Seminar

Graph-Grammatiken

Lehrstuhl für Informatik III
RWTH Aachen
Sommersemester 2004

Beschreibung von Prozessen mittels High-Level-Ersetzung

Maik Schwefer

Inhaltsverzeichnis

1 Motivation und Grundbegriffe

In dieser Arbeit wird das Thema „Beschreibung von Prozessen mittels High-Level-Ersetzung“ vorgestellt. Hierzu wird zunächst der Aufbau der Arbeit beschrieben und eine Struktur namens Komma-Kategorie entwickelt, mit welcher später Graphen, Petrinetze und Zustandscharts definiert werden.

1.1 Motivation und Aufbau der Arbeit

Graphen und Graphtransformationen sind suggestive Mittel, um Systeme zu beschreiben. Graphen repräsentieren die Struktur des Systems, während die Graphersetzungsregeln das dynamische Verhalten festlegen. Mit Graphtransformations-Systemen können dynamische Systeme modelliert werden, deren Struktur verändert wird, zum Beispiel durch Löschen oder Hinzufügen von Prozessen. In dieser Arbeit werden verschiedene Systeme mittels High-Level-Ersetzung beschrieben.

In dem ersten Kapitel dieser Arbeit werden wichtige Grundbegriffe der Graphentheorie, wie Ableitbarkeit, Parallele Unabhängigkeit und HLR-Bedingungen (High Level Replacement Conditions) definiert. Im zweiten Kapitel wird der Komma-Kategorie-Ansatz vorgestellt. Dieser dient als Grundlage für die folgenden Kapitel. Anschließend wird auf spezielle Typen von Graphen und Graphtransformations-Systemen eingegangen, welche mit Hilfe des Komma-Kategorie-Ansatzes definiert werden. Das dritte Kapitel beschäftigt sich mit Graphen und Markierungstechniken, unter anderem mit Hypergraphen, markierten Graphen und der Technik des Zwei-Stufen-Markierens. Petrinetze werden im vierten Kapitel mittels zweiteiliger Graphen eingeführt, wobei besonders auf die Bedingungs-Ereignisnetze und die Stellen-Transitionsnetze eingegangen wird. Im fünften Kapitel werden Zustandscharts beschrieben, anschließend werden einige der vorgestellten Modelle kombiniert, um Zustandscharts als Komma-Kategorie zu definieren. Kapitel 6 gibt einen kurzen Überblick über die vorhergehenden Kapitel und zeigt, welche Möglichkeiten der Komma-Kategorie-Ansatz noch bietet.

1.2 Grundbegriffe

In diesem Abschnitt werden wichtige Grundbegriffe der Graphentheorie eingeführt. Dazu wird eine beliebige Kategorie vorausgesetzt, zunächst ohne direkten Bezug zu Graphen. Die Annahme einer beliebigen Kategorie vereinfacht die Darstellung, ohne auf ein bestimmtes System festgelegt zu sein, denn die Grundbegriffe, die in diesem Kapitel vorgestellt werden, sollen in den nächsten Kapiteln auf verschiedene Graphstrukturen angewendet werden. Kategorien bestehen aus Objekten und Morphismen, bei Graphen wären die Objekte die Knoten und die Morphismen die Kanten.

1.2.1 Ableitbarkeit

In diesem Abschnitt werden zuerst Produktionen definiert, welche benötigt werden, um anschließend Ableitbarkeit zwischen zwei Objekten zu definieren.

Definition 1. Produktion: $p = (\ B^l \xleftarrow{p^l} K \xrightarrow{p^r} B^r\ )$.

Eine Produktion ist ein Morphismenpaar (p^l, p^r) mit $\mathrm{dom}(p^l) = \mathrm{dom}(p^r)$. Das bedeutet, dass beide Morphismen den selben Ursprung K haben müssen, um zusammen eine Produktion zu bilden. K wird das „Klebeobjekt“ genannt, und B^l (bzw. B^r) die linke (bzw. rechte) Seite der Produktion.

Sei p eine Produktion mit $p =(\ B^l \xleftarrow{p^l} K \xrightarrow{p^r} B^r\ )$ in einer beliebigen Kategorie.

Dann heißt G^r aus G^l mittels p ableitbar ($G^l \Rightarrow^p G^r$) genau dann, wenn ein g: $K \to C$ existiert, so dass gilt:

$$\begin{array}{ccccc} B^l & \xleftarrow{p^l} & K & \xrightarrow{p^r} & B^r \\ {\scriptstyle g^l}\downarrow & PO & \downarrow{\scriptstyle g} & PO & \downarrow{\scriptstyle g^r} \\ G^l & \xleftarrow[p^{-l}]{} & C & \xrightarrow[p^{-r}]{} & G^r \end{array}$$

□

B, K, C und G sind Objekte der gewählten Kategorie, zum Beispiel Graphen oder Mengen. B^l ist das Objekt, welches mit Hilfe der Produktion p durch B^r ersetzt wird. p und g sind Morphismen, die auf den Objekten operieren. Der linke und rechte Teil des Systems müssen Pushouts sein. Bei einem Pushout werden zwei verschiedene Objekte, die das gleiche Objekt als Ursprung haben, mit Hilfe von Morphismen zu einem Objekt zusammengeführt. Die linke Seite in obiger Abbildung ist beispielsweise ein Pushout, wenn B^l und C, welche aus K durch die Anwendung von p^l und g entstanden sind, durch weitere Morphismen zu G^l zusammen geführt werden können. Wenn p und g gegeben sind, dann können G^l und G^r eindeutig bestimmt werden. Dies wird an folgenden Beispiel verdeutlicht. Seien p^l und g injektiv.

$$\begin{array}{ccccc} B^l = \{1,2,3\} & \xleftarrow{p^l} & K = \{1,2\} & \xrightarrow{p^r} & B^r\{1,2,6,7\} \\ & & \downarrow{\scriptstyle g} & & \\ & & C = \{1,2,4,5\} & & \end{array}$$

Dann können G^l und G^r bestimmt werden. Nach Anwenden von g auf B^l und B^r entsteht folgendes Diagramm:

$$\begin{array}{ccccc} B^l = \{1,2,3\} & \xleftarrow{p^l} & K = \{1,2\} & \xrightarrow{p^r} & B^r = \{1,2,6,7\} \\ {\scriptstyle g^l}\downarrow & PO & \downarrow{\scriptstyle g} & PO & \downarrow{\scriptstyle g^r} \\ G^l = \{1,2,3,4,5\} & \xleftarrow[p^{-l}]{} & C = \{1,2,4,5\} & \xrightarrow[p^{-r}]{} & G^r = \{1,2,4,5,6,7\} \end{array}$$

Somit ist G^r ={1,2,4,5,6,7} mittels der Produktion p aus G^l ={1,2,3,4,5} ableitbar. Auf der linken Seite sind die Mengen B^l und C gegeben, welche durch p^l und g aus K hervorgegangen sind. Die linke Seite ist ein Pushout, weil B^l und C durch die Morphismen g und p zu dem Objekt G^l zusammengeführt werden können. Das gleiche gilt für die rechte Seite.

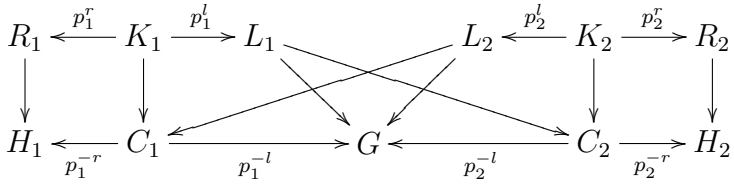
1.2.2 Parallele Unabhängigkeit

Eine wichtige Eigenschaft von Ersetzungssystemen ist die Parallele Unabängigkeit. Wenn eine Ableitungssequenz durchlaufen wird, das heißt, wenn mehrere Ableitungsschritte hintereinander ausgeführt werden, stellt sich die Frage, ob sich das Ergebnis ändert, wenn die Reihenfolge der Ableitungsschritte verändert wird. Es ist auch möglich, dass Elemente, welche die zweite Produktion benötigt, nicht mehr verfügbar sind, nachdem die erste Produktion ausgeführt wurde, und somit die zweite Produktion nicht mehr angewendet werden kann.

Wenn zwei Ableitungsschritte in beliebiger Reihenfolge ausgeführt werden können, werden sie parallel unabhängig genannt. Dann gilt, dass die eine Produktion keine

Elemente löscht oder verändert, welche die andere Produktion benötigt.

Definition 2. Zwei Ableitungsschritte $G \Rightarrow_1^p H_1$ und $G \Rightarrow_2^p H_2$ sind parallel unabhängig genau dann, wenn es Morphismen $L_1 \to C_2$ und $L_2 \to C_1$ gibt, so dass $L_1 \to C_2 \to G = L_1 \to G$ und $L_2 \to C_1 \to G = L_2 \to G$.



Das bedeutet, dass Elemente, welche von L_1 nach G abgebildet werden, nicht durch die vorherige Anwendung von p_2 entfernt oder verändert worden sein dürfen. Wenn zum Beispiel in K_2 die Elemente a und b vorkommen, und diese durch p_2 auf die Elemente a und c in G abgebildet werden, darf der Morphismus von L_1 nach G nicht auf b abbilden. Denn das Element b wäre durch die Produktion p_2 entfernt worden, wenn p_2 vor p_1 ausgeführt würde. In Abbildung 1 wird ein Beispiel für parallele Unabhängigkeit gegeben. □

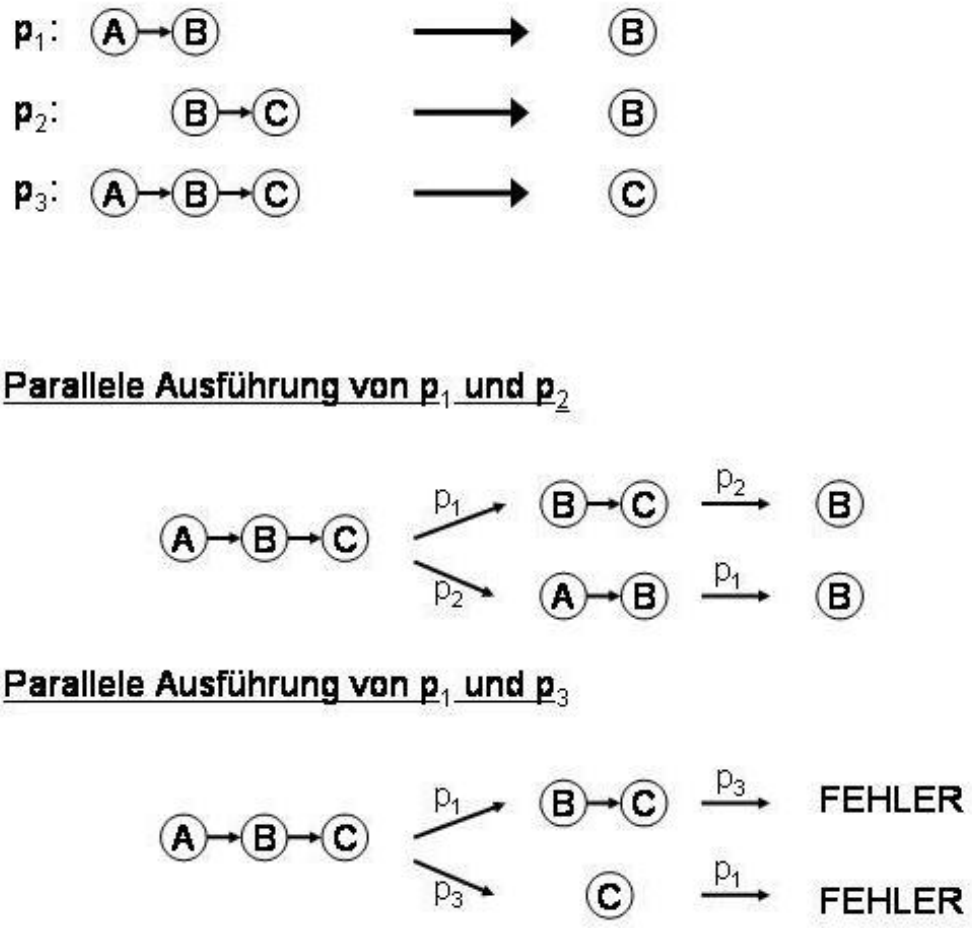


Abbildung 1: Parallele Unabhängigkeit

In der Abbildung sind drei Produktionen p_1, p_2 und p_3 gegeben. Die erste entfernt aus einem Graphen den Knoten A, die zweite den Knoten C und die dritte die Knoten A und B. Die beiden Produktionen p_1 und p_2 sind parallel unabhängig, weil sie ohne anderes Ergebnis hintereinander ausgeführt werden können. Falls jedoch die Produktionen p_1 und p_3 nacheinander ausgeführt werden, tritt jeweils nach dem ersten Schritt ein Problem auf. Zum Beispiel ist nach dem Ausführen von p_1 der Knoten A nicht mehr vorhanden, dieser wird jedoch für die Ausführung von p_3 benötigt. p_1 und p_3 sind also nicht parallel unabhängig.

1.2.3 HLR1-Kategorien

Kategorien bestehen aus Objekten (A,B,C...) und Morphismen ($f : A \to B$,...) zwischen diesen Objekten (Relationen, Abbildungen oder ähnliches). Dabei ist zu beachten, dass die Objekte und die Verbindungen jeweils Mengen sind. Die Kategorientheorie betrachtet nur die Begriffe „Objekt“ und „Morphismus“, sie abstrahiert also von der konkreten Struktur verschiedener Theorien (Mengentheorie, Graphentheorie,...). Deswegen ist es möglich, verschiedene Theorien, welche jeweils aus Objekten und Morphismen bestehen, in einer Kategorie zu behandeln. Es muss die Assoziativität zwischen den Verbindungen der Objekte gelten und zu jedem Objekt muss die Identitätsabbildung existieren. Desweiteren muss für alle $f : A \to B$ und $g : B \to C$ auch die Komposition $g \cdot f : A \to C$ gelten.

Ein einfaches Beispiel für eine Kategorie ist *Set*. Die Objekte sind Mengen, $Mor_{Set}(A, B)$ ist die Menge aller Abbildungen von A nach B und die Komposition ist die Hintereinanderausführung der Abbildungen. Eine weitere Kategorie ist *Rel* welche anstelle von Abbildungen auf Relationen zwischen den Objekten basiert.

Eine High-Level-Replacement-Kategorie, kurz HLR-Kategorie, ist eine Kategorie, deren Morphismen bestimmten Bedingungen genügen müssen. Es gibt verschieden starke HRL-Bedingungen, welche zum Beispiel die HRL-Kategorien $HRL0.5$ und $HRL1$ definieren. Eine Kategorie, die keine extra Bedingungen erfüllt, wird $HRL0$-Kategorie genannt.

Definition 3. Sei M die Menge der Morphismen der HLR-Kategorie C. C ist eine $HLR0.5$-Kategorie, wenn die zwei folgenden Bedingungen erfüllt sind:

(a) Sind $f, g \in M$, dann existiert das Pushout-Diagramm $q \cdot f = p \cdot g$ für beliebige $p, q \in M$.

(b) Wenn zwei Morphismen f und g Elemente von M sind, ist auch $f + g$ Element von M.

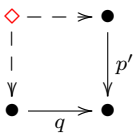
□

Definition 4. Eine $HLR1$-Kategorie muss zusätzlich zu der $HLR0.5$-Kategorie noch zwei weitere Bedingungen erfüllen:

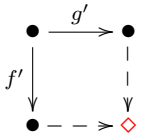
(a) Sind zwei beliebige Morphismen $p', q \in M$, so existiert das Pullback-Diagramm $p' \cdot g' = q \cdot f'$ mit $f', g' \in M$. Ein Pullback ist sozusagen das Gegenteil von einem Pushout, weil nicht zwei Objekte zusammengeführt werden, sondern zu zwei Objekten das Objekt gesucht wird, aus welchem diese beiden Objekte entstanden sind. Dies wird durch die folgende Abbildung verdeutlicht. Das linke Diagramm zeigt die Ausgangssituation bei einem Pullback, das rechte

bei einem Pushout. Das Rechteck ist jeweils das Objekt, welches durch den Pullback bzw. Pushout konstruiert werden soll.

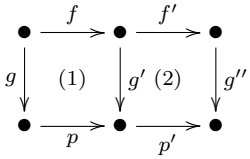
Pullback



Pushout



(b) Seien in dem folgendem Diagramm $f, f', g, g', g'', p, p' \in$ M. Es gibt zwei kommutative Diagramme (1) mit $g' \cdot f = p \cdot g$ und (2) mit $g'' \cdot f' = p' \cdot g'$. Wenn nur die äußeren Morphismen betrachtet werden, bilden $(1 + 2)$ das kommutative Diagramm $g'' \cdot (f' \cdot f) = (p' \cdot p) \cdot g$. Ist nun $(1 + 2)$ ein Pushout und (2) ein Pullback, dann sind (1) und (2) jeweils eigenständige Pushout-Diagramme.



□

Desweiteren gilt, dass wenn (1) und (2) Pushouts sind, auch $(1 + 2)$ ein Pushout-Diagramm ist. Die Morphismen f und f' können zu f'' verknüpft werden, somit lassen dich f'' und g durch g'' und p'' zu einem Pushout-Konstrukt erweitern, wobei p'' die Verknüpfung von p und p' ist. Umgekehrt gilt, wenn $(1 + 2)$ und (1) Pushout-Diagramme sind, dann ist (2) auch eines.

Typische $HLR1$-Kategorien sind Set und $Graph$ mit injektiven Morphismen. Set und $Graph$ mit nicht-injektiven Morphismen erfüllen nicht alle $HLR1$-Bedingungen, weil ein nicht-injektiver Morphismus mehrere Elemente auf ein Element abbildet, und so einige Elemente entfernt werden. Dieses führt dazu, dass nicht alle Morphismen parallel unabhängig sind, und nicht immer ein Pushout oder Pullback konstruiert werden kann.

1.2.4 Paralleles Unabhängigkeits-Theorem

Dem Parallelen Unabhängigkeits-Theorem liegt die Definition der Parallele Unabhängigkeit (Abschnitt 1.2.2) zugrunde. Das Theorem besagt, dass parallel unabhängige Ableitungschritte vertauscht werden können, ohne dabei das Ergebnis zu verändern. Wenn in einer $HLR1$-Kategorie zwei Ableitungsschritte $G \Rightarrow_{p_1} H_1$ und $G \Rightarrow_{p_2} H_2$ parallel unabhängig sind, dann existiert ein G', so dass $H_1 \Rightarrow_{p_2} G'$ und $H_2 \Rightarrow_{p_1} G'$.

2 Komma-Kategorie-Ansatz

Komma-Kategorien sind Strukturen, welche auf Kategorien und Morphismen zwischen diesen Kategorien aufbauen. Ein Morphismus zwischen zwei Kategorien heißt Funktor. In der Kategorientheorie wurde der Begriff Funktor 1972 von Eilenberg und Mac Lane eingeführt, um eine bestimmte Art von Abbildungen zwischen Kategorien zu bezeichnen. Der englische Begriff „functor“ wird in Wörterbüchern als „something that performs a function or an operation“ [2] erklärt. Ein Funktor $F : C \to D$ ist

sozusagen eine Abbildung von einer Kategorie in die andere. Er besteht aus zwei Abbildungen. Eine Abbildung bildet die Objekte von C auf Objekte in D ab, die andere bildet die Morphismen von C auf Morphismen in D ab. Desweiteren erhält ein Funktor die Komposition der Morphismen der Kategorien. Falls also die Komposition $f \cdot g$ zweier Morphismen f und g in C enthalten ist, existiert auch $F(f) \cdot F(g)$ in D. Funktoren sind Struktur-erhaltend. Das bedeutet, dass die Morphismen und Kompositionen zwischen Objekten durch Anwenden eines Funktors nicht verändert werden. Wenn zum Beispiel zwei Objekte durch einen bestimmten Morphismus verbunden sind, besteht diese Morphismus auch nach der Abbildung durch einen Funktor in eine andere Kategorie. Ein einfaches Beispiel ist die Identität 1_C, die jedes Objekt und jeden Morphismus auf sich selbst abbildet.

Komma-Kategorien bestehen, genauso wie Kategorien, aus Objekten und Morphismen zwischen diesen Objekten. Die Objekte einer Komma-Kategorie bestehen aus je zwei Objekten zweier Kategorien und einer Funktion zwischen diesen beiden Objekten. Ein Objekt der Komma-Kategorie der Graphen besteht zum Beispiel aus einer Kantenmenge, welcher über eine Funktion eine Knotenmenge zugewiesen wird. Die Kantenmenge ist aus der anderen Kategorie, welche Kantenmengen als Objekte enthält. Die Knotenmenge ist aus einer Kategorie mit Knotenmengen als Objekten. Die ausgewählten Kanten spannen zusammen mit den ihnen zugewiesenen Knoten einen Graphen auf. In diesem Kapitel wird zunächst auf die grundlegende Definition einer Komma-Kategorie eingegangen. In den nächsten Kapiteln werden dann Graphen, Petrinetze und Zustandscharts als Komma-Kategorie definiert.

2.1 Definition einer Komma-Kategorie

Seien $F : C \to E$ und $G : D \to E$ Funktoren. Die Komma-Kategorie (F, G) ist die Kategorie mit Tripeln (A, f, B) als Objekten, wobei $A \in C$, $B \in D$ und $f : F(A) \to G(B)$ eine Abbildung von der Kategorie E auf sich selbst ist. Die Morphismen der Komma-Kategorie bestehen aus Paaren $(g, h) : (A, f, B) \to (A', f', B')$ mit $(g : A \to A', h : B \to B')$ in $C \times D$, so dass das folgende Diagramm kommutativ ist.

$$\begin{array}{ccc} F(A) & \xrightarrow{f} & G(B) \\ {\scriptstyle F(g)}\downarrow & & \downarrow{\scriptstyle G(h)} \\ F(A') & \xrightarrow[f']{} & G(B') \end{array}$$

Objekte einer Komma-Kategorie bestehen aus drei Elementen A, B und f. A ist ein Objekt einer Kategorie C, welches über den Funktor F in die Kategorie E abgebildet wird. B ist ein Objekt der Kategorie D, durch den Funktor G nach E abgebildet, und die Funktion f verbindet diese beiden Objekte in der Kategorie E. In der Abbildung sind die nach E abgebildeten Objekte F(A) und G(B) (bzw. F(A') und G(B')) zu sehen, welche durch die Funktion f (bzw. g) verbunden sind. Morphismen zwischen Komma-Kategorie-Objekten (A, B, f) und (A', B', f') sind Tupel (g, h). Der Morphismus g bildet A nach A' ab, genauso wie B durch h nach B' abgebildet wird. In der Abbildung ist F(g) anstelle von g angegeben, weil F(g) die Funktion g in E repräsentiert. F(g) bildet F(A) nach F(A') ab. In den nächsten Kapiteln werden verschiedene Typen von Graphen, Petrinetze und Zustandscharts als Komma-Kategorien definiert.

2.2 Eigenschaften der Komma-Kategorie

In diesem Abschnitt werden einige Eigenschaften der Komma-Kategorie aufgeführt, unter anderem in Verbindung mit $HLR1$-Kategorien. Die Konstruktion von Kategorien als Komma-Kategorie bringt viele Vorteile mit sich. Die Komma-Kategorien basieren auf Kategorien, welche in Kapitel 1 definiert wurden, deren Eigenschaften übernommen werden. Wenn zum Beispiel eine Komma-Kategorie auf einer Kategorie basiert, deren paralleles Verhalten bekannt ist, kann dieses Verhalten aufgrund der Eigenschaften, welche in diesem Abschnitt vorgestellt werden, sofort auf die Komma-Kategorie transferiert werden.

Konstrukte wie Pushouts oder Pullbacks heißen Kolimits. Eine Kategorie ist kovollständig, wenn sie ein Kolimit besitzt. Zum Beispiel ist eine Kategorie kovollständig, wenn es immer möglich ist, ein Pushout zu konstruieren. Funktoren die Kolimits einer Kategorie auf Kolimits einer anderen Kategorie abbilden, heißen kokontinuierliche (oder engl. „kolimit preserving“) Funktoren. Die Komma-Kategorie (L,R) mit $L : A \to C$ und $R : B \to C$ ist kovollständig, wenn die beiden Funktoren L und R kokontinuierlich und A und B kovollständig sind. Falls also A und B Kolimits besitzen, werden diese durch die kokontinuierlichen Funktoren „kolimit-erhaltend“ nach C abgebildet, also ist (L,R) auch kovollständig. Desweiteren gilt, dass wenn A und B die $HLR1$-Bedingungen erfüllen und L und R kokontinuierlich sind, die Komma-Kategorie (L,R) eine $HLR1$-Kategorie ist. Außerdem gilt die parallele Unabhängigkeit in (L,R), wenn alle Morphismen in A und B parallel unabhängig sind, und L und R kokontinuierlich sind.

3 Graphen und Markierte Graphen

In diesem Kapitel wird der Nutzen des Komma-Kategorie-Ansatzes anhand von Graphen und Markierungstechniken dargestellt. Es werden Graphen, Hypergraphen und markierte Graphen als Komma-Kategorien definiert.

3.1 Graphen

In diesem Abschnitt werden Graphen als Komma-Kategorie definiert.

Definition 5. Die Komma-Kategorie *Graph* benutzt die zwei Funktoren Id und X. Id ist der Identitäts-Funktor und ist definiert als Id: $Set \to Set$. X: $Set \to Set$ wird Diagonal-Funktor genannt, und ist durch $\mathrm{X}(S) := S \times S$ und $\mathrm{X}(f)(a,b) := (f(a), f(b))$ definiert. *Graph* ist die Komma-Kategorie der Graphen.

□

Ein Graph ist im Komma-Kategorie-Ansatz ein 3-Tupel $G = (E, V, c)$, wobei E die Menge der Kanten, V die Menge der Knoten und $c : E \to V \times V$ eine Funktion ist, welche jede Kante mit einem Start- und Zielknoten verbindet. Ein Graphmorphismus besteht aus zwei Abbildungen $g_E : E_1 \to E_2$ und $g_V : V_1 \to V_2$, g_E bildet die Kanten und g_V die Knoten aufeinander ab. Die Komma-Kategorie *Graph* sieht in der graphischen Darstellung folgendermaßen aus:

$$\begin{array}{ccc} Id(E_1) & \xrightarrow{c_1} & X(V_1) \\ {\scriptstyle Id(g_E)}\downarrow & & \downarrow{\scriptstyle X(g_V)} \\ Id(E_2) & \xrightarrow[c_2]{} & X(V_2) \end{array} \qquad \begin{array}{ccc} E_1 & \xrightarrow{c_1} & V_1 \times V_1 \\ {\scriptstyle g_E}\downarrow & & \downarrow{\scriptstyle (g_V, g_V)} \\ E_2 & \xrightarrow[c_2]{} & V_2 \times V_2 \end{array}$$

Das linke Diagramm zeigt den Graphmorphismus in seiner formal definierten Darstellung, wobei die Funktoren Id und X noch nicht auf die Kanten und Knoten an-

gewendet worden sind. c_1 und c_2 sind die Verbindungen zwischen den Knoten und Kanten des jeweiligen Graphen, und g_E und g_V die Morphismen, welche die Kanten bzw. Knoten vom ersten Graphen auf die Kanten (bzw. Knoten) des zweiten Graphen abbilden. Auf der rechten Seite wurden die Funktoren Id und X ausgeführt. Jeder Kante E werden zwei Knoten $V \times V$ zugewiesen.

Im nächsten Abschnitt werden Hypergraphen untersucht, welche im Komma-Kategorie-Ansatz ähnlich definiert werden, wie die in diesem Abschnitt vorgestellten Graphen.

3.2 Hypergraphen

Bei Hypergraphen können Kanten, im Gegensatz zu Standard-Graphen, mehr als zwei Knoten verbinden. Diese Kanten werden Hpyerkanten genannt. Hypergraphen eignen sich gut für Berechnungen und Schätzungen. Die Kanten stellen dann die Operationen dar, die Knoten die Parameter oder Resultate. Es werden im folgenden Graphen der Komma-Kategorie $Hgraph$ vorgestellt, welche im Gegensatz zu $STh graph$ nicht zwischen Quellknoten (Parametern) und Zielknoten (Resultaten) unterscheiden.

Definition 6. Der Funktor $Q : Set \to Set$ ist definiert durch $Q(S) := S^+$ und $Q(f)(w) := f^+(w)$ (f^+ ist definiert durch $f^+(\varepsilon) = \varepsilon$ und $f^+(aw) = f(a)f^+(w)$ für alle $a \in S$, $w \in S^+$). Dieser Funktor bildet Knoten aus einer Knotenmenge auf ein Wort von Knoten ab. Zum Beispiel definiert ein Wort $w = ABCD$ vier Knoten A, B, C und D, welche durch die Funktion $f(w)$ der Reihe nach ausgelesen werden können. Die Komma-Kategorie der Hypergraphen $Hgraph$ besteht aus den Funktoren Id und Q. Ein Hypergraph $H = (E, V, c)$ mit $Q : E \to V^+$ sieht in der grafischen Darstellung folgendermaßen aus:

$$\begin{array}{ccc} Id(E_1) & \xrightarrow{c_1} & Q(V_1) \\ {\scriptstyle Id(g_E)}\downarrow & & \downarrow{\scriptstyle Q(g_V)} \\ Id(E_2) & \xrightarrow[c_2]{} & Q(V_2) \end{array} \qquad \begin{array}{ccc} E_1 & \xrightarrow{c_1} & V_1^+ \\ {\scriptstyle g_E}\downarrow & & \downarrow{\scriptstyle g_V^+} \\ E_2 & \xrightarrow[c_2]{} & V_2^+ \end{array}$$

□

Der Unterschied zu Standard-Graphen ist, dass jede Kante mit mehr als zwei Knoten verbunden sein kann. Die Funktion c_1 verbindet zum Beispiel Kanten aus E_1 mit mehreren Knoten aus V_1, welche als ein Wort zusammengefasst sind.

3.3 Auswahlfunktoren

In Komma-Kategorien können verschiedene Markierungstechniken in einer Struktur zusammengefasst werden. Im nächsten Abschnitt wird ein Markierungsmechanismus vorgestellt, der von einem Graphen die Knoten, die Kanten oder beide Mengen markiert. Hierfür werden Auswahlfunktoren benötigt. Auswahlfunktoren sind spezielle Funktoren, die aus einem Graphen entweder Knoten, Kanten oder Knoten und Kanten auswählen. Es gibt die Auswahlfunktoren W_E, W_V und W_{EV}, wobei $W_E(G)$ die Kanten, $W_V(G)$ die Knoten und $W_{EV}(G)$ die Kanten und Knoten von einem Graphen G auswählt.

Die Auswahlfunktoren $W_E, W_V, W_{EV} : G \to Set$ sind folgendermaßen definiert:

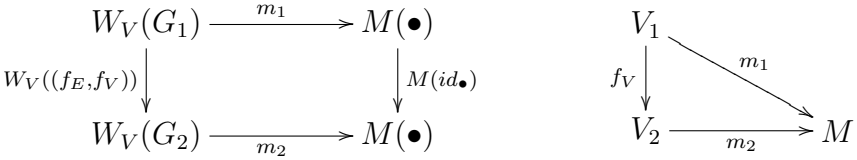
$$W_E((E, V, c)) := E \qquad W_E((f_E, f_V)) := f_E$$

$$W_V((E,V,c)) := V \qquad W_V((f_E,f_V)) := f_V$$
$$W_{EV}((E,V,c)) := E+V \qquad W_{EV}((f_E,f_V)) := f_E+f_V$$

$G = (E,V,c)$ ist ein beliebiger Graph und (f_E, f_V) sind die Graphmorphismen, wobei f_E die Kanten auf Kanten und f_V die Knoten auf Knoten abbildet. Wenn aus einem Graphen zum Beispiel die Kanten ausgewählt werden sollen, wählt der Auswahlfunktor W_E aus dem Graphen $G = (E,V,c)$ die Kantenmenge E und die zugehörigen Kantenmorphismen f_E aus.

3.4 Markierte Graphen

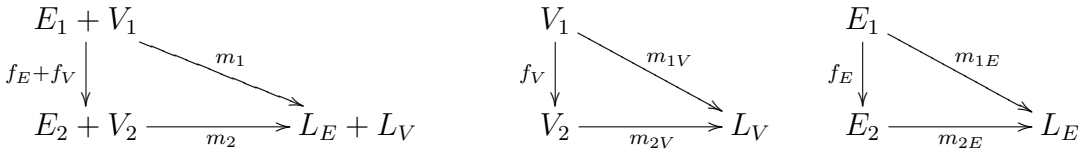
Dieser Abschnitt befasst sich mit markierten Graphen, es werden Knoten und/oder Kanten eines Graphen mit Elementen einer gegebenen Menge markiert. Dafür wird eine Kategorie **1** benötigt, welche nur aus einem Objekt $\bullet$ und einem Identitätsmorphismus $id_\bullet$ besteht. Sei nun Funktor M: **1** $\rightarrow$ *Set* definiert durch $\mathrm{M}(\mathbf{1}) := M$ und $\mathrm{M}(id_{\mathbf{1}}) := id_M$. Dieser Funktor bildet alle Objekte der Kategorie **1** auf die Markierung M ab. Zusammen mit einem Auswahlfunktor W, definiert M die Komma-Kategorie $Lgraph_M = (W,\mathrm{M})$, die Kategorie der mit M markierten Graphen. Die Bezeichnung *Lgraph* kommt aus dem Englischen von „labelled graph“. W wird entweder durch W_E, W_V oder W_{EV} ersetzt, je nachdem ob die Kanten, die Knoten oder die Knoten und Kanten markiert werden sollen. Sei W = W_V, also ist (W_V, M) die Kategorie der Knoten-markierten Graphen.



Wie bei dem rechten Diagramm zu sehen ist, bestehen die Objekte dieser Komma-Kategorie aus Knoten V denen über die Funktion m die Markierung M zugewiesen wird. Die Knoten werden über den Knotenmorphismus f_V aufeinander abgebildet. Auf der linken Seite ist die Struktur der Komma-Kategorie zu sehen, bevor die Funktoren angewandt wurden. Der Funktor $W_V(G)$ wählt aus dem Graphen G die Knoten V, der Funktor $M(id_\bullet)$ die Markierung M aus.

Wenn nun Knoten und Kanten gleichzeitig mit Markierungen aus verschiedenen Mengen markiert werden sollen, ergibt sich ein Problem, weil markierte Graphen nicht zwischen verschiedenen Markierungsmengen unterscheiden können. Es ist nicht möglich, den Knoten spezielle Knotenmarkierungen und den Kanten spezielle Kantenmarkierungen zuzuweisen. Deswegen muss eine andere Lösung gefunden werden, die im nächsten Abschnitt vorgestellt wird. Zunächst wird das Problem genauer betrachtet. In der folgenden Abbildung zeigt das linke Diagramm $Lgraph_M = (W_{EV}, M)$, die Komma-Kategorie der Knoten- und Kanten-markierten Graphen, jedoch diesmal mit verschiedenen Markierungen für Knoten und Kanten. L_V ist die Menge mit den Knotenmarkierungen und L_E die Menge mit den Kantenmarkierungen. Diese Definition ist jedoch falsch, weil es könnten zum Beispiel Kanten aus E mit Knotenmarkierungen aus L_V markiert werden. Es werden zwei verschiedene Komma-Kategorien benötigt, bei der getrennt die Kanten und Knoten markiert werden. Die beiden rechten Graphen stellen diese Kategorien dar. Bei dem mittleren Graphen werden Knotenmarkierungen L_V den Knoten aus V zugewiesen, bei dem rechten Graphen werden Kantenmarkierungen aus L_E den Kanten aus E

zugewiesen.



In dem nächsten Abschnitt wird die Zwei-Stufen-Markierung beschrieben, welche verschieden markierte Kanten und Knoten in einer Komma-Kategorie zusammenfasst.

3.5 Zwei-Stufen-Markierung

Das Problem, welches auftritt wenn Knoten und Kanten zusammen markiert werden sollen, wird gelöst, indem „Zwei-Stufen-Komma-Kategorien“ verwendet werden. Zuerst wird wie bei $Lgraph$ = (W$_E$,F) begonnen, die Kanten zu markieren. Diesmal wird jedoch F$_E$: $L_E \rightarrow Set$ anstelle von F verwendet, also ein Funktor der speziell Kanten-Markierungen aus L_E abbildet. Objekte dieser Komma-Kategorie (W_E, F_E), der Kanten-markierten Graphen, haben die Form (G,L_E, m_E) und die Morphismen $((f_E, f_V), f_{LE})$. Als nächstes wird der Funktor W_V so modifiziert, dass ihm der Kanten-markierte Graph übergeben werden kann. Dazu wird W_V folgendermaßen modifiziert.

$W'_V : (W_E, F_E) \rightarrow Set$:

$$W'_V(G, L_E, m_e) = W_V(G)$$
$$W'_V(f, f_{LE}) = W_V(f)$$

Die Komma-Kategorie (W'_V, F_V), welche auch als (W_E, W_V, F_E, F_V) geschrieben wird, ist die Kategorie der Graphen, welche an Knoten und Kanten mit Markierungen aus verschiedenen Mengen markiert sind. Die Objekte bestehen wieder aus einem 3-Tupel $((G, L_E, m_E), L_V, m_V)$, wobei das erste Element des Tupels ein Objekt von (W_E, F_E) ist. Genauso verhält es sich bei dem ersten Element der Morphismen $(((f_E, f_V), f_{LE}), f_{LV})$ der neuen Kategorie. Dieses besteht aus den Morphismen von (W_E, F_E), einem an den Kanten markierten Graphen. Zuerst wurde also die Komma-Kategorie der an den Kanten markierten Graphen definiert, danach wurden diese Graphen benutzt, um die Zwei-Schritt Komma-Kategorie der an den Knoten und Kanten markierten Graphen zu definieren.

4 Petrinetze

Petrinetze sind ein klassisches Modell, um nebenläufige und verteilte Prozesse und Systeme zu beschreiben. Petrinetze erlauben eine anschauliche grafische Darstellung und Analyse eines solchen Systems. Mit ihnen ist es möglich, Erreichbarkeit, Terminierung und partielle bzw. totale Verklemmung eines Systems zu untersuchen.

4.1 Petrinetze und zweiteilige Graphen

Ein Petrinetz ist ein gerichteter Graph, der aus Knoten und Kanten besteht. Es wird zwischen zwei Knotenarten unterschieden: den Stellen und den Transitionen. Stellen (üblicherweise als Kreise dargestellt) entsprechen Zuständen, Transitionen, die als Rechtecke visualisiert werden, entsprechen den Aktionen bzw. Ereignissen. Die Stellen und Transitionen des Netzes werden durch Kanten miteinander verbunden, wobei eine Kante jeweils nur von einer Stelle zu einer Transition führen darf, bzw. von einer Transition zu einer Stelle. Kanten zwischen gleichartigen Knoten

(z.B. von Stelle zu Stelle) sind nicht erlaubt.

Das dynamische Verhalten wird mittels Markierungen simuliert, welche sich auf den Stellen befinden und mittels Transitionen von Stelle zu Stelle gefeuert (engl. „firing a transition“ [3]) werden. Dies geschieht gemäß Schaltbedingungen, welche wie in Abbildung 2 aussehen. In Schaltbedingungen werden Transitionen oft als Strich

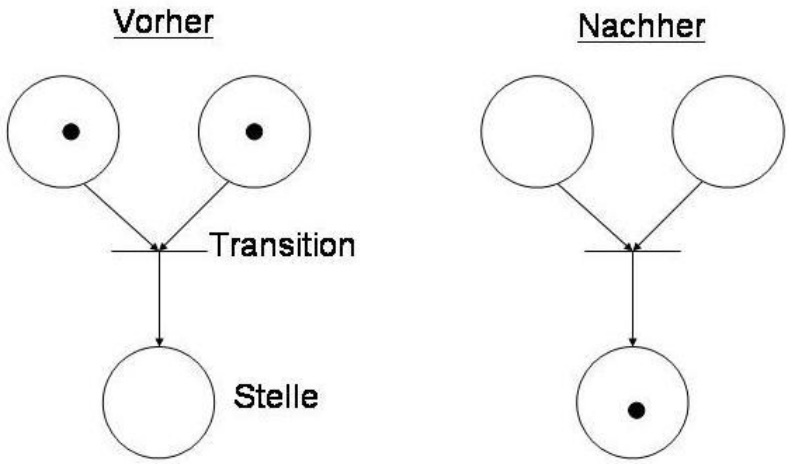


Abbildung 2: Schaltbedingung eines Petrinetzes

anstelle eines Rechtecks dargestellt. Die linke Seite zeigt den Zustand eines Petrinetzes vor dem Feuern der Transition, die rechte Seite zeigt das Petrinetz nach dem feuern. Eine Transition feuert, wenn alle Vorbedingungen erfüllt sind. In Abbildung 2 ist dies der Fall, wenn die beiden oberen Stellen mindestens eine Marke „besitzen“. Nach dem Feuern ist die untere Stelle markiert. Die Anzahl der Marken im gesamten System ist also nicht konstant, sie kann bei jedem feuern variieren.

Es gibt zwei verschiedene Arten von Petrinetzen, die Bedingungs-Ereignisnetze und die Stellen-Transitionsnetze. Bedingungs-Ereignisnetze sind Petrinetze, bei denen immer genau eine Marke über eine Kante fließt. Demgegenüber bieten die Stellen-Transitionsnetze die Möglichkeit, Kanten zu gewichten. Hierzu wird jeder Kante eine Zahl zugeordnet, welche die Anzahl der Marken angibt, die über diese Kante beim Schalten der Transition gefeuert werden. Die in Abbildung 3 dargestellte Transition eines Stellen-Transitionsnetzes feuert nur dann, wenn im Vorbereich zwei Marken vorhanden sind. In diesem Fall bekommen die Stellen im Nachbereich so viele Marken, wie ihre Gewichtung vorsieht.

Beide Arten von Petrinetzen werden in diesem Kapitel als Komma-Kategorie vorgestellt. Zunächst werden zweiteilige Graphen definiert, da diese die Grundlage von Petrinetzen bilden. In zweiteiligen Graphen gibt es zwei verschiedene Knotenmengen S und T. S ist die Knotenmenge der Stellen und T die Knotenmenge der Transitionen. Eine Kante verbindet jeweils einen Knoten aus S mit einem Knoten aus T, sie verbindet jedoch nie Knoten aus der gleichen Knotenmenge. Petrinetze basieren auf der Grundlage von zweiteiligen Graphen, da es auch zwei Knotenmengen gibt, nämlich die Stellen und die Transitionen. Diese sind untereinander durch Kanten verbunden, aber es kann nie eine Stelle mit einer Stelle oder eine Transition mit einer Transition verbunden sein.

Definition 7. Die Komma-Kategorie der zweiteiligen Graphen *BGraph* (engl.

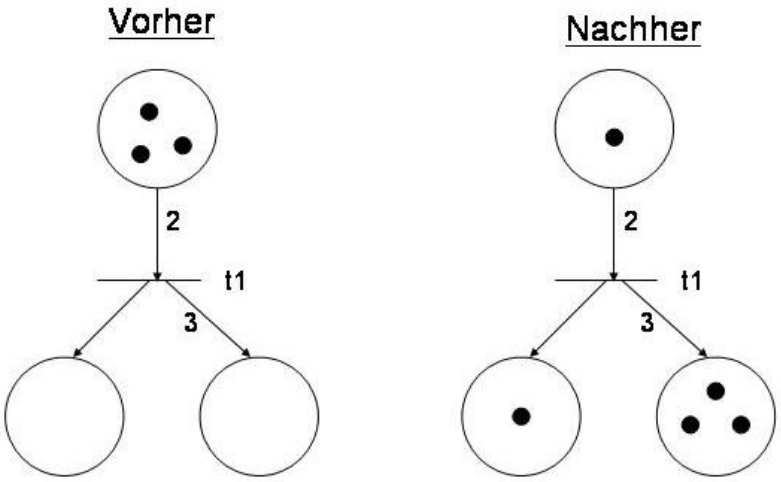


Abbildung 3: Schaltbedingung eines Stellen-Transitionsnetzes

„bipartite graphs“) besteht aus dem Identitätsfunktor Id und dem Funktor B: $Set \times Set \to Set$. Dieser ist definiert als B(S,T):= $S \times T + T \times S$. Der Funktor B bildet aus den beiden Knotenmengen S und T auf $S \times T$ oder $T \times S$ ab. So ist sichergestellt, das B aus jeder Knotenmenge genau einen Knoten wählt. Eine Funktion c weist diesesm Knotenpaar eine Kante aus E zu, somit ist jede Kante mit je einem Knoten aus S und T verbunden. Die Objekte der Komma-Kategorie (Id,B) sind dann zweiteilige Graphen $(E, S+T, c)$ mit $c : E \to S \times T + T \times S$. E ist die Kantenmenge, S und T sind die Knotenmengen der Stellen und Transitionen, und c ist die Funktion, welche den Kanten jeweils einen Knoten aus den Mengen S und T zuweist. Ein Morphismus dieser Komma-Kategorie bildet Kanten auf Kanten, Stellen auf Stellen und Transitionen auf Transitionen ab. Er ist ein 3-Tupel $(g_E : E_1 \to E_2, g_S : S_1 \to S_2, g_T : T_1 \to T_2)$. Die folgende Abbildung zeigt die Struktur der Komma-Kategorie $BGraph$.

$$\begin{array}{ccc} E_1 & \xrightarrow{c_1} & S_1 \times T_1 + T_1 \times S_1 \\ {\scriptstyle g_E}\downarrow & & \downarrow{\scriptstyle B(g_S,g_T)} \\ E_2 & \xrightarrow[c_2]{} & S_2 \times T_2 + T_2 \times S_2 \end{array}$$

□

Es werden Knotenpaare ausgewählt, wobei jeweils ein Knoten aus S und T stammt. Ein Knotenpaar ist entweder ein Element aus $S \times T$ oder aus $T \times S$, wobei das erste Element des Tupels der Startknoten und das zweite der Zielknoten ist. Einer Kante aus der Kantenmenge E wird nun über die Funktion c ein solches Knotenpaar zugewiesen. Da jeder Kante aus E zwei Knoten zugewiesen werden, spannen alle Kanten und Knoten einen zweiteiligen Graphen auf.

Für Bedingungs-Ereignisnetze wird eine weitere Definition benötigt. Da es bei markierten Graphen nur möglich ist, Knoten und Kanten (jedoch keine Markierungen), mittels Morphismen zu verändern, müssen die Markierungen durch ein Bündel von Knoten und Kanten repräsentiert werden. Sie können dann durch Morphismen verändert werden, und so ist es möglich, das Feuern von Transitionen dar-

zustellen, bei denen Markierungen verändert werden. Hierfür wird die Kategorie PO_D („partially ordered“) benötigt, welche als Objekte die Ordnung $(D, \sqsubseteq)$ und als Morphismen alle nicht-absteigenden Funktionen $h : D \to D$ hat. Eine Funktion $h : D \to D$ ist nicht-absteigend, wenn für alle $d \in D$, $d \sqsubseteq h(d)$ gilt. Wenn zum Beispiel $\sqsubseteq$ auf einer Menge von Atomen und Variablen aus der Logik gilt, bedeutet dies, dass Variablen durch Variablen und Atome ersetzt werden dürfen, aber Atome nicht durch Variablen. Die Kategorie der geordneten markierten Graphen ist definiert durch $POgraph = (W, D)$. W ist ein Auswahlfunktor, entweder W_E, W_V oder W_{EV}, je nachdem welcher Teil eines Graphen markiert werden soll. Der Funktor D: $PO_D \to Set$ wählt Elemente aus einer Ordnung $(D, \sqsubseteq)$ aus, mit welchen dann die Kanten und/oder Knoten eines Graphen markiert werden. Graphen aus $POgraph$ sind somit geordnet markiert, dies bedeutet, dass die Markierungen unterschieden werden können. Bei Bedingungs-Ereignisnetzen können Kanten und/oder Knoten entweder markiert ($\bullet$) oder nicht markiert ($\circ$) sein.

4.2 Komma-Kategorie der Bedingungs-Ereignisnetze

In diesem Abschnitt wird die Kategorie der Bedingungs-Ereignisnetze definiert, welche auf zweiteiligen Graphen basiert und eine partielle Ordnung benutzt. In der

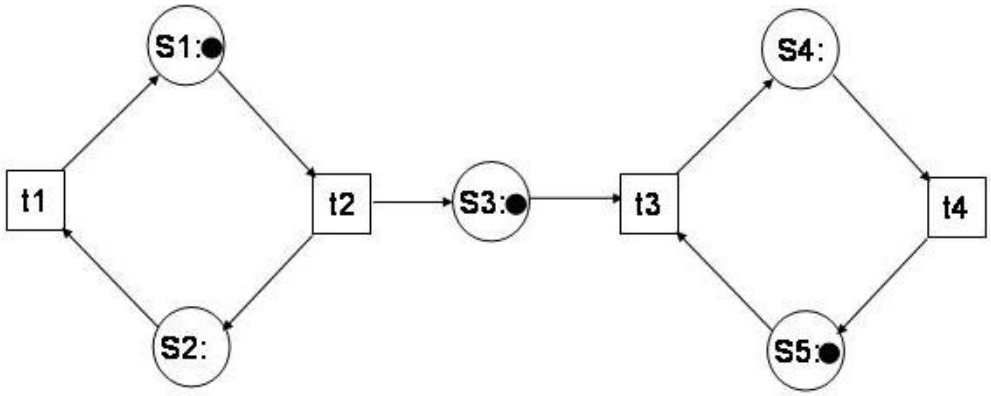


Abbildung 4: Bedingungs-Ereignisnetz

Abbildung 4 ist ein Bedingungs-Ereignisnetz mit fünf Stellen und vier Transitionen dargestellt. Die Stellen S1, S3 und S5 sind markiert, die Stellen S2 und S4 haben keine Markierungen. Als nächstes würde die Transition t3 feuern, weil die Stellen, die zu t3 führen, alle markiert sind. Nach dem Ffeuern wäre die Stelle S4 markiert.

Definition 8. Es wird die vorher definierte Ordnung $(D, \sqsubseteq)$ benutzt, in der alle Variablen v, entweder $v \sqsubseteq \bullet$ oder $v \sqsubseteq \circ$ erfüllen. Desweiteren wird der Vergesslichkeits-Funktor D und der Funktor P_T benötigt, welcher durch $\mathrm{P}_T(\bullet) := TL$ und $\mathrm{P}_T(id_\bullet) := id_{TL}$ definiert ist. TL ist eine Menge von Transitions-Markierungen. P_T weist also einer Markierung eine spezielle Transitions-Markierung aus TL zu. Zusammen mit dem Stellen-Auswahlfunktor $\mathrm{W}_{VS} : Bgraph \to Set$ und dem Transitionen-Auswahlfunktor $\mathrm{W}_{VT} : Bgraph \to Set$ bilden die Funktoren D und P_T die Zwei-Stufen-Komma-Kategorie $Pnet = (\mathrm{W}_{VS}, \mathrm{W}_{VT}, \mathrm{D}, \mathrm{P}_T)$. Diese ist die Kategorie der Bedingungs-Ereignisnetze, welche in der folgenden Abbildung als Diagramm dargestellt ist.

$$\begin{array}{ccc} S+T & \xrightarrow{m_S+m_T} & D+TL \\ {\scriptstyle f_S+f_T}\downarrow & & \downarrow{\scriptstyle h+id_{TL}} \\ S'+T' & \xrightarrow[m'_s+m'_T]{} & D+TL \end{array} \qquad \begin{array}{ccc} S & \xrightarrow{m_S} & D \\ {\scriptstyle f_S}\downarrow & & \downarrow{\scriptstyle h} \\ S' & \xrightarrow[m'_S]{} & D \end{array} \qquad \begin{array}{ccc} T & \xrightarrow{m_T} & TL \\ {\scriptstyle f_T}\downarrow & & \downarrow{\scriptstyle id_{TL}} \\ T' & \xrightarrow[m'_T]{} & TL \end{array}$$

□

Das linke Diagramm stellt das gleiche dar, wie die beiden rechten Diagramme zusammen, nur zeigen diese die genauen Verbindungen in der Struktur des Petrinetzes. Im mittleren Diagramm ist zu sehen, dass jeder Stelle S über die Funktion m_S ein Element aus der geordneten Menge D zugewiesen wird. Im rechten Diagramm wird jeder Transition T ein Element aus der Menge der Transitions-Markierungen TL zugewiesen.

4.3 Komma-Kategorie der Stellen-Transitionsnetze

In diesem Abschnitt werden Petrinetze im Sinne von Stellen-Transitionsnetzen untersucht. Der Unterschied zu den im letzten Abschnitt definierten Bedingungs-Ereignissnetzen ist, dass Stellen mit Mengen unterschiedlicher Markierungen markiert sein können. Dieses wird an dem Fünf-Philosophen-Problem verdeutlicht. Fünf Philosophen sitzen an einem runden Tisch und denken. Insgesamt liegen fünf Gabeln auf dem Tisch, je eine zwischen zwei Philosophen. Wenn ein Philosoph Hunger bekommt und die beiden Gabeln neben ihm frei sind, fängt er an zu essen. Wie in Abbildung 5 zu sehen ist, gibt es in einem Stellen-Transitionsnetz des 5-Philosophen-Problems die beiden Transitionen *satt* und *hungrig* und drei Stellen. Die Stellen

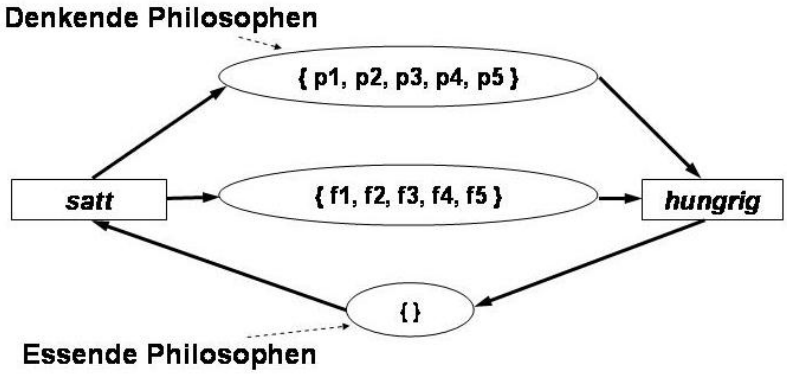


Abbildung 5: Startnetz des Philosophen Problems

bestehen zum einen aus der Menge der hungrigen Philosophen $\{p_1 \cdots p_5\}$ und aus der Menge der vorhandenen Gabeln $\{f_1 \cdots f_5\}$. Wenn Gabeln f_n und f_{n+1} frei sind, kann Philosoph p_n essen. Die Gabeln f_n und f_{n+1} werden aus der Stelle der vorhandenen Gabeln entfernt und die Markierung p_n wird durch eine Produktion aus der Stelle mit den hungrigen Philosophen entfernt und zu der dritten Stelle hinzugefügt. Dies ist die Stelle mit den Philosophen, welche sich beim Essen befinden und am Anfang leer ist. Abbildung 6 zeigt die Produktion, durch welche Philosoph p1 mit dem Essen beginnt. Abbildung 7 zeigt das Petrinetz, während der erste Philosoph isst.

Um Stellen-Transitionsnetze zu beschreiben, werden Multi-Mengen M(S) benötigt, wobei S eine beliebige Basismenge ist. Wenn für zwei Mengen M_1 und

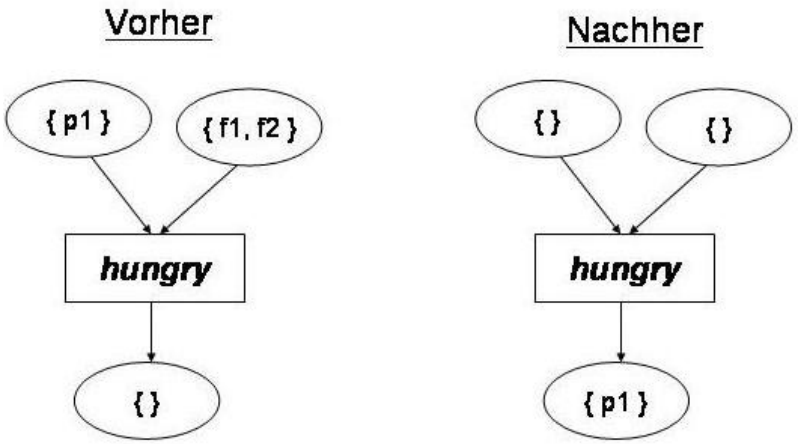


Abbildung 6: Eine Schaltbedingung des Philosophen Problems

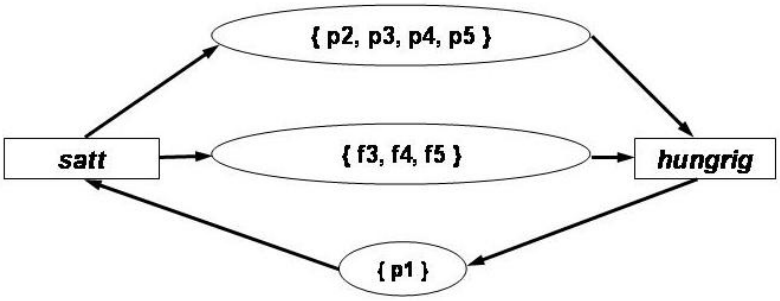


Abbildung 7: Petrinetz während Philosoph 1 isst

M_2 die Inklusion $M_1 \subseteq M_2$ gilt, bedeutet dies, dass jedes Element aus M_1 in M_2 vorkommen muss. Eine Ordnung, bestehend aus M(S) und der Inklusion $\subseteq$, kann als Objekt benutzt werden, um eine Kategorie für geordnetes Markieren zu definieren. Nun werden die Multisets um eine Indexmenge I erweitert. In dem Beispiel mit den Philosophen könnten zum Beispiel die Stellenbezeichner v_i als Indizes verwendet werden. v_1 für die Menge mit den denkenden Philosophen, v_2 für die Menge der Gabeln und v_3 für die Menge der essenden Philosophen. Dann würde die Menge I aus v_1, v_2 und v_3 bestehen. Die Kategorie, deren Objekte $I \times M(S)$ sind, wird mit $PO^I_{M(S)}$ bezeichnet. Die Morphismen dieser Kategorie sind Funktionen $h : I \times M(S) \to I \times M(S)$.

Definition 9. Seien $\mathrm{W}_{VS} : Bgraph \to Set$ und $\mathrm{W}_{VT} : Bgraph \to Set$ die Auswahlfunktoren für Stellen bzw. Transitionen eines zweiteiligen Graphs. Die Kategorie der Stellen-Transitionsnetze wird durch die Zwei-Stufen-Komma-Kategorie $PTnet = (\mathrm{W}_{VS},\mathrm{W}_{VT},\mathrm{P}_S,\mathrm{P}_T)$ beschrieben, wobei der Funktor $\mathrm{P}_S : PO^I_{M(S)} \to Set$ durch $\mathrm{P}_S(I \times M(S)) = I \times M(S)$ und $\mathrm{P}_S(h) = h$ definiert ist. □

5 Zustandscharts

In diesem Kapitel werden Zustandscharts eingeführt. Sie sind ein visueller Formalismus, welcher Möglichkeiten bietet, Zustände hierarchisch zu strukturieren und Kontrollflüsse zwischen diesen Zuständen zu beschreiben. Zuerst wird der Aufbau eines Zustandscharts erklärt, welcher unter anderem an einem Beispiel verdeutlicht wird. Im zweiten Abschnitt werden Zustandscharts als Komma-Kategorie definiert.

5.1 Aufbau eines Zustandscharts

Zustände werden in Zustandscharts als Boxen, Transitionen durch Pfeile dargestellt. Wenn eine Transition ausgeführt wird, wird die Kontrolle an den Zustand übergeben, auf den der Pfeil zeigt. Ein Pfeil kann auch wiederum auf einen Zustandschart zeigen, dann wird dieser Unter-Zustandschart aktiviert. Es wird bei Eintritt in einen Zustandschart in einem Start-Zustand gestartet. Dieser ist durch eine Pfeilspitze gekennzeichnet, die auf ihn zeigt. Falls in einem Unter-Chart ein Geschichts-Knoten H (engl. „history node“) enthalten ist, kann auch an dem Zustand gestartet werden, auf den der Knoten H zeigt. In diesem Fall wird zur Laufzeit entschieden, an welchen Zustand die Kontrolle übergeht. Das dient dazu, dass wenn Prozesse unterbrochen werden, sie an dem Punkt wieder starten können, an dem sie unterbrochen wurden. Ein weiterer Knoten, der Kontroll-Knoten C (engl. „control node“), markiert den Zustand, an dem sich die Kontrolle aktuell befindet. Parallele Prozesse werden durch gestrichelte Linien zwischen ihren Zustandscharts gekennzeichnet. Wenn zum Beispiel eine Transition ausgeführt wird, die zu einem Chart mit zwei parallelen Unter-Charts A und B führt, wird bei A und B ein Kontroll-Knoten auf den Startzustand gesetzt. Diese laufen dann parallel durch das jeweilige System. In Abbildung 8 wird ein System betrachtet, welches aus einem Sender und

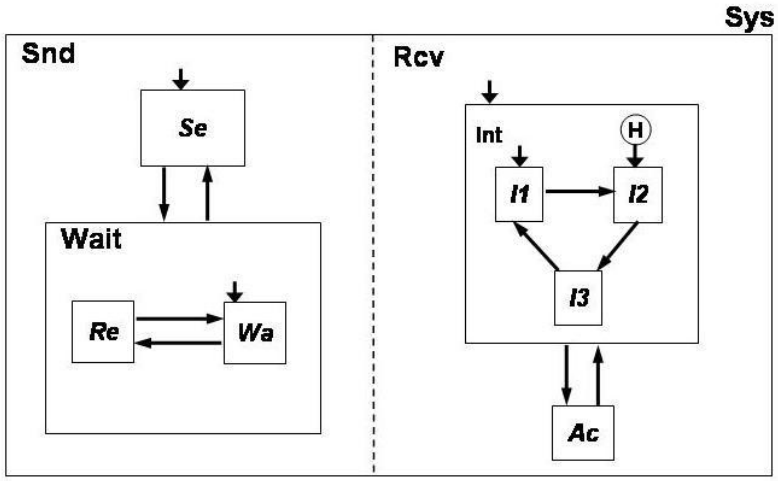


Abbildung 8: Beispiel eines Zustandscharts mit Sender und Empfänger

einem Empfänger besteht. Da der Sender und der Empfänger durch eine gestrichelte Linie getrennt sind, arbeiten sie parallel. Der Sender Snd schickt im Zustand Se eine Nachricht an den Empfänger und geht dann in den Unterchart $Wait$. Die Pfeilspitze auf den Knoten Wa legt fest, dass in $Wait$ bei dem Knoten Wa begonnen wird. Wenn der Empfänger nach einer in Wa festgelegten Zeit keine Empfangsbestätigung

schickt, geht die Kontrolle an den Knoten *Re* über, welcher die Nachricht wiederholt abschickt. Der Empfänger *Rcv* verrichtet in dem Unterchart Int solange interne Aufgaben, bis eine Nachricht vom Sender verfügbar ist. Dann gibt es zwei Möglichkeiten. Zum einen kann der Sender seine internen Aufgaben weiterführen. Zum anderen kann er seinen aktuellen Zustand mit dem Knoten H markieren, zum Zustand *Ac* gehen und eine Empfangsbestätigung senden. Danach wird der Empfänger die internen Aufgaben an dem Knoten H weiterführen, an dem er unterbrochen wurde. Zu beachten ist, dass der Sender und der Empfänger parallel laufen, d.h. der Sender sendet und wartet solange auf eine Bestätigung, bis der Empfänger seine internen Aufgaben unterbricht und den Empfang bestätigt.

Als nächstes werden einige Produktionen vom Zustandschart des Beispiels vorgestellt. Es muss nicht der komplette Zustandschart in der Produktion aufgeführt sein, es reicht aus, wenn alle Zustände, Transitionen und Knoten, die beim Ausführen der Produktion benötigt werden, vorhanden sind.

Als erstes einfaches Beispiel wird in Abbildung 9 eine Produktion vorgestellt, die eine lokale Transition von Int beschreibt. Wenn der Kontroll-Knoten C auf den

Abbildung 9: Kontrollübergabe zwischen zwei Zuständen

Zustand $I2$ zeigt und von $I2$ nach $I3$ eine Transition führt, wird C nach Ausführen der Produktion auf $I3$ zeigen. Dies ist eine einfache Produktion, welche die Kontrolle um einen Zustand weiter setzt.

Abbildung 10 zeigt die Startsituation des Systems, bevor Sender und Empfänger anfangen zu arbeiten. Der Kontroll-Knoten C zeigt auf den äußersten Zustandschart. Nach dem Ausführen der Produktion zeigt je ein C auf Se und Int. C zeigt auf Se, weil Se mit einer Pfeilspitze markiert ist, welche den Anfangszustand eines Unter-Charts markiert. Da jetzt zwei Prozesse parallel ablaufen, gibt es zwei Kontroll-Knoten.

Abbildung 11 verdeutlicht die Wirkungsweise des Geschichts-Knoten H. Auf der linken Seite ist der Zustandschart Int zu sehen, auf den C zeigt. Wenn ein Knoten H existiert, welcher auf I_2 zeigt, wird nach der Produktion der Kontroll-Knoten C auf I_2 zeigen und H gelöscht sein. Die Zustände und Transitionen, welche die Produktion nicht beeinflussen, müssen nicht aufgeführt sein. In diesem Fall sind das I_1, I_3 und die mit I_1 und I_3 verbundenen Transitionen.

Abbildung 12 zeigt eine Produktion des Empfängers, die auf eine Nachricht vom Sender reagiert. Der Kontroll-Knoten C in Int wird zum Geschichts-Knoten H, um nach Abschicken der Sendebestätigung den Prozess im unterbrochenen Zustand wieder aufnehmen zu können. Die Kontrolle geht an den Knoten *Ac*, welcher die Sendebestätigung an Sender schicken wird.

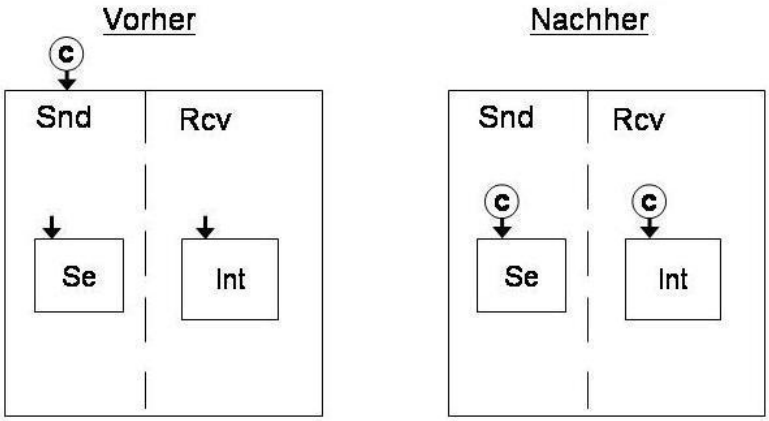


Abbildung 10: übergabe der Kontrolle an Sender und Empfänger

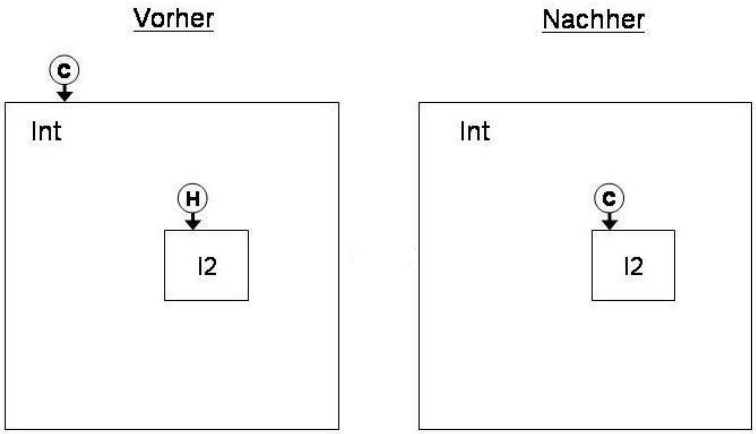


Abbildung 11: Funktion des Geschichts-Knotens in einem Petrinetz

Wie Zustandscharts zeigen, müssen Graphtransformations-Systeme nicht unbedingt Kanten-Markierungen zur Verfügung stellen, um komplizierte Prozesse zu simulieren. Spezielle Knoten, wie C oder H, sorgen dafür, dass die richtigen Transitionen ausgeführt werden.

5.2 Definition als Komma-Kategorie

Dieser Abschnitt beschreibt, wie Zustandscharts im Komma-Kategorie-Ansatz definiert werden. Die Definition wird hier auf Knotenmarkierungen beschränkt, es wäre aber auch möglich, Zustandscharts mit Knoten- und Kantenmarkierungen zu definieren. Dafür müsste nur die Technik der Zwei-Stufen-Markierung angewendet werden, welche in Abschnitt 3.5 beschrieben wurde. Das bedeutet, dass zuerst eine Komma-Kategorie für Zustandscharts mit Kantenmarkierungen definiert werden müsste, welche dann als Grundlage für die Kategorie der Zustandscharts mit Knoten- und Kantenmarkierungen benutzt wird.

Um Zustandscharts als Komma-Kategorie zu definieren, wird eine Menge P und $Lgraph_M$ benötigt, die Kategorie der mit Elementen aus M markierten Graphen.

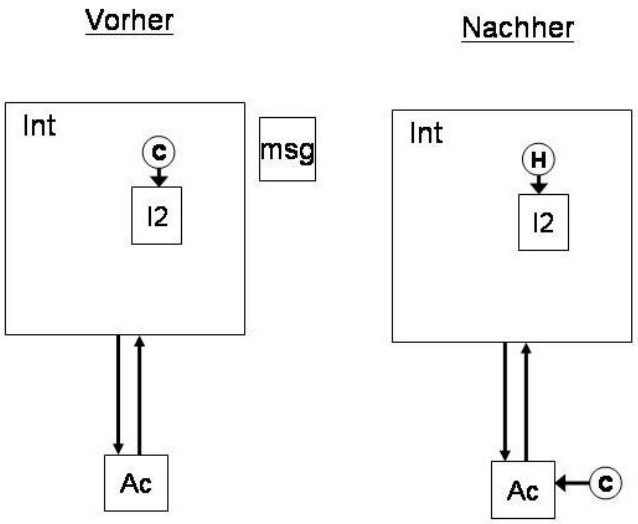


Abbildung 12: Kontrollfluss bei Nachrichteneingang im Empfänger

M ist in diesem Fall eine Menge von Identifizierern, P ist eine Menge von Aktionen. Identifizierer sind als Zustände anzusehen, welche weitere Zustände aufrufen. Dies bedeutet, dass sie auf weitere Identifizierer, Aktionen, Transitionen, Kontroll-Knoten oder Geschichts-Knoten verweisen. Aktionen hingegen sind Zustände die final sind, was bedeutet, dass sie keine weiteren Unterzustände oder Aktionen besitzen. In dem Sender/Empfänger-Beispiel sind Se, Re, Wa, Ac, I_1, I_2 und I_3 Aktionen und $Snd, Rcv, Wait, Int$ und Sys Identifizierer. $Wait$ verweist zum Beispiel auf einen Unterchart, der die Aktionen Re und Wa und zwei Transitionen zwischen den beiden Aktionen enthält. Eine Funktion σ definiert einen Zustandschart, sie weist Identifizierern entweder markierte Graphen, also Untercharts, oder Aktionen zu. Die Funktion sieht dann folgendermaßen aus: $\sigma : M \rightarrow Obj_{Lgraph_M} + P'$, wobei P' zusätzlich zu den Aktionen P noch die Knoten C und H enthält. In dem Beispiel ist Snd also ein Graph der zwei Knoten enthält, der eine ist mit $Wait$ markiert, der andere mit einem Identifizierer, dessen Name nicht bekannt ist. Die Funktion σ weist $Wait$ einen Graphen zu, der die beiden Knoten Wa und Re enthält. Dem anderem Identifizierer wird die Aktion Se zugewiesen.

Definition 10. Sei I eine Menge von Identifizierern, $M \subseteq I$ und ST_I die Kategorie, welche als Objekt die Menge der mit I markierten Graphen enthält.

Sei PO_I die Kategorie, welche als Objekte Teilmengen $M \subseteq I$ und als Morphismen Inklusionen enthält. Der Funktor I: $PO_I \rightarrow Set$ mit $\mathrm{I}(M) = M$ und $\mathrm{I}(M \subseteq M')$, welcher die Injektion von M nach M' ist, bildet zusammen mit dem Funktor St die Komma-Kategorie der Zustandscharts. St: $ST_I + \mathbf{1} \rightarrow Set$ ist folgendermaßen definiert:

$$\begin{array}{ll} \mathrm{St}(G) = G & \mathrm{St}(\bullet) = P' \\ \mathrm{St}(f) = f & \mathrm{St}(id_\bullet) = id_{P'} \end{array}$$

$Stcharts$ = (I,St) ist dann die Komma-Kategorie der Zustandscharts. Die Struktur von $Stcharts$ ist in folgender Abbildung dargestellt.

$$\begin{array}{ccc} M & \xrightarrow{\sigma} & Obj_{Lgraph_I} + P' \\ \subseteq\downarrow & & \downarrow f+id_{P'} \\ M' & \xrightarrow[\sigma']{} & Obj_{Lgraph_I} + P' \end{array}$$

□

Das bedeutet, dass Zustandscharts aus einer endlichen Menge von Identifizierern bestehen, welche von einer Funktion σ Graphen aus Obj_{Lgraph_I} oder Aktionen aus P' zugewiesen bekommen. In dem Sender/Empfänger Beispiel aus Abbildung 8 wird dem Identifizierer *Rcv* ein Graph mit zwei Knoten zugewiesen. Ein Knoten ist mit der Markierung *Int* markiert, welcher wiederum ein Graph zugewiesen ist. Der andere Knoten ist mit einem unbekannten Identifizierer markiert, welcher auf die Aktion *Ac* zeigt. Morphismen in Zustandscharts erhalten Identifizierer und Aktionen, das bedeutet, dass Aktionen und Identifizierer durch Ausführung von Morphismen nicht verändert werden können. Genauso können zwei Knoten, die mit der gleichen Markierung markiert sind, nicht auf verschiedene Graphen verweisen. Sollte in dem Sender/Empfänger-Beispiel in *Rcv* noch ein $Wait$ auftauchen, müsste der $Wait$ zugehörige Graph genauso aussehen wie der von $Wait$ in *Snd*.

Diese Definition von Zustandscharts erlaubt es, dass mehrere Startzustände existieren. Startzustände sind alle Knoten die nicht in $\sigma[M]$ vorkommen, also diejenigen, die nicht durch die Ausführung von σ erreicht werden können. In dem Sender/Empfänger-Beispiel gibt es nur einen Startzustand, nämlich den durch *Sys* markierten.

6 Zusammenfassung

Mit Hilfe von Komma-Kategorien können komplizierte Strukturen aufgebaut werden. So wurden in dieser Arbeit beispielsweise markierte und zweiteilige Graphen als Komma-Kategorien definiert, welche auf einer allgemeinen Komma-Kategorie-Definition für Graphen basieren. Die markierten und zweiteiligen Graphen wurden anschließend dazu benutzt, Petrinetze als Komma-Kategorie zu definieren. Zusätzlich wurde eine Definition von Zustandscharts gegeben, die analog zu den Petrinetzen auf der Komma-Kategorie der markierten Graphen basiert.

Es wurde gezeigt, wie verschiedene Strukturen, wie Graphen, markierte Graphen und zweiteilige Graphen, kombiniert werden können, um komplizierte Strukturen, wie Petrinetze und Zustandscharts, zu definieren. Genauso können sogenannte Prozess-Systeme beschrieben werden. Die Grundstruktur von Prozess-Systemen wird durch Hypergraphen beschrieben. Die Knoten des Graphen stellen Zustände und Kommunikationskanäle dar, welche mit Teilen eines globalen Dschungels markiert sind. Dieser globale Dschungel stellt die Datenmenge des Prozess-Systems dar, und jeder Zustand ist mit der von ihm benötigten Untermenge des Dschungels markiert. In dieser Arbeit wurde gezeigt, dass es möglich ist, unterschiedliche Strukturen mit Hilfe von Komma-Kategorien zu definieren. Auf der Grundlage einfacher Komma-Kategorien, können komplizierte Systeme, wie ein Prozess-System, als Komma-Kategorie beschrieben werden.

Literatur

[1] **Handbook of Graph Grammars and Computing by Graph Transformation, Volume 3**; H. Ehrig, H.-J. Kreowski, U. Monta-

nari, G. Rozenberg; World Scientific; ISBN 981-02-4021-X;

[2] **Merriam-Webster Online** http://www.m-w.com/home.htm

[3] http://worldserver.oleane.com/adv/elstech/petrinet.htm