werden.

   - (a1) Für alle Netzwerkkanten $e : x \to y \in G$ mit Urbild in I gilt: der lokale Morphismus h_e^G zwischen den lokalen Graphen von x und y muss auch in $Kontext(p, m)$ definiert sein. Das heißt es muss gelten: $h_e^G(Context(p_x, m_{l(x)})) \subseteq Context(p_y, m_{l(y)}$.
   - (a2) Für alle Netzwerkkanten $e : x \to y \in G$, deren Zielknoten y ein Urbild in I haben und für die selber kein Urbild in L existiert, gilt: der lokale Morphismus h_e^G zwischen den lokalen Graphen von x und y muss auch in $Kontext(p, m)$ definiert sein. Das heißt, es muss gelten: $h_e^G(Context(p_x, m_{l(x)})) \subseteq Context(p_y, m_{l(y)}$.
   - (b) Für alle Netzwerkknoten $x \in G$, die ein Urbild in I und alle Kanten $e : m(l(x)) \to z \in G - m(L)$, die kein Urbild in L haben, gilt: die lokalen Regelmorphismen l_x und r_x müssen bijektiv sind.

4. Netzwerkbedingungen

   - (a1) Für alle Netzwerkknoten $x \in L - l(I)$, die kein Urbild in I haben, gilt, dass das Vorkommen m_x des lokalen Graphen von x bijektiv ist. Dies bedeutet, dass die lokalen Graph von x uns von $m(x)$ identisch sind.

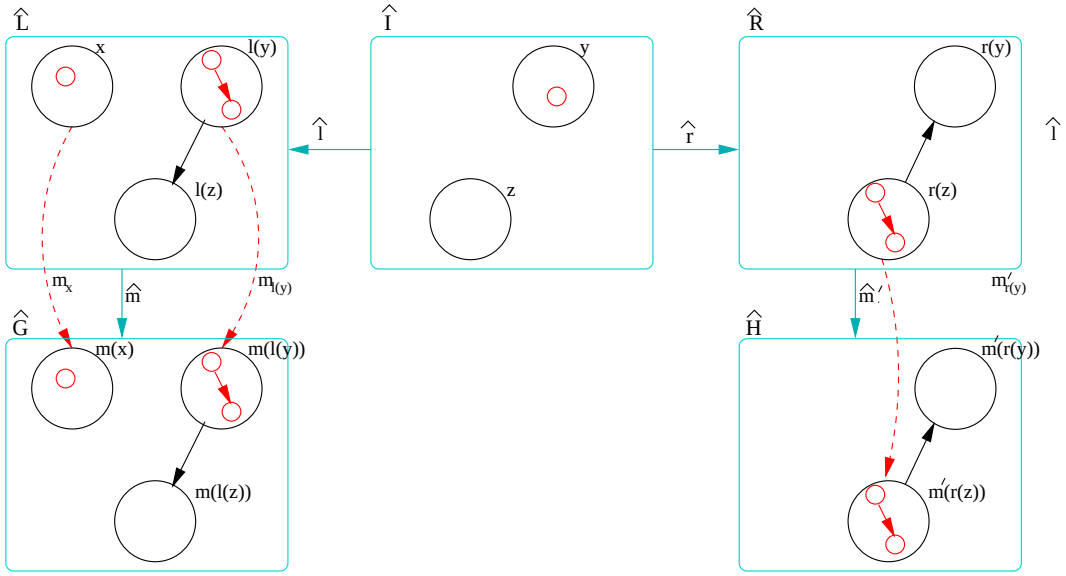

Abbildung 14: Beispiel für die Erfüllung der Netzwerkbedingung

- (a2) Für alle Netzwerkknoten $y \in I_V$ und alle Kanten $e : l(y) \to z$, die durch die Regel p gelöscht werden, ist das lokale Vorkommen $m_{l(y)}$ bijektiv.
- (b) Für alle Knoten y des Netzwerkgraphen I und alle Kanten $e : r(y) \to z \in R_E$, die kein Urbild in I_E haben, gilt, dass $m_{r(y)}$ bijektiv ist.

Beispiel 4.2.2 (Verbindungs- und Netzwerkbedingungen)

1. Als Verdeutlichung der Punkte (a1) und (a2) der Verbindungsbedingung dient die Abbildung 13. Die Punkte $m(l(x)), m(l(y)) \in G$ und die Kante $e1 : m(l(x)) \to m(l(y)) \in G$ haben ein Urbild im Netzwerkgraphen I. Daher muss nach Punkt (a1) der Verbindungsbedingung der lokale Graphmorphismus h^G_{e1} alle Knoten vom lokalen Kontextgraphen von $m(l(x))$ auf einen Knoten in $Context(p_y, m_y)$ abbilden. In diesem Fall ist h^G_e in $Kontext(p, m)$ definiert.

   Der Knoten $z \in G$ und die Kante $e2 : z \to m(l(y)) \in G$ haben kein Urbild in L und somit muss der Morphismus h^G_{e2} für den lokalen Graphen von z in den Kontextgraphen von p_y definiert sein.

   Sowohl h^G_{e1} als auch h^G_{e2} sind im Kontextgraphen definiert und erfüllen die Verbindungsbedingung (a1) bzw. (a2). Für die Kante $e3 : m(l(x)) \to z$ in $G - L$, deren Quelleknoten ein Urbild $x \in I$ hat, müssen l_x und m_x bijektiv, was in Beispiel 13 der Fall ist.

2. In Abbildung 14 ist ein Beispiel für die Netzwerkbedingungen zu sehen. Der Netzwerkknoten $x \in L$ hat kein Urbild in I. Diese Situation entspricht der in Punkt 1 der Netzwerkregel. Dies bedeutet, dass für den Knoten x das lokale Vorkommen $m_x : \tilde{L}(x) \to \tilde{G}(m(x))$ bijektiv sein muss. Da die lokalen Graphen von $x \in L$ und von $m(x) \in L$ jeweils nur aus einem lokalen Knoten bestehen, ist dies der Fall. Die Kante $e : l(y) \to l(z)$ hat kein Urbild in I,

das heißt, dass sie durch die Regel gelöscht wird. Um die Netzwerkbedingung zu erfüllen, muss das lokale Vorkommen $m_{l(y)}$ des Quellknoten von e bijektiv sein. Diese Bedingung ist erfüllt, denn der lokale Graph von $m(l(y)) \in G$ ist identisch zu dem von $l(y) \in L$. Gleiches gilt für die Quellknoten der Kante $e : r(z) \to r(y)$, die auf der rechten Regelseite neu hinzugefügt wurde.

4.3 Verteilter Kontextgraph

Wie schon erwähnt sichert die verteilte Klebe-Bedingung die Existenz eines verteilten Kontextgraphen. Dies bedeutet, dass eine Transformation entsprechend einer verteilten Regel $\hat{p}$ für ein Vorkommen $\hat{m}$ der linken Regelseite in dem zu verändernden Graphen $\hat{G}$ durchgeführt werden kann. Als ersten Schritt der Transformation wird der verteilte Kontextgraph von $\hat{p}$ und $\hat{m}$ konstruiert. Bei einem Kontextgraphen handelt es sich um eine Art von einfachem Pushout. Der Unterschied zu dem Pushout aus Kapitel 3 ist, dass einer der beiden n-injektiven Morphismen, aus denen er konstruiert wird, andersherum gerichtet ist. Diese Art von Pushout wird auch als Pushout-Komplement bezeichnet. Der Kontextgraph wird, wie der einfache Pushout, komponentenweise konstruiert.

Definition 4.3.1 (Konstruktion des verteilten Kontextgraph)
Sei $\hat{p} = \hat{L} \xleftarrow{\hat{l}} \hat{I} \xrightarrow{\hat{r}} \hat{R}$ eine verteilte Regel und $\hat{m} : \hat{L} \to \hat{G}$ ein Vorkommen für $\hat{p}$. Die Morphismen $\hat{c} : \hat{m} \circ \hat{l} : \hat{I} \to \hat{C}$ und $\hat{g} : \hat{C} \to \hat{G}$ werden als Inklusionen definiert. Das heißt, dass sie $\hat{I}$ bzw. $\hat{C}$ auf den Teilgraphen des jeweiligen Zielgraphen abbilden, der $\hat{I}$ bzw. $\hat{C}$ entspricht. Der verteilte Kontextgraph $\hat{C} = \langle C, \tilde{C} \rangle$ wird wie folgt konstruiert [HGGT]:

1. Konstruktion des Kontextgraph C für einfache Graphen auf der Netzwerkebene: ergibt sich aus der Netzwerkregel p und dem Vorkommen m. Dieser Kontextgraph repräsentiert den Teil des Netzwerkgraphen G, der von der Regel unverändert bleibt.

2. Die Abbildung $\tilde{C} : C \to AGr$, die jedem Netzwerkknoten den zugehörigen lokalen Graphen zuweist, wird für den verteilten Kontextgraphen konstruiert durch:

   - $\tilde{C}(i)$ = lokaler Kontextgraph der Regel p_y und des Vorkommens $m_{l(y)}$, falls der Netzwerkknoten $i \in C$ ein Urbild im Graphen I hat. Der lokale Kontextgraph wird entsprechend der Definition für einfache Kontextgraphen gebildet.
   - $\tilde{C}(i)$ = lokaler Graph $\tilde{G}(i)$, falls Knoten $i \in C$ nur in G vorkommt, aber nicht in L.

   Für die lokalen Morphismen $h_e^C : \tilde{C}(s(e)) \to \tilde{C}(t(e))$, die den Netzwerkkanten e aus C entsprechen gilt:

   - $h_e^C = g_{t(e)}^{-1} \circ h_{g(e)}^G \circ g_{s(e)}$. Dabei ist $h_{g(e)}^G$ der Morphismus zwischen $\tilde{G}(g(s(e)))$ und $\tilde{G}(g(t(e)))$ aus $\hat{G}$.

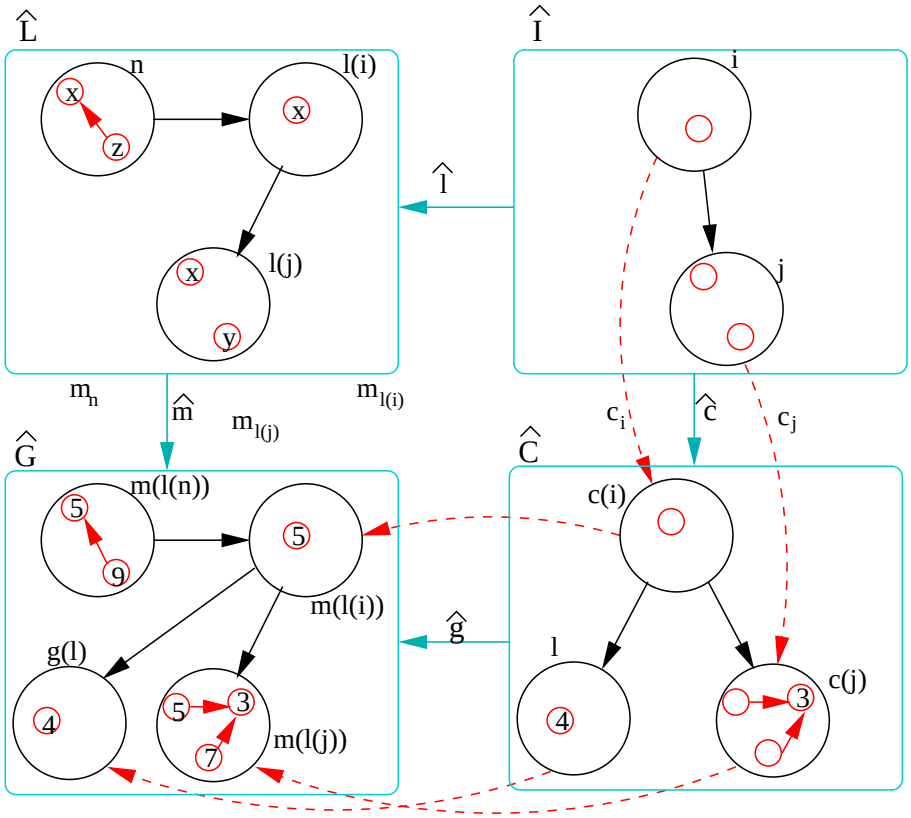

Abbildung 15: Beispiel der Konstruktion eines Kontextgraphen $\hat{C}$

Beispiel 4.3.2 (Verteilter Kontextgraph)
In Abbildung 15 ist die Konstruktion eines verteilten Kontextgraphen für ein Vorkommen $\hat{m}$ zu sehen. Dieses Vorkommen erfüllt sowohl auf Netzwerkebene für m Netzwerkregel p als auch auf der lokalen Ebene für m_i und m_j und die jeweilige Regel p_i und p_j die einfache Klebe-Bedingung. Da die beiden lokalen Regeln p_i und p_j keine lokalen Objekte löschen oder in einem lokalen Quellgraphen hinzufügen, ist auch die Verbindungsbedingung erfüllt. Das Gleiche gilt für die Netzwerkbedingung, da für den gelöschten Netzwerkknoten $n \in L$ und für den Knoten $l(i)$ die lokalen Vorkommen m_n und $m_{l(i)}$ bijektiv sind. Damit sind alle Bedingungen der verteilten Klebe-Bedingung für das Vorkommen $\hat{m}$ erfüllt.

Somit kann der Kontextgraph wie folgt konstruiert werden. Zuerst wird der Netzwerkgraph konstruiert, der aus den Knoten $c(i), c(j)$ und n besteht. Für die Knoten, die in C übernommen werden, gilt, dass sie entweder, wie n, nicht von der Regel betroffen sind, oder, wie $c(i)$ und $c(j)$, durch die Regel erhalten bleiben. Danach werden die lokalen Kontextgraphen der Knoten n, $c(i)$ und $c(j)$ erstellt. Da der Knoten n nicht in L existiert, ist sein lokaler Graph gleich dem aus $\hat{G}$. Für $\tilde{C}(i)$ wird der Graph aus $\hat{G}$ bis auf das Attibut $x = 5$ übernommen. Bei $\tilde{C}(j)$ bleibt das Attribut $x = 3$ für den Knoten erhalten, der nicht von der Regel betroffen ist.

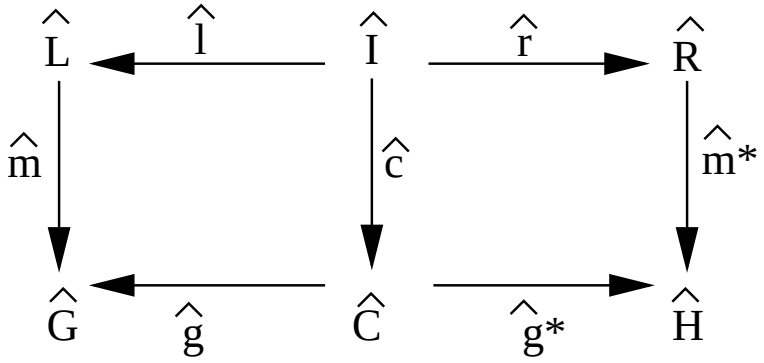

Abbildung 16: Double-Pushout, bildet die direkte Ableitung von $\hat{G}$ nach $\hat{H}$

5 Direkte Ableitung

Für eine verteilte Regel $\hat{p}$ und ein passendes Vorkommen $\hat{m}$ kann die Konstruktion des Kontextgraphen $\hat{C}$ und eines einfachen Pushout zu einem doppelten Pushout zusammenfügt werden. Der Aufbau dieses doppelten Pushouts ist in Abbildung 16 beschrieben. Dabei bilden die linke Regelseite, das Vorkommen $\hat{m}$ und der daraus resultierende Kontextgraph die linke Seite des doppelten Pushouts. Auf der rechten Seite wird aus den Graphen $\hat{R}$ und $\hat{C}$ der einfache Pushout $\hat{H}$ konstruiert (vergleiche Abbildung 16 und 17). $\hat{H}$ entspricht dem verteilten Graphen, der sich durch die Transformation entsprechend der Regel $\hat{p}$ aus $\hat{G}$ ergibt. Das Ergebnis des doppelten Pushouts wird auch als direkte Ableitung von $\hat{G}$ bezeichnet.

Definition 5.0.3 (Direkte Ableitung)

Sei $\hat{p} = \hat{L} \xleftarrow{\hat{l}} \hat{I} \xrightarrow{\hat{r}} \hat{R}$ eine verteilte Regel und $\hat{m} : \hat{L} \to \hat{G}$ ein Vorkommen dafür. Dann ist

$$\hat{G} \overset{\hat{p},\hat{m}}{\Longrightarrow} \hat{H}$$

eine verteilte attributierte Graphtransformation, gegeben durch eine direkte Ableitung entsprechend des doppelten Pushouts aus Abbildung 16 [HGGT].

Beispiel 5.0.4 (Direkte Ableitung)

In Abbildung 17 ist ein Beispiel für einen doppelten Pushout dargestellt. Der Graph $\hat{H}$ ergibt sich aus der direkten Ableitung von $\hat{G}$ durch die Anwendung der verteilten Regel $\hat{p} = \hat{L} \xleftarrow{\hat{l}} \hat{I} \xrightarrow{\hat{r}} \hat{R}$. Dabei bewirkt $\hat{p}$, dass der Netzwerkknoten $n' \in \hat{G}$ gelöscht und der Knoten $k' \in \hat{H}$ hinzugefügt wird. Auch auf lokaler Ebene finden derartige Transformationen statt. Außerdem werden die Attribute verändert, wie es am Beispiel des lokalen Graphen des Knoten j zu erkennen ist.

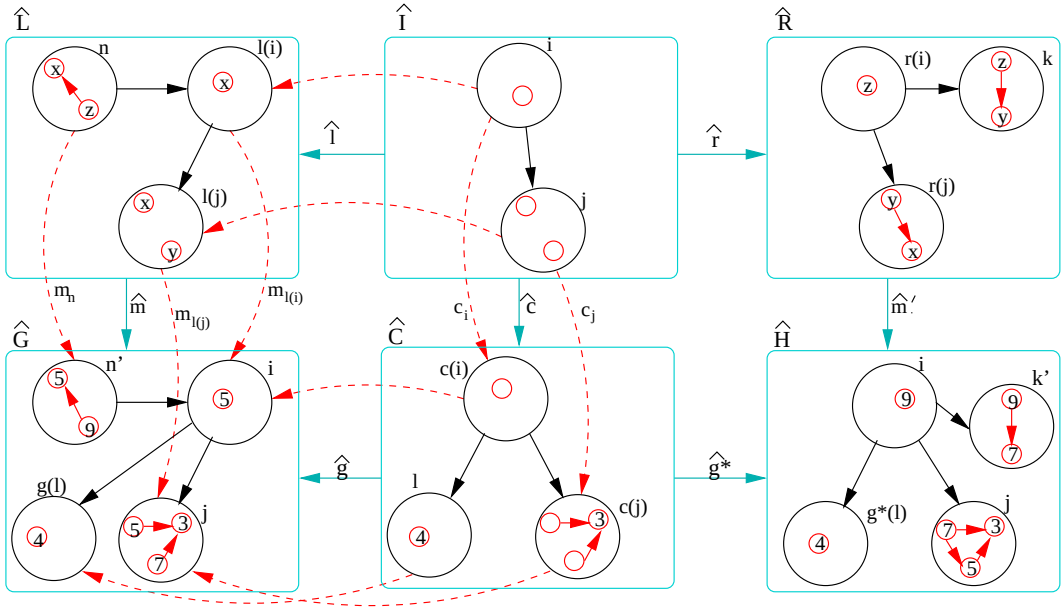

Abbildung 17: Beispiel einer Graphtransformation durch Konstruktion eines doppelten Pushouts

6 Zusammenfassung und Ausblick

Ziel dieser Arbeit war es, eine graphische Notation für Transformationen auf verteilten Graphen zu beschreiben. Dies geschah vor dem Hintergrund, das verteilte Graphen ein wichtiges Hilfsmittel zur Organisation komplexer Systeme sind. Als Beispiel für ein verteiltes System wurde ein kleines Banksystem vorgestellt, das aus einer Bank und zwei Kunden besteht (vgl. Abbildung 1). Zur Beschreibung der Beziehungen zwischen den Kunden und der Bank wurde eine graphische Notation verwendet, die auf dem Prinzip der verteilten Graphen basiert. Die Vorgänge, die in einem solchen System stattfinden können, werden bei diesem Ansatz mit Hilfe von Transformationen auf verteilten Graphen definiert.

Ein verteilter Graph besteht aus einer Netzwerkebene und einer lokalen Ebene. Um die lokale Ebene darzustellen, wurden attributierte Graphen eingeführt. Aufbauend auf den attributierten Graphen konnten verteilte Graphen definiert werden. Diese bestehen aus einem L-markierten Netzwerkgraphen und einer Abbildung, die jedem Netzwerkknoten einen lokalen attributierten Graphen zuweist. Durch die Definition von Regeln können auf den Graphen Transformationen beschrieben werden. Als Lösungsansatz für die Durchführung von Transformationen auf verteilten Graphen wurde ein doppelter Pushout wie folgt hergeleitet. Die linke Seite des Pushouts wurde durch eine verteilte Regel und ein Vorkommen der linken Regelseite in dem zu transformierenden Graphen definiert. Den mittleren Teil des Pushouts bildet der Kontextgraph, der alle Elemente beinhaltet, die nicht durch die Regel verändert werden. Die rechte Seite entspricht der Konstruktion des einfachen Pushouts der rechten Regelseite und des Kontextgraphen. Sie liefert den transformierten verteilten Graphen.

Bei den Beispielen, die in dieser Ausarbeitung verwendet wurden, handelt es

sich um kleine verteilte Graphen, die nicht der Struktur eines komplexen verteilten Systems entsprechen. Dennoch kann der hier eingeführte Ansatz auch bei großen Systemen als Grundlage für die Beschreibung der Strukturen und der Transformationen verwendet werden. Im Rahmen der Anforderungen, die an das jeweilige verteilte System gestellt werden, sind Erweiterungen der Notation möglich. Eine wichtige Anforderungen könnte zum Beispiel sein, dass eine Transformation nicht in jedem Fall und beliebig oft durchgeführt werden kann. Um die Anwendung von verteilten Regeln zu beschränken, werden weitere Bedingungen bzw. verbotene Kontexte an den verteilten Graphen gestellt. Eine solche Voraussetzung, die erfüllt sein muss, kann zum Beispiel sein, dass bei dem zu Anfang erwähnten Bankbeispiel ein Kunde nur zwei Konten haben darf. Dies bedeutet, dass eine Regel, die die Neueinrichtung eines Kontos für einen Kunden beschreibt, nicht angewendet werden darf, wenn dieser Kunde schon zwei Konten besitzt.

Literatur

[HGGT] H.Ehrig, H.-J. Kreewski, U.Montanari, G.Rozenberg. *Handbook of Graph Grammers and Computing by Graph Transformation Volume 3, Kapitel 5, World Sientific Verlag, 1997.*