Modellierung und Petrinetze

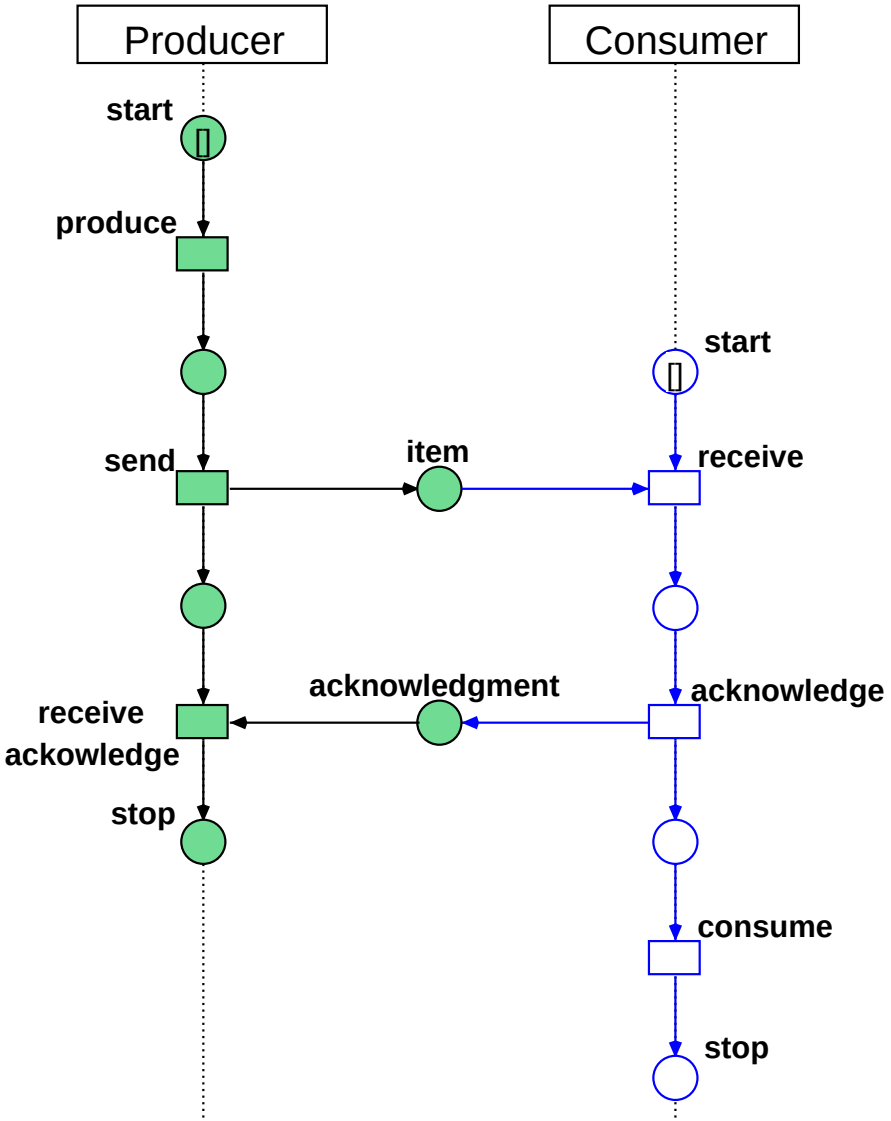
Reflexive Selbstorganisation in Multiagentensystemen

Michael Köhler

Department für Informatik
Universität Hamburg

10. Januar 2008

Das Dienstnetz: Producer/Consumer



Dienste

- Ein Organisationsschema ist die Grundstruktur, innerhalb deren die Aktivitäten einer Organisation stattfinden.
- Die eigentlichen Aktivitäten werden durch Dienstnetze beschrieben.
- Dienstnetze sind gefärbte Petrinetze, deren Transitionen Rollen zugeschrieben sind.
- Die Farben der Marken werden durch eine Konzeptbeschreibungssprache definiert.

Ablaufnetze

- Nur ungewichtete Netze $N = (P, T, F)$
- keine isolierten Transitionen: $\forall t \in T : {}^\bullet t \neq \emptyset \wedge t^\bullet \neq \emptyset$.
- Solche Netze heißen *T-schlicht*.
- Jeder Netzknoten liegt auf einem Pfad zwischen den Randknoten.
- Mit ${}^\circ N = \{n \in (P \cup T) \mid {}^\bullet n = \emptyset\}$ und $N^\circ = \{n \in (P \cup T) \mid n^\bullet = \emptyset\}$:

$$\forall n \in (P \cup T) : \exists i \in {}^\circ N, o \in N^\circ : iF^*nF^*o \tag{1}$$

- Beachte: Für T-schlichte Netze muss ${}^\circ N, N^\circ \subseteq P$ gelten.

Definition

Ein *Ablaufnetz* ist ein T-schlichtes Petrinetz $N = (P, T, F)$, bei dem jeder Knoten $n \in P \cup T$ auf einem Pfad zwischen einem ${}^\circ N$ und N° liegt.
kanonische Initialmarkierung m_i/ Finalmarkierung m_f:

$$m_i(p) = \begin{cases} 1 & \text{falls } p \in {}^\circ N \\ 0 & \text{sonst} \end{cases} \quad \text{und} \quad m_f(p) = \begin{cases} 1 & \text{falls } p \in N^\circ \\ 0 & \text{sonst} \end{cases}$$

Heterogene Signatur

Definition

Eine (heterogene) *Signatur* $\Sigma = (K, \Omega)$ besteht aus

- einer endlichen Menge an *Sorten* K und
- einer endlichen indizierten Menge $\Omega = \{\Omega_{w,k}\}_{w \in K^*, k \in K}$ von *Operatoren*.

Notation: $\omega : k_1 \times \ldots \times k_n \rightarrow k$ für $\omega \in \Omega_{k_1 \ldots k_n, k}$ notiert.

Operatoren $\omega \in \Omega_{\lambda,k}$ bezeichnen Konstanten der Sorte k.

Terme I

Sei eine Signatur $\Sigma = (K, \Omega)$ und eine indizierte Variablenmenge $X = (X_k \mid k \in K)$ mit disjunkten Variablenmengen X_k gegeben.

Definition

Die Menge der *Terme* $\mathbb{T}^k_\Sigma(X)$ der Sorte k über den Variablen X ist induktiv definiert:

1. Für alle Variablen $x_k \in X_k$ gilt $x_k \in \mathbb{T}^k_\Sigma(X)$.
2. Für alle Konstanten $\omega : \lambda \to k$ gilt $\omega \in \mathbb{T}^k_\Sigma(X)$.
3. Sei $\omega : k_1 \cdots k_n \to k$ und $t_i \in \mathbb{T}^{k_i}_\Sigma(X)$ für alle $1 \leq i \leq n$, dann gilt $\omega(t_1, \ldots, t_n) \in \mathbb{T}^k_\Sigma(X)$.

Dann bezeichnet $\mathbb{T}_\Sigma(X) := \bigcup_{k \in K} \mathbb{T}^k_\Sigma(X)$ die Menge aller Terme mit den Variablen X.
Grundterme $\mathbb{T}_\Sigma := \bigcup_{k \in K} \mathbb{T}^k_\Sigma$ mit $\mathbb{T}^k_\Sigma := \mathbb{T}^k_\Sigma(\emptyset)$

Natürliche Zahlen I

Sorten: ELEM, NAT und LIST

	0	$\in$	$\Omega_{\lambda,\mathrm{NAT}}$
Operatoren	*suc*	$\in$	$\Omega_{\mathrm{NAT},\mathrm{NAT}}$
	add	$\in$	$\Omega_{\mathrm{NAT}\cdot\mathrm{NAT},\mathrm{NAT}}$

In MAUDE-Syntax:

```
sort Nat .
op 0 : -> Nat .
op suc : Nat -> Nat .
op _+_: Nat Nat -> Nat .

vars M N : Nat .
eq 0 + N = N .
eq suc(M) + N = suc(M + N) .
```

Natürliche Zahlen II

```
sort Elem .
op a b c d e : -> Elem .

sort List .
op [] : -> List .
op _._  : Elem List -> List .
op length :  List -> Nat .

var E : Elem .
vars L : List .
eq length([]) = 0 .
eq length(E.L) = suc(length(L)) .
```

Die Gleichungen werden intern als Ersetzungsregeln von links nach rechts angewendet und erzeugen eine Normalform.

Membership Equational Logic I

Zusätzlich zu den Sorten K können Teilsorten definiert werden, wobei mehrere Teilsorten einer Sorte zugeordnet werden.

Definition

Eine MEL-Signatur $\Sigma = (K, \Omega, S)$ besteht aus einer endlichen Menge an Sorten K, so dass (K, Ω) eine Signatur ist, und einer K-indizierten Menge $S = (S_k \mid k \in K)$ an paarweise disjunkten *Teilsorten*, wobei $S_k \cap K = \emptyset$ angenommen wird.

- Werden keine Teilsorten definiert, so entspricht die MEL einer mehrsortigen Signatur.
- In jeder Algebra einer MEL sind die Träger der Teilsorten durch Teilmengen der Trägermenge der Hauptsorte definiert.
- Eine MEL erlaubt es, Aussagen über das Enthaltensein eines Terms in einer Teilsorte zu treffen.

Formeln I

Definition

Eine *MEL-Formel* ϕ ist von der Form:

1. membership: $M \in s$ für $s \in S_k$ und $M \in \mathbb{T}^k_\Sigma(x)$.
2. $M = N$ für $M, N \in \mathbb{T}^k_\Sigma(X)$.
3. $(\phi_1 \wedge \ldots \wedge \phi_n)$ für Formeln ϕ_i.

Ein *MEL-Axiom* hat die Form (für eine MEL-Formel ϕ):

1. $\forall X : \phi \Longrightarrow (M \in s)$.
2. $\forall X : \phi \Longrightarrow (M = N)$.

Eine *Datentyp-Spezifikation* $\mathcal{D} = (\Sigma, V, E)$ besteht aus einer Signatur $\Sigma = (K, \Omega, S)$, einer disjunkten Familie von Variablen $V = (V_k \mid k \in K)$ und einer Familie $E = (E_k \mid k \in K)$ von Axiomen.

Teilsorten werden eingesetzt, um wertabhängigige Typisierungen aussprechen zu können.

- Betrachten wir dazu das Beispiel eines Graphen (V, E) mit Knotenmenge V und Kantenmenge E.
- Eine Kantenwort ist ein Element $w \in E^*$, also ein Element des freien Monoids $(E^*, \cdot, \epsilon)$.
- Nun ist aber nicht jedes Kantenwort auch ein Pfad im Graph.
- Ein Pfad w des Graphen zeichnet sich dadurch aus, dass für jede Zerlegung $w = w_1 \cdot w_2$ gilt, dass der Endknoten von w_1 mit dem Anfangsknoten von w_2 identisch ist.
- Dies ist aber eine Bedingung, die wir nicht mehr auf der Ebene der Sorte der Worte direkt festmachen können, da die Eigenschaft, ob $w_1 \cdot w_2$ ein Pfad ist, nicht nur von $\cdot$, sondern auch an den Attributen von w_1 und w_2 abhängt.
- Diese wertabhängige Typisierung der Teilsorte Path lässt sich leicht in MEL formulieren.

- Sei e eine Kantenvariable und p eine Pfadvariable, dann fomulieren wir das Axiom:

$$\forall e, p : \text{target}(e) = \text{source}(p) \Longrightarrow (e \cdot p) \in \text{Path}$$

Betrachten wir die Repräsentation des Graphen in Form einer Signatur. Der Graph sei konkret gegeben als:

$$\begin{array}{ccc} n_1 & \xrightarrow{a} & n_2 \\ {\scriptstyle b}\downarrow & & \downarrow{\scriptstyle c} \\ n_3 & \xrightarrow{d} & n_4 \end{array}$$

```
fmod A-GRAPH is
  sorts Edge Node .
  ops n1 n2 n3 n4 : -> Node .
  ops a b c d : Edge .
  ops source target : Edge -> Node .

  eq source(a) = n1 .   eq target(a) = n2 .
  eq source(b) = n1 .   eq target(b) = n3 .
  eq source(c) = n2 .   eq target(c) = n4 .
  eq source(d) = n3 .   eq target(d) = n4 .
endfm
```

Die Sorte `Path?` beschreibt alle potentiellen Pfade, die syntaktisch durch den Operator `;` bildbar sind. Ein echter Pfad des Graphen wird durch die Sorte `Path` beschrieben. Jede Kante ist ein echter Pfad. Alle weiteren Pfade sind die Konkatenation zweier Pfade, was durch die folgende Membership-Bedingung ausgedrückt wird:

```
cmb (E ; P) : Path if target(E) == source (P) .
```

Die gesamte Spezifikation von Pfaden ergibt sich wie folgt:

```
fmod PATH is
  protecting NAT .
  protecting A-GRAPH .

  sorts Path Path? .
  subsorts Edge < Path < Path? .

  op _;_ Path? Path? -> Path? [assoc] .
  ops source target : Path -> Node .
  op length : Path -> Nat .

  var E : Edge .
  var P : Path .

  cmb (E ; P) : Path if target(E) == source (P) .

  eq source(E ; P) = source(E) .
  eq target(E ; P) = target(P) .
```

```
    eq length(E) = suc(0) .
    eq length(E ; P) = suc(0) +  length(P) .
endfm
```

In diesem Beispiel werden Subsorten verwendet, um Pfade nicht erst auf der Termebene, sondern schon auf Ebene des Typsystems auszuzeichnen.

Beispielsweise erwartet der Operator `length` eine Argument vom Typ `Path`. In der Gleichung `eq length(E ; P) = suc(0) + length(P)` ist `(E ; P)` zunächst vom Typ `Path?`. Es ist also notwendig, zunächst `(E ; P) : Path` herzuleiten, bevor die Gleichung angwendet werden kann.

Algebra

Definition

Eine Σ-*Algebra* $A = (\llbracket K \rrbracket_A, \llbracket \Omega \rrbracket_A, \llbracket S \rrbracket_A)$ zu der Signatur $\Sigma = (K, \Omega, S)$ besteht aus

- einer indizierten Familie von disjunkten Trägermengen $\llbracket K \rrbracket_A = (\llbracket k \rrbracket_A \mid k \in K)$,
- einer indizierten Familie von Funktionen

$$\llbracket \Omega \rrbracket_A = (\llbracket \omega \rrbracket_A : \llbracket k_1 \rrbracket_A \times \cdots \times \llbracket k_n \rrbracket_A \to \llbracket k \rrbracket_A \mid \omega \in \Omega_{k_1 \cdots k_n, k})$$

- einer indizierten Familie von disjunkten Submengen $\llbracket S \rrbracket_A = (\llbracket S_k \rrbracket_A \mid k \in K)$, so dass für alle $\llbracket s \rrbracket_A \in \llbracket S_k \rrbracket_A$ auch $\llbracket s \rrbracket_A \subseteq \llbracket k \rrbracket_A$ gilt.

Variablenbelegung

Eine *Variablenbelegung* $\alpha : X \to [\![K]\!]_A$ ist eine Familie $(\alpha_k \mid k \in K)$ von Abbildungen $\alpha_k : X_k \to [\![k]\!]_A$.

Auswertungsabbildung $\overline{\alpha} : \mathbb{T}_\Sigma(X) \to [\![K]\!]_A$ auf Termen:

1. $\overline{\alpha}(x_k) = \alpha(x_k)$ für $x_k \in X_k$.
2. $\overline{\alpha}(\sigma(t_1, \ldots, t_n)) = [\![\sigma]\!]_A(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_n))$ für $\sigma(t_1, \ldots, t_n) \in \mathbb{T}^k_\Sigma(X)$.

Gültigkeit

Definition

Gültigkeit einer MEL-Formel ϕ unter einer Belegung α:

1. $A, \alpha \models (M \in s)$, gdw. $\alpha(M) \in [\![s]\!]_A$.
2. $A, \alpha \models (M = N)$, gdw. $\alpha(M) = \alpha(N)$.
3. $A, \alpha \models (\phi_1 \wedge \ldots \wedge \phi_n)$, gdw. $A, \alpha \models \phi_i$ für alle $1 \leq i \leq n$.

- Ein Axiom $\forall X : \phi \Longrightarrow \phi'$ ist in der Algebra A gültig (notiert als $A \models \forall X : \phi \Longrightarrow \phi'$), gdw. ϕ' in allen Belegungen α gültig ist, für die auch ϕ gültig ist.
- Eine Σ-Algebra A, für die alle Axiome der Spezifikation $\mathcal{D} = (\Sigma, X, E)$ gültig sind, heißt $\mathcal{D}$-*Theorie*.

Konzeptbeschreibung

- Beschreibungslogik bildet Formeln über elementaren Konzepten und Relationen zwischen ihnen
- Relationen werden als *Rollen* bezeichnet – nicht zu Verwechseln mit den Rollen der nachfolgenden Dienstnetze
- Die Subsumption $\sqsubseteq$ ist das alleinige Prädikat der Logik.

Beispiel I

- Seien Woman und Man atomare Konzepte und hasChild die Eltern/Kind-Relation.
- Terminologie (die T-Box):

$$\begin{aligned} \text{Person} &\equiv \text{Woman} \sqcup \text{Man} \\ \text{Mother} &\equiv \text{Woman} \sqcap \exists \text{hasChild.Person} \\ \text{Father} &\equiv \text{Man} \sqcap \exists \text{hasChild.Person} \\ \text{Parent} &\equiv \text{Mother} \sqcup \text{Father} \\ \text{Grandmother} &\equiv \text{Woman} \sqcap \exists \text{hasChild.Parent} \end{aligned}$$

- Zyklischen Definitionen sind hierbei nicht erlaubt.
- Die Subsumption bezeichnet das Enthaltensein eines Konzeptes in einem anderen:

$$\text{Mother} \sqsubseteq \text{Woman} \sqsubseteq \text{Person}$$

Beispiel II

- Daneben speichert eine Wissensdatenbank (die A-Box) Fakten ab:

$$\text{Woman}(\mathit{eva})$$
$$\text{Man}(\mathit{adam})$$
$$\text{hasChild}(\mathit{eva}, \mathit{abel}), \text{hasChild}(\mathit{eva}, \mathit{kain})$$

Definition

Seien N_C und N_R paarweise disjunkte, abzählbar unendliche Mengen von *Konzept-*, respektive *Rollennamen*.
Dann ist die *Konzeptbeschreibungen* der Beschreibungslogik $\mathcal{ALC}$ folgendermaßen definiert:

1. Jeder atomare Ausdruck A ist eine $\mathcal{ALC}$-Konzeptbeschreibung.
2. $\top$ und $\bot$ sind $\mathcal{ALC}$-Konzeptbeschreibungen.
3. Sind C und D jeweils $\mathcal{ALC}$-Konzeptbeschreibungen und R eine Rolle, so sind auch $\neg C$, $(C \sqcup D)$, $(C \sqcap D)$, $\exists R.C$ und $\forall R.C$ auch $\mathcal{ALC}$-Konzeptbeschreibungen.

Semantik I

Man erkennt, dass die Konzepte unäre und die Rollen binäre Relationen sind, die in der Beschreibungslogik allerdings in einer variablenfreien Notation verwendet werden:

$$C \sqcap D \text{ steht für } \forall x : C(x) \wedge D(x)$$

Konzepte werden durch Teilmengen eines Universums und Rollen als binäre Relationen definiert.

Definition

Eine *Interpretation* $(\Delta, \cdot^{\mathcal{I}})$ besteht aus einer nicht-leeren Grundmenge Δ und einer Interpretationsfunktion $\cdot^{\mathcal{I}}$, die

- jedem Konzeptbezeichner $A \in N_C$ eine Menge $A^{\mathcal{I}} \subseteq \Delta$ und
- jedem Rollenbezeichner $R \in N_R$ eine binäre Relation $R^{\mathcal{I}} \subseteq \Delta \times \Delta$ zuweist.

Semantik II

Eine Interpretation erweitert sich induktiv:

$$
\begin{aligned}
\top^{\mathcal{I}} &= \Delta \\
\bot^{\mathcal{I}} &= \emptyset \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \Delta \setminus C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \{x \in \Delta \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} &= \{x \in \Delta \mid \forall y : (x,y) \in R^{\mathcal{I}} \Longrightarrow y \in C^{\mathcal{I}}\}
\end{aligned}
$$

Damit ergibt sich die Gültigkeit folgendermaßen:

- Ein Konzept C wird bzgl. des Modells $\mathcal{I}$ von D *subsumiert* (notiert $C \sqsubseteq D$), falls $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ gilt.
- Die Konzepte C und D sind bzgl. des Modells $\mathcal{I}$ *äquivalent* (notiert $C \equiv D$), falls $C^{\mathcal{I}} = D^{\mathcal{I}}$ gilt.

Semantik III

- Ein Konzept heißt *erfüllbar* bzgl. des Modells $\mathcal{I}$, falls $C^{\mathcal{I}}$ nicht leer ist.

Komplexität

Subsumption und Erfüllbarkeit stehen in enger Verbindung wie das folgende Theorem zeigt (Baader u. a., 2003, Theorem 2.13).

Theorem

Seien C und D Konzepte, dann gilt:

- *C wird von D subsumiert, gdw. $C \sqcap \neg D$ unerfüllbar ist.*
- *C und D sind äquivalent, gdw. $C \sqcap \neg D$ und $\neg C \sqcap D$ unerfüllbar sind.*
- *C und D sind disjunkt, gdw. $C \sqcap D$ unerfüllbar ist.*

Schmidt-Schauß und Smolka (1991) zeigten die Verbindungen der Beschreibungslogik zum Erfüllbarkeitsproblem der multimodalen Logik *K*, dessen Erfüllbarkeitsproblem als PSPACE-hart bekannt ist.

Theorem

Das Erfüllbarkeitsproblem für $\mathcal{ALC}$ ist PSPACE-hart.

Konzepte in MEL I

- Die Membership Equational Logic ist reich genug, um bereits Typkonstruktionen wie Schnitt oder Vereinigung darzustellen.
- Wir vergleichen im folgenden das Verhältnis der memberships zu der Subsumptionsbeziehung der Beschreibungslogik.
- Wir können die elementaren Konzepte direkt als Teilsorten übernehmen.
- Die Rollen R betrachten wir als Prädikate $R(x, y)$, d.h. als Abbildungen in die Sorte BOOL.

Konzepte in MEL II

- Betrachten wir die Formelterme als Sortenbezeichner, dann können wir einige Formel direkt in MEL einbetten.

$$
\begin{array}{rcrcl}
\phi(C \sqcap D) & := & \forall x : (x \in C \wedge x \in D) & \Longrightarrow & x \in (C \sqcap D) \\
\phi(C \sqcup D) & := & \forall x : (x \in C \vee x \in D) & \Longrightarrow & x \in (C \sqcup D) \\
\phi(\top) & := & \forall x : \textsc{True} & \Longrightarrow & x \in \top \\
\phi(\bot) & := & \forall x : x \in \bot & \Longrightarrow & \textsc{False} \\
\phi(\exists R.C) & := & \forall x, y : (R(x, y) \wedge y \in C) & \Longrightarrow & x \in (\exists R.C)
\end{array}
$$

- Jede Subsortenbeziehung $x \in C \Longrightarrow x \in D$ spiegelt dabei die Subsumptionsbeziehung $C \sqsubseteq D$ der Konzeptdefinition wieder.
- Die Konzeptebeschreibungen $(\neg C)$ und $\forall R.C$ sind dagegen nicht so ohne weiteres darstellbar, da man in MEL keine negierten Eigenschaften ausdrücken kann.

Konzepte in MEL III

- Folgende Darstellung der Negation sowie der Allquantifizierung ergeben keine Axiome:

$$\begin{array}{rcrcl} \phi(\neg C) & := & \forall x : (\neg x \in C) & \Longrightarrow & x \in (\neg C) \\ \phi(\forall R.C) & := & \forall x, y : (\neg R(x,y) \vee y \in C) & \Longrightarrow & x \in (\forall R.C) \end{array}$$

- Dass die Negation nicht ausdrückbar ist, ergibt sich auch aus der Tatsache, dass MEL äquivalent zur Hornklausellogik mit Gleichheit ist und in Hornlogik Negation auch gesondert zu behandeln ist.
- Dies ist die in Prolog bekannte *negation as failure*-Semantik.

Rollenstrukturen I

- Rollen sind zunächst einmal nur Namen, deren Bedeutung sich erst später durch ihre Verknüpfung mit den Dienstnetzen ergeben.
- Sei *Rol* eine Menge von Rollen.
- Wir gehen im folgenden davon aus, dass die Menge *Rol* eine spezielle Rolle r_* enthält, die wir nur für formale Konstruktionen benötigen.
- Nichtleere Teilmengen $R \subseteq Rol$ werden als *Rollenprofile* (kurz: Profile) bezeichnet.
- Die einelementigen Profile $\{r\}$ mit $r \in Rol$ werden mit der Rolle r identifiziert.

Rollenstrukturen II

Definition

Sei *Rol* eine Menge von Rollen, die die spezielle Rolle r_* enthält Die *Rollenstruktur* über *Rol* ist die durch Mengeninklusion partiell geordnete Menge $\mathcal{R} := 2^{Rol} \setminus \{\emptyset\}$.

- Rollenprofile können durch Vereinigung aggregiert werden.
- Sie sind durch Mengeninklusion partiell geordnet:
- Gilt $R_1 \subseteq R_2$ für zwei Rollenprofile $R_1, R_2 \in 2^{Rol}$, so ist R_2 umfassender als R_1, bzw. R_1 ist spezialisierter als R_2.

Dienstnetze I

- Sei N ein Ablaufnetz.
- Typsystem $\mathcal{S}_C = (\Sigma, X, E)$
- Typsystem formalisiert Syntax der Datenobjekte und die Subtypisierung $\Longrightarrow$ Agentenkommunikationssprache
- Die Farbe einer Marke ist ein Term einer Teilsorte s.
- Jede Farbsorte besitzt ihre eigenen Operatoren und Gleichungen in $\mathcal{S}_C$.
- Sei $A = [\![\mathcal{S}_C]\!]$ eine Theorie zur Datentypspezifikation $\mathcal{S}_C$.
- Jede Stelle eines Dienstnetzes ist durch die Abbildung d typisiert, d.h. alle Marken auf p sind vom Typ $d(p)$.
- Kanten algebraische Netze werden mit Termen beschriftet, die Variablen enthalten können.
- Der Ausdruck $W(f)$ ist dabei von der Sorte $d(p)$, d.h. $W(f) \in \mathbb{T}^{d(p)}_{\Sigma,E}(X)$.

Dienstnetze II

- Eine Markierung $M : P \rightarrow MS(\llbracket K \rrbracket_A)$ des Netzes ist eine Abbildung, die jeder Stelle einen Wert $M(p) \in MS(\llbracket d(p) \rrbracket_A)$ in der Algebra A zuweist, d.h. eine Multimenge über $\llbracket d(p) \rrbracket_A$.
- Jeder Markierung M von D ordnen wir eine Projektionsmarkierung $\pi(M)$ zu, indem wir die Farbe der Marke vergessen und nur noch die Anzahl der Marken zählen:

$$\pi(M)(p) = |M(p)| \tag{2}$$

- Als Besonderheit der Dienstnetze wird jeder Transition t die Rolle $r(t) \in Rol$ zugewiesen.
- Dies bedeutet, dass zur Ausführung eines Tasks ein Agent, der diese Rolle inne hat, notwendig ist.
- Wir erweitern die Abbildung r auf Transitionsmengen: $R(T') = \{r(t) \mid t \in T'\}$

Dienstnetze III

- Wir fordern, dass in Dienstnetzen die Rollen konfliktfrei verbunden sind, d.h. Vor- und Nachbereich einer Stelle enthalten jeweils Transitionen mit gleicher Rollenzuweisung:

$$\forall p \in P : |R({}^\bullet p)|, |R(p^\bullet)| \leq 1 \tag{3}$$

- Eine Stelle p heißt *Kommunikationskanal*, wenn sie Transitionen unterschiedlicher Rollen verbindet, d.h. wenn $R({}^\bullet p) \neq R(p^\bullet)$ gilt.
- Die Menge der Kommunikationskanäle zum Paar $(r_1, r_2) \in Rol^2$ ist:

$$P_{KK}(r_1, r_2) := \{p \mid R({}^\bullet p) = \{r_1\} \wedge R(p^\bullet) = \{r_2\}\} \tag{4}$$

Die Menge der Kommunikationskanäle ist dann:

$$P_{KK}(D) := \bigcup_{r_1, r_2 \in R(D), r_1 \neq r_2} P_{KK}(r_1, r_2) \tag{5}$$

Also gilt $p \in P_{KK}(D)$ genau dann, wenn $R({}^\bullet p) \neq R(p^\bullet)$ gilt.

Dienstnetze IV

- Für Dienstnetze wird gefordert, dass jede Rollenaufteilung durch einen Kommunikationskanal verbunden wird:

$$\begin{aligned} &\forall R \subseteq R(D) : R \neq \emptyset \\ &\Longrightarrow \exists r_1 \in R, r_2 \in R(D) \setminus R : P_{KK}(r_1, r_2) \cup P_{KK}(r_2, r_1) \neq \emptyset \end{aligned} \tag{6}$$

Dienstnetze V

Definition

Ein *Dienstnetz* $D = (N, [\![\mathcal{S}_C]\!], d, r, W, G, M_0)$ ist:

1. $[\![\mathcal{S}_C]\!]$ ist eine Theorie zur Datentypspezifikation $\mathcal{S}_C$.
2. $N = (P, T, F)$ ist ein Ablaufnetz.
3. $d : P \to \bigcup_{k \in K} S_k$ ist die Stellentypisierung.
4. $r : T \to \mathit{Rol}$ ist die Rollenzuweisung, die (3) und (6) erfüllt.
5. $W : F \to \mathbb{T}_{\Sigma,E}(X)$ ist die Kanteninschrift.
6. $G : T \to \mathit{MEL}_{\mathcal{S}}$ ist die Aktivierungsbedingung.
7. M_0 ist eine Initialmarkierung mit $\pi(M_0) = m_0$, wobei m_0 die kanonische Initialmarkierung von N ist.

Dann ist $\pi(D) := N$ das *D zugrundliegende Ablaufnetz*.

Schaltregel

Die Schaltregel ergibt sich mit Hilfe von Variablenbindungen. Jede *Bindung* $\alpha : X \to [\![K]\!]_A$ der Variablen, die das Aktivierungsprädikat $G(t)$ der Transition t erfüllt, erzeugt einen *Schaltmodus*.

Definition

Eine Transition t unter der Variablenzuweisung $\alpha : X \to [\![K]\!]_A$ heißt in M *aktiviert*, gdw. die Schaltbedingung erfüllt ist: $A, \alpha \models G(t)$ und für alle $p \in P$ auch $M(p) \geq \overline{\alpha}(W(p,t))$ gilt.

Das Schalten der Transition t unter der Belegung α wird als $M \xrightarrow{t,\alpha} M'$ notiert, wobei sich die *Nachfolgemarkierung* für jede Stelle $p \in P$ wie folgt ergibt:

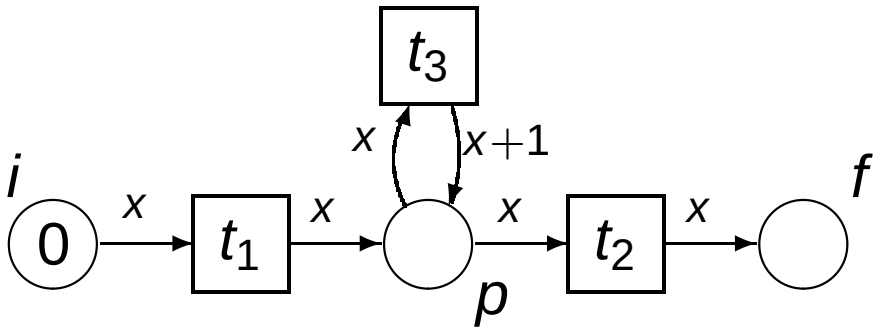
$$M'(p) = M(p) - \overline{\alpha}(W(p,t)) + \overline{\alpha}(W(t,p))$$

Beschränktheit I

Definition

Ein Dienstnetz heißt *beschränkt*, wenn die Menge der erreichbaren Markierungen endlich ist.

Man beachte, dass die Anzahl der Marken auf jeder Stelle in der Anzahl beschränkt sein kann, ohne dass das Netz selbst beschränkt sein muss.



Projektionen I

Jeder Schaltvorgang eines algebraischen Netzes ist auch in den Projektionen möglich.

Theorem

Sei $D = (N, A, d, r, W, G, M_0)$ *ein Dienstnetz, dann gilt für die Projektionen:*

$$M \xrightarrow[D]{t,\alpha} M' \Longrightarrow \pi(M) \xrightarrow[\pi(D)]{t} \pi(M')$$

und

$$\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$$

Projektionen II

Sei $M \xrightarrow[D]{t,\alpha}$. Dies ist nach Definition erfüllt, wenn $A, \alpha \models G(t)$ und wenn für alle $p \in P$ gilt:

$$
\begin{aligned}
& M(p) \geq \overline{\alpha}(W(p,t)) \\
\iff \quad & \forall c \in [\![d(p)]\!]_A : M(p)(c) \geq \overline{\alpha}(W(p,t))(c) \\
\implies \quad & \sum_{c \in [\![d(p)]\!]_A} M(p)(c) \geq \sum_{c \in [\![d(p)]\!]_A} \overline{\alpha}(W(p,t))(c) \\
\iff \quad & |M(p)| \geq |\overline{\alpha}(W(p,t))| \\
\iff \quad & \pi(M)(p) \geq F(p,t)
\end{aligned}
$$

Also ist t auch in N aktiviert. Bei der Umformung haben wir $\pi(M(p)) = \pi(M)(p)$ und die Eigenschaft $|\overline{\alpha}(W(p,t))| = F(p,t)$ ausgenutzt. Für die Nachfolgemarkierung gilt:

$$
\begin{aligned}
& M'(p) = M(p) - \overline{\alpha}(W(p,t)) + \overline{\alpha}(W(t,p)) \\
\iff \quad & \forall c \in [\![d(p)]\!]_A : M'(p)(c) = M(p)(c) - \overline{\alpha}(W(p,t))(c) + \overline{\alpha}(W(t,p \\
\implies \quad & \sum_{c \in [\![d(p)]\!]_A} M'(p)(c) = \sum_{c \in [\![d(p)]\!]_A} M(p)(c) - \overline{\alpha}(W(p,t))(c) + \overline{\alpha}(W(t, \\
\iff \quad & |M'(p)| = |M(p)| - F(p,t) + F(t,p) \\
\iff \quad & \pi(M')(p) = \pi(M)(p) - F(p,t) + F(t,p)
\end{aligned}
$$

Projektionen III

Also ist $\pi(M')$ die Nachfolgemarkierung von $\pi(M)$.

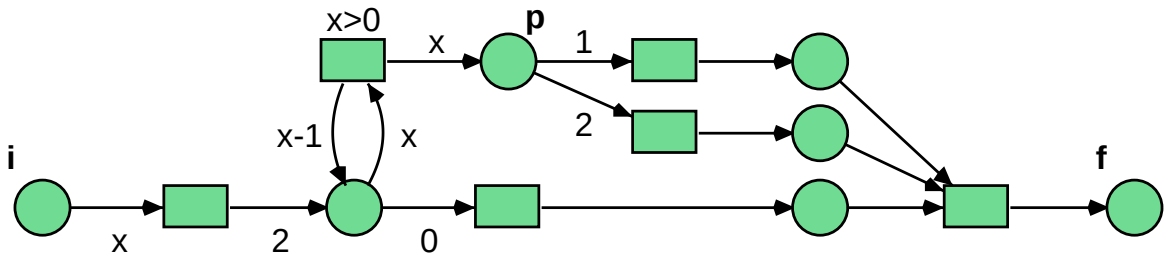
Die zweite Aussage folgt induktiv aus der ersten.

Beschränktheit der Projektion I

- Aus der Eigenschaft $\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$ können wir leider nicht schließen, dass aus der Beschränktheit von $\pi(D)$ auch die Beschränktheit von D folgt.
- Aus der Beschränktheit von $\pi(D)$ folgt die Endlichkeit von $RS(\pi(D), \pi(M_0))$ und
- aufgrund der Inklusion auch die Endlichkeit der Projektion $\pi(RS(D, M_0))$,
- jedoch impliziert letzteres nicht die Endlichkeit von $RS(D, M_0)$ selbst.
- So ist das obige Netz unbeschränkt, da die Stelle p zwar maximal eine Marke enthält, so dass p in $\pi(D)$ beschränkt ist, die Marke aber eine beliebig große Zahl als Wert haben kann.

Beschränktheit der Projektion II

- Die Umkehrung gilt auch nicht, denn folgendes beschränktes Dienstnetz D besitzt eine unbeschränkte Projektion $\pi(D)$.



Theorem

Sind die Farbdomänen alle endlich, so folgt aus der Beschränktheit von $\pi(D)$ auch die von D.

Aus der Beschränktheit von $\pi(D)$ folgt die Endlichkeit von $RS(\pi(D), \pi(M_0))$.

Es gibt also nur endliche viele erreichbare Markierungen m.

Beschränktheit der Projektion III

Nach Theorem 16 gilt $\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$ und $\pi(RS(D, M_0))$ ist eine endliche Menge.

Da alle $[\![d(p)]\!]_A$ endlich sind, gibt es zu jedem m nur endlich viele Verteilungen der $M(p)(c)$, so dass $\pi(M) = \sum_{c \in [\![d(p)]\!]_A} M(p)(c) = m$ ist.

Also ist für alle m die Menge $\{M \mid \pi(M) = m\}$ endlich.

Also gibt es zu jedem $m \in \pi(RS(D, M_0))$ nur endliche viele Elemente in $RS(D, M_0)$, so auch $RS(D, M_0)$ endlich ist.

Also ist D beschränkt.

Prozesse von Dienstnetzen I

Ein Prozess ist ein Kausalnetz $K = (B, E, \lessdot)$ zusammen mit

1. $\phi^P : B \to P$
2. $\phi^T : E \to T$
3. $\phi^A : B \to A$ ist die Farbe der Bedingungen.
4. $\phi^\alpha : E \to (X \to A)$ ist die Bindung der Ereignisse.

Jeder Bedingungsmengen $Q \subseteq B$ wird eine Multimenge von Farben zugewiesen:

$$\phi^A(Q)(p)(c) := |\{b \in Q \mid \phi^B(b) = p \wedge \phi^A(b) = c\}| \tag{7}$$

Damit ein Ereignis $e \in E$ das Schalten der Transition $t = \phi^T(e)$ wiederspiegelt, muss dessen Guard $G(t)$ in der Bindung $\alpha = \phi^\alpha(e)$ gültig sein, und die Multimengen $\phi^A(e^\bullet)(p)$ und $\phi^A({}^\bullet e)(p)$ müssen für

Prozesse von Dienstnetzen II

alle Stellen p den Kantengewichtungen $\alpha(W(p,t))$ bzw. $\phi^{\alpha}(e)(W(t,p))$ entsprechen.

Prozesse von Dienstnetzen III

Definition

Ein Prozess des Dienstnetzes $D = (N, A, d, r, W, G, M_0)$ ist das Tupel (K, ϕ) mit $K = (B, E, \prec)$ und $\phi = (\phi^P, \phi^T, \phi^A, \phi^\alpha)$, wobei gilt:

1. (K, ϕ^P, ϕ^T) ist ein Prozess von N.
2. Der Prozess beschreibt den Anfangszustand: $\phi^A(^\circ K) = M_0$
3. $\phi^A : B \to A$ weist jeder Bedingung einen Wert der Algebra zu.
4. $\phi^\alpha : E \to (X \to A)$ weist jedem Ereignis seine Bindung zu.
5. Die Netzstruktur beschreibt das Schalten von $(\phi^T(e), \phi^\alpha)$:

$$\begin{aligned} \forall e \in E: \quad & A, \phi^\alpha(e) \models G(\phi^T(e)) \wedge \\ & \forall p \in {}^\bullet\phi^T(e) : \phi^A({}^\bullet e)(p) = \phi^\alpha(e)(W(p, \phi^T(e))) \wedge \\ & \forall p \in \phi^T(e)^\bullet : \phi^A(e^\bullet)(p) = \phi^\alpha(e)(W(\phi^T(e), p)) \end{aligned}$$

Die Menge aller Prozesse von D wird mit $Proc(D)$ bezeichnet.

Korrektheit von Dienstnetzen

Definition

Ein Dienstnetz D heißt *korrekt*, wenn gilt:

1. Termination: Wird die finale Markierung übertroffen, so wird sie exakt erreicht.

$$\forall M \in RS(D, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f$$

2. Fortsetzbarkeit: Von jeder aus der initialen Markierung erreichbaren Markierung ist die finale Markierung erreichbar.

$$\forall M \in RS(D, M_0) : \exists M' \in RS(D, M) : \pi(M') = m_f$$

3. Aktivierbarkeit: Jede beliebige Transition ist für eine Anfangsmarkierung aktivierbar.

$$\forall t \in T : \exists M_0 : \pi(M_0) = m_0 \wedge \exists M \in RS(D, M_0) : M \xrightarrow{t}$$

Relation von Dienst- zu Workflownetzen I

Definition

Ein Workflow-Netz ist ein Petrinetz $N = (P, T, F)$, das genau eine Stelle $i \in P$ mit ${}^\bullet i = \emptyset$ und genau eine Stelle $f \in P$ mit $f^\bullet = \emptyset$ besitzt und für das jeder Knoten $n \in (P \cup T)$ auf einem Pfad zwischen i und f liegt.

Relation von Dienst- zu Workflownetzen II

Definition

Ein Workflow-Netz $N = (P, T, F)$ heißt *korrekt*, wenn gilt:

1. Termination: Wird die finale Markierung übertroffen, so wird sie exakt erreicht.

$$\forall m \in RS(N, m_0) : m_f \leq m \Longrightarrow m_f = m$$

2. Fortsetzbarkeit: Von jeder aus der initialen Markierung erreichbaren Markierung ist die finale Markierung erreichbar.

$$\forall m \in RS(N, m_0) : m_f \in RS(N, m)$$

3. Aktivierbarkeit: Jede beliebige Transition ist aktivierbar.

$$\forall t \in T : \exists m \in RS(N, m_0) : m \xrightarrow{t}$$

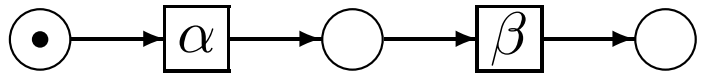
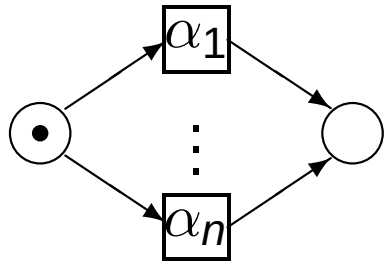
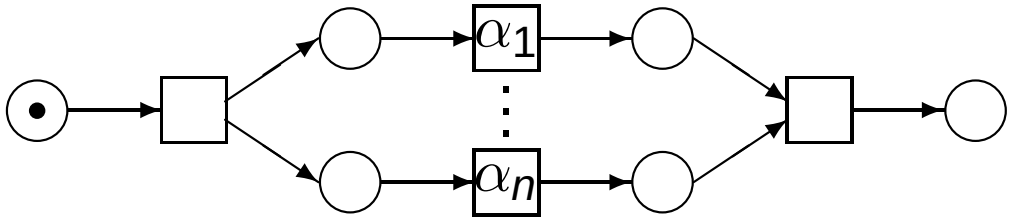
Relation von Dienst- zu Workflownetzen III

- Korrektheit eines Workflows erfordert, dass von jeder erreichbaren Markierung exakt in m_f terminiert werden kann. Somit ist m_f die einzige Verklemmung.
- Für potentiell terminierende Workflows ist die Korrektheitbedingung dahingehend abgeschwächt, dass die finale Markierung von der initalen aus erreichbar ist, nicht jedoch unbedingt von jeder von der initalen aus erreichbaren Markierung.
- Es ist dann im allgemeinen notwendig, das Netz zu steuern (s.u.), um Verklemmungen zu vermeiden.

schwache Korrektheit : Termination und Fortsetzbarkeit:

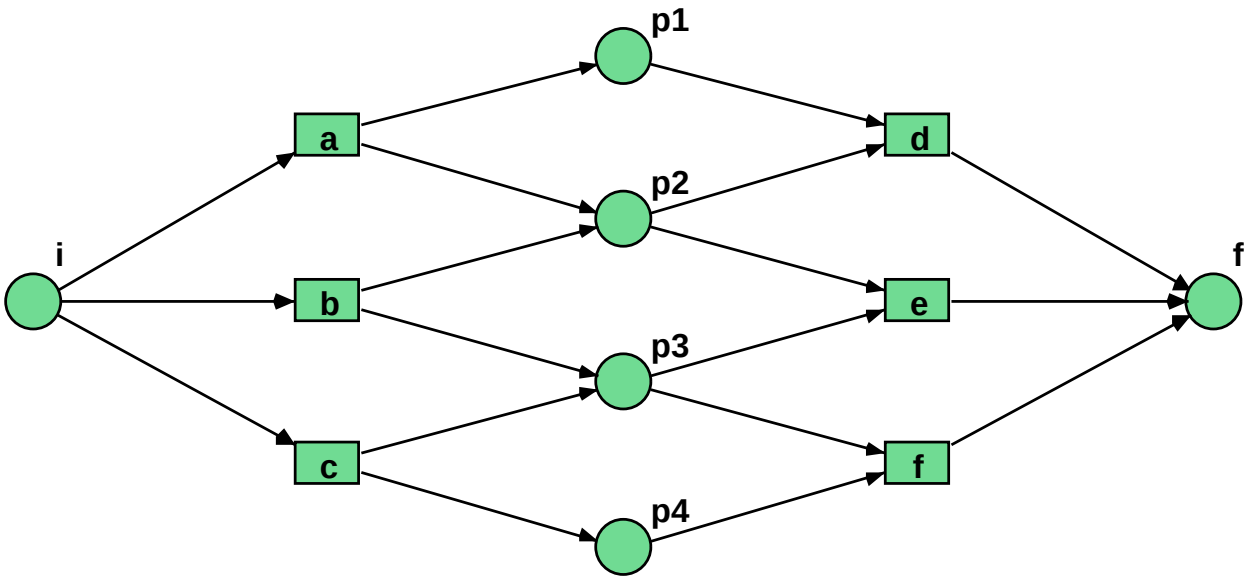
$$\forall m \in RS(N, m_0) : (m_f \in RS(N, m)) \wedge (m_f \leq m \Longrightarrow m_f = m)$$

Verfeinerung I

Operator	Petrinetz
Sequenz $\alpha \cdot \beta$	
Alternative (XOR) $\sum_{i=1}^{n} \alpha_i$	
Parallelität (AND) $\Big\|_{i=1}^{n} \alpha_i$	

Monotonie der Korrektheit I

Korrektheit ist nicht monoton: Ein Netz, das korrekt für $m_0 = i$ ist, muss dies nicht für eine andere Markierung (z.B. $m = 2i$) sein.



Neustart

- Ein Workflow-Netz N hat in der finalen Markierung einen Deadlock.
- Neustart: Transition t_N legt Marken in f wieder auf i zurück.

Definition

Der Abschluß eines Workflow-Netz $N = (P, T, F)$ ist das Petrinetz $N^* = (P, T \cup \{t_N\}, F \cup \{(i, t_N), (t_N, o)\})$, wobei $t_N \notin (P \cup T)$ gilt.

Theorem

Ein Workflow-Netz N ist genau dann korrekt, wenn sein Abschluß N^* in der Initialmarkierung m_0 beschränkt und lebendig ist.

Reduktion von Ablaufnetzen auf Workflows I

- Workflow-Netze sind spezielle Ablaufnetze: $|{}^\circ N| = |N^\circ| = 1$
- jedes Ablaufnetz kann durch das Hinzufügen einer intialen und einer finalen Transition in ein Workflow-Netz überführt werden.
- Ebenso Dienstnetze:
- Die Rolle der neuen Transitionen ist beliebig und wird auf die Spezialrolle r_* festgelegt.

Definition

Sei $N = (P, T, F)$ ein Ablaufnetz und sei $\{i, f\}$, $\{t_i, t_f\}$, P und T paarweise disjunkt, dann ist $\widetilde{N}$ das Netz:

$$\widetilde{N} := (P \uplus \{i, f\}, T \uplus \{t_i, t_f\}, \widetilde{F})$$

mit $\widetilde{F} = F \cup \{(i, t_i), (t_f, f)\} \cup (\{t_i\} \times {}^\circ N) \cup (N^\circ \times \{t_f\}))$.

Reduktion von Ablaufnetzen auf Workflows II

Definition

Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz und sei ${}^\circ N = \{p_1, \ldots, p_n\}$ und $N^\circ = \{p'_1, \ldots, p'_{n'}\}$, dann ist der Abschluß von D definiert als der Dienst:

$$\widetilde{D} = (\widetilde{N}, A, \tilde{d}, \tilde{r}, \tilde{W}, \tilde{d}, \tilde{M}_0)$$

Reduktion von Ablaufnetzen auf Workflows III

Dabei erweitern wir die Abbildungen folgendermaßen:

$$
\begin{aligned}
\tilde{d}(i) &= d(p_1) \times \cdots \times d(p_n) \\
\tilde{d}(f) &= d(p'_1) \times \cdots \times d(p'_{n'}) \\
\tilde{r}(t_i) &= \tilde{r}(t_f) = r_* \\
\tilde{W}(p, t_i) &= x_p = \tilde{W}(t_f, p') \\
\tilde{W}(i, t_i) &= (x_{p_1}, \ldots, x_{p_n}) \\
\tilde{W}(t_f, f) &= (x_{p'_1}, \ldots, x_{p'_{n'}}) \\
\tilde{M}_0(i) &= (M_0(p_1), \ldots, M_0(p_n))
\end{aligned}
$$

Mit Hilfe dieser Konstruktion kann jedes Dienstnetz auf ein Workflownetz reduziert werden.

Reduktion von Ablaufnetzen auf Workflows IV

Theorem

Ist N ein Ablaufnetz, dann ist $\widetilde{N}$ ein Workflownetz.
Ist D ein Dienstnetz, dann ist $\pi(\widetilde{D})$ ein Workflownetz.

Beweis.

Folgt direkt aus den Definitionen 1, 13 und 20. □

Reduktion von Ablaufnetzen auf Workflows I

Im Vergleich mit D hat $\widetilde{D}$ nur zwei Gruppen zusätzlicher erreichbare Markierungen, nämlich die Initialmarkierung M_0 und die neuen Finalmarkierungen M_f, die $\pi(M_f) = m_f$ erfüllen.

Analog hat $\widetilde{N}$ als korrekte Workflow im Vergeleich mit $N = \pi(D)$ nur m_i und m_f als zusätzliche Markierungen. Es ist daher nicht verwunderlich, dass beide bezüglich der Korrektheit äquivalent sind:

Theorem

Ein Dienstnetz D ist genau dann korrekt, wenn $\widetilde{D}$ dies ist.

Zunächst einmal stellen wir fest, dass alle Schaltfolgen in $\widetilde{D}$ die folgende Form haben:

$$\widetilde{M}_0 \xrightarrow[\widetilde{D}]{t_i} M_0 \xrightarrow[D]{w} M \quad \text{bzw.} \quad \widetilde{M}_0 \xrightarrow[\widetilde{D}]{t_i} M_0 \xrightarrow[D]{w} M \xrightarrow[\widetilde{D}]{t_f} M', \text{ falls } \pi(M) = m_f$$

Reduktion von Ablaufnetzen auf Workflows II

Also gilt: $M \in RS(D, M_0)$ impliziert $M \in RS(\widetilde{D}, \tilde{M}_0)$.

Für Korrektheit ist die Äquivalenz der Termination und Fortsetzbarkeit in D und $\widetilde{D}$ zu zeigen.

$\Longrightarrow$ Es gelte in D Termination und Fortsetzbarkeit:

$$\forall M \in RS(D, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f$$
$$\forall M \in RS(D, M_0) : \exists M' \in RS(D, M) : \pi(M') = m_f$$

Dann ist für $\widetilde{D}$ analog zu zeigen:

$$\forall M' \in RS(\widetilde{D}, \widetilde{M}_0) : \pi(M') \geq \tilde{m}_f \Longrightarrow \pi(M') = \tilde{m}_f$$
$$\forall M'' \in RS(\widetilde{D}, \widetilde{M}_0) : \exists M''' \in RS(\widetilde{D}, M'') : \pi(M''') = \tilde{m}_f$$

Reduktion von Ablaufnetzen auf Workflows III

1. Termination: Sei $M' \in RS(\widetilde{D}, \widetilde{M}_0)$ eine beliebige Markierung, die $\pi(M') \geq \tilde{m}_f$ erfüllt.
   Jetzt impliziert $\pi(M') \geq \tilde{m}_f$, dass t_f geschaltet hat.
   Die Vorgängermarkierung M'' von M' ist aber in D final, erfüllt also $\pi(M'') \geq m_f$, woraus $\pi(M'') = m_f$ folgt, was wiederum $\pi(M') = \tilde{m}_f$ impliziert.
   Termination gilt also.
2. Fortsetzbarkeit: Ist M in D erreichbar, so auch in $\widetilde{D}$.
   Zu M existiert $M' \in RS(D, M)$ mit $\pi(M') = m_f$, das auch in $\widetilde{D}$ erreichbar ist.
   Schalten wir dann t_f in $\widetilde{D}$, so haben wir $\tilde{m}_f$ erreicht.

$\Longleftarrow$ Sei nun umgekehrt in $\widetilde{D}$ Termination und Fortsetzbarkeit gültig, dann ist dies auch für D zu zeigen:

1. Termination: Gilt $\pi(M') \geq \tilde{m}_f \Longrightarrow \pi(M') = \tilde{m}_f$, dann hat t_f geschaltet und die Vorgängermarkierung wurde exakt erreicht.

Reduktion von Ablaufnetzen auf Workflows IV

2. Fortsetzbarkeit: Ist M'' in $\widetilde{D}$ erreichbar, so markiert entweder M'' nur Stellen von D oder $M'' = \widetilde{M}_0$ oder M'' markiert auch f.
   Im ersten Fall kann stets ein (M''') mit $\pi(M''') = \tilde{m}_f$ erreicht werden, was impliziert, dass zuvor ein (M') mit $\pi(M') = m_f$ erreicht wurde.
   Im zweiten Fall kann M_0 erreicht werden und davon ein (M') mit $\pi(M') = m_f$.
   Im dritten Fall darf die Markierung M''' wegen der Gültigkeit der Terminationsbedingung sogar nur f markieren, so dass $\pi(M'') = \tilde{m}_f$ gilt, was impliziert, dass für die Vorgängermarkierung M'''' von M'' auch $\pi(M'''') = m_f$ gilt.

Beschränktheit/Lebendigkeit des Abschlusses $\widetilde{D}^*$ I

Analog zu den Workflownetzen gilt:

Theorem

Ist $\widetilde{D}^*$ beschränkt und lebendig, dann ist das Dienstnetz D korrekt.

Ist $\widetilde{D}^*$ in M_0 beschränkt und lebendig, dann ist insbesondere t_N lebendig, und wir erreichen von jeder Markierung aus stets eine Markierung M, die t_N aktiviert, also: $\pi(M) \geq m_f$.

Dann muss aber auch $\pi(M) = m_f$ gelten, denn ansonsten könnte das Netz nicht beschränkt sein.

Also gilt Fortsetzbarkeit und Termination.

Beachten wir, dass t_N jede beliebige Anfangsmarkierung M_0' mit $\pi(M_0) = m_0$ generiert werden kann, so können wir aus der

Beschränktheit/Lebendigkeit des Abschlusses $\widetilde{D}^*$ II

Lebendigkeit folgern, dass es für jede Transition t eine Anfangsmarkierung gibt, so dass t aktiviert werden kann.

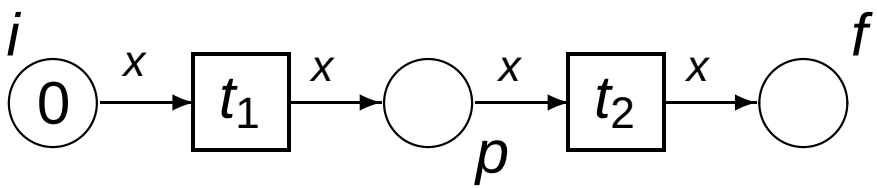
Also gilt die Aktivierungsbedingung.

Beschränktheit I

Man beachte, dass die Beschränktheit für $\widetilde{D}^*$ gefordert wird und nicht für $\widetilde{D}$, denn es ist selbst für korrekte Netze nicht möglich, von der Beschränktheit von D auf die von D^* zu schließen.

Dies gilt sogar in struktureller Formulierung: Sei $\widetilde{D}$ ein korrektes Dienstnetz, dass für alle M_0 mit $\pi(M_0) = m_0$ beschränkt ist, dann gilt im allgemeinen nicht auch noch, dass auch $\widetilde{D}^*$ beschränkt ist.

Korrektes, für alle M_0 beschränktes Dienstnetz, für das $\widetilde{D}^*$ nicht beschränkt ist (die Stelle p kann jeden beliebigen Wert annehmen):



Korrektheit $\Longrightarrow$ Beschränktheit? I

Die Umkehrung gilt nur in folgender eingeschränkter Form.

Theorem

Wenn das Dienstnetz D strukturell korrekt ist, dann ist $\widetilde{D}^*$ für alle Markierungen M_0 mit $\pi(M_0) = m_0$ lebendig.

Betrachten wir ein strukturell korrektes Dienstnetz in einer Markierungen M_0 mit $\pi(M_0) = m_0$.

Ist das Netz strukturell korrekt, so ist mit der Fortsetzbarkeitseigenschaft von jeder erreichbaren Markierung $M \in RS(D, M_0)$ die finale Markierung M_f mit $\pi(M_f) = m_f$ erreichbar – unabhängig von der Wahl für M_0.

Also ist t_N aktivierbar.

Korrektheit $\Longrightarrow$ Beschränktheit? II

Um zu zeigen, dass jede Transition t aktiviert werden kann, nutzen wir aus, dass t_N jede beliebige Anfangsmarkierung M_0' mit $\pi(M_0') = m_0$ generieren kann, und da D strukturell korrekt ist, gilt die Aktivierungsbedingung auch für M_0'.

Die Aktivierungsbedingung garantiert, dass jede Transition t für mindestens eine Anfangsmarkierung $M_{0,t}$ aktiviert werden kann.

Da diese von t_N generiert werden kann, ist $\widetilde{D}^*$ lebendig.

Bemerkung: Wenn das Dienstnetz D korrekt oder strukturell korrekt ist, dann muss $\widetilde{D}^*$ jedoch nicht beschränkt sein, wie das unbeschränkte, aber strukturell korrekte Netz (oben) zeigt.

Betrachten wir nur beschränkte Dienstnetze, so können wir Theorem 28 und 29 folgendermaßen zusammenfassen:

Theorem

Sei $\widetilde{D}$ ein Dienstnetz, das für alle M_0 mit $\pi(M_0) = m_0$ beschränkt ist, dann gilt: Das Dienstnetz D ist genau dann korrekt, wenn $\widetilde{D}^*$ für alle Markierungen M_0 mit $\pi(M_0) = m_0$ lebendig ist.

Das Dienstnetz $\widetilde{D}$ ist genau dann strukturell korrekt, wenn D dies ist.

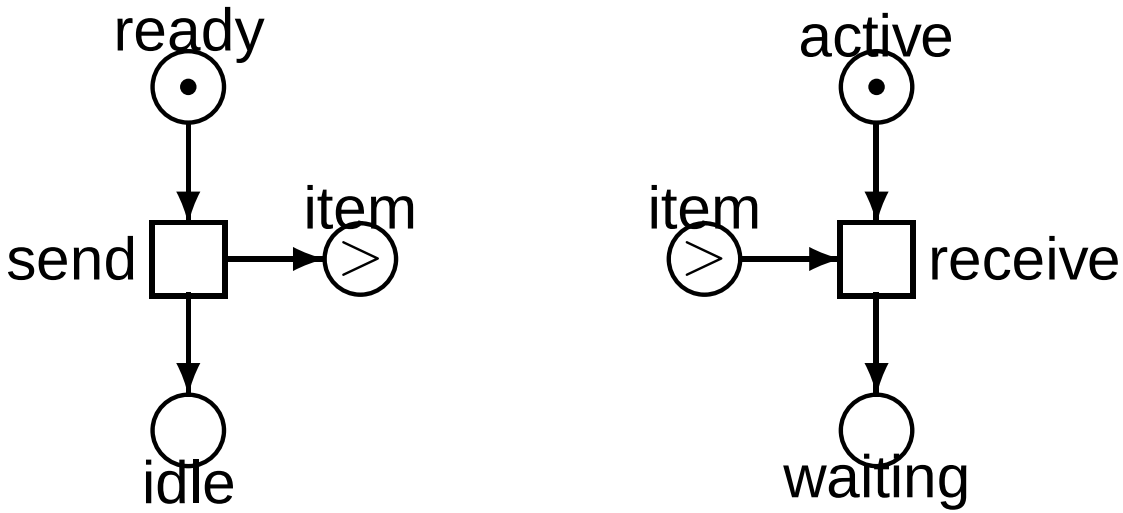
Sei $\widetilde{D}$ für jedes M_0 mit $\pi(M_0) = m_0$ beschränkt.

Wenn $\widetilde{D}$ strukturell korrekt ist, dann ist es für alle M_0 mit $\pi(M_0) = m_0$ korrekt und damit ist $\widetilde{D}^*$ nach Thm. 29 auch für alle M_0 lebendig.

Für jedes M_0 mit $\pi(M_0) = m_0$ folgt aus der Lebendigkeit (zusammen mit der gegebenen Beschränktheit) in M_0 aus Thm. 28 die Korrektheit.

Gilt Lebendigkeit für alle diese M_0, so folgt die Korrektheit für alle M_0, also die strukturelle Korrektheit.

Producer/Consumer-Interaktion



Komposition

Definition

Für zwei einfache Netze $N_i = (P_i, T_i, F_i), i = 1, 2$ ist die Komposition als komponentenweise Vereinigung zu definieren:

$$(N_1 \| N_2) = (P_1 \cup P_2, T_1 \cup T_2, F_1 \cup F_2)$$

Stellenfusion

- Stellenfusion beschreibt eine asynchrone Kopplung der Systeme.
- Stellenfusion modelliert den gemeinsamen Zugriff zweier Komponenten auf ein geteiltes Objekt, z.B. auf einen Nachrichtenpuffer oder eine gemeinsam benutzte Ressource.
- Eine Komposition fügt nur Übergänge zwischen Zuständen hinzu.
- Die Fortsetzbarkeitseigenschaften der Einzelkomponenten auf das Gesamtsystem.
- Die Eigenschaft der Fortsetzbarkeit ist gleichbedeutend mit der Forderung, dass das Transitionssystem der Komposition $N_1 \| N_2$ in der Einschränkung auf die Aktionen von N_1 das Transitionssystem von N_1 enthält.

Transitionsverschmelzung

- Transitionsverschmelzung beschreibt eine synchrone Aktion.
- Transitionverschmelzung koppelt zwei Systeme eng aneinander.
- Der Raum der erreichbaren Zustände wird sowohl eingeschränkt als auch erweitert.
- Er wird eingeschränkt, indem Zustandsübergänge der Komponente N_1 direkt an einen Übergang von N_2 gebunden werden können. Ist dieser nicht möglich, so wird der Übergang auch in N_1 nicht mehr möglich sein.
- Er wird erweitert, indem N_2 durch Hinzufügen von Transitionen den Erreichbarkeitsgraph von N_1 erweitern kann.

Netzkomponenten

Definition

Eine *Netzkomponente* $\Gamma = (N, m_0, I, O)$ besteht aus einem folgenden Komponenten:

1. $N = (P, T, F)$ ist ein T-schlichtes Petrinetz mit Markierung $m_0 : P \to \mathbb{N}$.
2. $I, O \subseteq P$ sind Mengen an Input- bzw. Output-Schnittstellen mit ${}^\bullet T \cap O = T^\bullet \cap I = \emptyset$ und $m_0(p) = 0$ für alle $p \in I \cup O$.

Eine Komponente heißt *geschlossen*, wenn $I \cup O = \emptyset$ gilt.
Zwei Komponenten Γ_1 und Γ_2 sind komponierbar, wenn $(T_1 \cap T_2) = \emptyset$ und $(P_1 \cap P_2) \subseteq ((I_1 \cap O_2) \cup (I_2 \cap O_1))$ gilt.
Die Komposition zweier komponierbarer Komponenten ist:

$$\Gamma_1 \parallel \Gamma_2 := (N_1 \cup N_2, m_1 + m_2, I, O)$$

mit $I = ((I_1 \setminus O_2) \cup (I_2 \setminus O_1))$ und $O = ((O_1 \setminus I_1) \cup (O_2 \setminus I_1))$.

Komponentenprozesse

Zwei Arten von Ereignissen: intern und extern

Definition

Sei $\Gamma = (N, m_0, I, O)$ eine Komponente mit $N = (P, T, F)$ und $K = (B, E, \lessdot)$ ein vorgängerendliches Kausalnetz, dann ist $(K, \phi : B \to \mathcal{P})$ ein *Prozess* von Γ, falls gilt:

1. Die minimalen Stellen ${}^\circ K$ beschreiben die Anfangsmarkierung: $\phi({}^\circ K)|_P = m_0$.
2. Für jedes Ereignis $e \in E$ gilt eine der beiden Bedingungen:
   1. Interne Aktion: Es existiert eine Transition $t \in T$, so dass $\phi({}^\bullet e) = {}^\bullet t$ und $\phi(e^\bullet) = t^\bullet$.
   2. Externe Aktion: Für alle Stellen $x \in \mathcal{P}$ gilt $\phi({}^\bullet e)(x) \leq 1$ und $\phi(e^\bullet)(x) \leq 1$. Aus $\phi({}^\bullet e)(x) = 1$ folgt $x \in O \cup (\mathcal{P} \setminus P)$ und $\phi(e^\bullet)(x) = 1$ impliziert $x \in I \cup (\mathcal{P} \setminus P)$.

Die Menge aller Prozesse von Γ wird mit $Proc(\Gamma)$ bezeichnet.

Komponentenprozesse

Komposition $\Gamma_1 \parallel \Gamma_2$ entspricht dem Durchschnitt der Einzelverhalten (vgl. Kindler, 1997, Proposition 10):

Theorem

Seien Γ_1, Γ_2 zwei komponierbare Komponenten, dann gilt

$$Proc(\Gamma_1 \parallel \Gamma_2) = Proc(\Gamma_1) \cap Proc(\Gamma_2)$$

$\rightarrow$ kompositionale Semantik

Rollenkomponenten

Die R-Komponente eines Dienstnetzes N ist das Teilnetz, das die R zugeordneten Transitionen und deren umliegende Stellen enthält.

Definition

Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienst mit $N = (P, T, F)$ und sei $R \subseteq R(D)$.

- Die *R-Komponente* von N ist das durch $T_R := r^{-1}(R)$ und $P_R := ({}^\bullet T_R \cup T_R{}^\bullet)$ induzierte Teilnetz von N, das mit $N[R]$ bezeichnet wird.
- Jedes Teilnetz $N' = (P', T', F')$ von N induziert das Dienstnetz:

$$D|_{N'} = (N', A, d|_{P'}, r|_{T'}, W|_{F'}, G|_{T'}, M_0|_{P'}) \tag{8}$$

- Ist $N' = N[R]$ speziell eine Rollenkomponente, dann ist $D[R] := D|_{N'}$ die *R-Komponente* von D.

Vorteil von Rollenkomponenten: Die Umgebung ist klar.

Theorem

Sei D ein Dienst und sei $R_1, R_2 \subseteq R(D)$ mit $R_1 \cap R_2 = \emptyset$, dann gilt:

$$N[R_1 \cup R_2] = N[R_1] \| N[R_2]$$

Da R_1 und R_2 disjunkt sind, sind auch $T_{N[R_1]}$ und $T_{N[R_2]}$ disjunkt.

$$\begin{aligned} T_{N[R_1 \cup R_2]} &= r^{-1}(R_1 \cup R_2) \\ &= r^{-1}(R_1) \cup r^{-1}(R_2) \\ &= T_{N[R_1]} \cup T_{N[R_2]} \\ &= T_{N[R_1] \| N[R_2]} \end{aligned}$$

Außerdem ist:

$$\begin{aligned} P_{N[R_1 \cup R_2]} &= {}^\bullet T_{N[R_1 \cup R_2]} \cup T_{N[R_1 \cup R_2]}{}^\bullet \\ &= ({}^\bullet T_{N[R_1]} \cup {}^\bullet T_{N[R_2]}) \cup (T_{N[R_1]}{}^\bullet \cup T_{N[R_2]}{}^\bullet) \\ &= ({}^\bullet T_{N[R_1]} \cup T_{N[R_1]}{}^\bullet) \cup ({}^\bullet T_{N[R_2]} \cup T_{N[R_2]}{}^\bullet) \\ &= P_{N[R_1]} \cup P_{N[R_2]} = P_{N[R_1] \| N[R_2]} \end{aligned}$$

Da die Stellen und Transition übereinstimmen, generieren $N[R_1 \cup R_2]$ und $N[R_1] \| N[R_2]$ die gleichen Teilnetze. q.e.d.

Vollständige Rekonstruktion I

Theorem

Sei D ein Dienstnetz und $N = \pi(D)$ ohne isolierte Stellen. Für jede Mengenpartition $\mathcal{E}$ der Rollen $R(D)$ gilt:

$$N = \Big\|_{R \in \mathcal{E}} N[R] \quad \textit{und} \quad D = \Big\|_{R \in \mathcal{E}} D[R]$$

Wir zeigen zunächst $N = \|_{R \in \mathcal{E}} N[R]$. Da $\mathcal{E}$ eine Partition ist, gilt $\bigcup_{R \in \mathcal{E}} R = R(D)$ und damit auch:

$$\bigcup_{R \in \mathcal{E}} T_{N[R]} = \bigcup_{R \in \mathcal{E}} r^{-1}(R) = T$$

Hier gilt sogar, dass alle $T_{N[R]}$ disjunkt sind. Dies ist notwendig, damit die Komposition der $N[R]$ überhaupt definiert ist.

Vollständige Rekonstruktion II

Da N ohne isolierte Stellen ist, gilt ${}^\bullet p \cup p^\bullet \neq \emptyset$, und damit folgt:

$$\bigcup_{R \in \mathcal{E}} P_{N[R]} = \bigcup_{R \in \mathcal{E}} ({}^\bullet T_{N[R]} \cup T_{N[R]}{}^\bullet) = P$$

Die $P_{N[R]}$ sind nicht nowendigerweise disjunkt.

Sei $(p,t) \in F \cap (P \times T)$, dann gibt es genau eine Rolle R, so dass $(p,t) \in F_{N[R]}$, da t in genau einem $N[R]$ vorkommt. Analog für $(t,p) \in F \cap (T \times P)$. Also gilt

$$\bigcup_{R \in \mathcal{E}} F_{N[R]} = F$$

Damit ist $N = \|_{R \in \mathcal{E}} N[R]$ klar, da die Knotenmenge vom N von denen der $N[R]$ überdeckt wird und alle $N[R]$ Teilnetze von N sind.

Vollständige Rekonstruktion III

Da sich $D[R]$ stets als Einschränkung von D ergibt, folgt $D = \|_{R \in \mathcal{E}} D[R]$ aus $N = \|_{R \in \mathcal{E}} N[R]$, denn alle Abbildung auf geteilten Elementen sind konsistent definiert. q.e.d.

Die Teilnetze ergeben somit eine Überdeckung, die aber im allgemeinen nicht disjunkt ist, da die Kommunikationsstellen geteilt werden.

R-Komponenten als Komponenten

Definition

Sei D ein Dienst und $N[R] = (P_R, T_R, F_R)$ seine R-Komponente.
Definiere $\Gamma_D[R] := (N[R], m_0^R, I^R, O^R)$ mit

$$
\begin{aligned}
I^R &= \{p \in P_{KK}(N) \mid p^\bullet \subseteq T_R\} \\
O^R &= \{p \in P_{KK}(N) \mid {}^\bullet p \subseteq T_R\} \\
P_i^R &= \{p \in P \mid R({}^\bullet p) = \emptyset \wedge p^\bullet \subseteq T_R\} \\
P_f^R &= \{p \in P \mid R(p^\bullet) = \emptyset \wedge {}^\bullet p \subseteq T_R\} \\
m_0^R(p) &= 1, \text{ falls } p \in P_i^R \text{ und 0 sonst}
\end{aligned}
$$

Theorem

Wenn D ein Dienst und $N[R]$ eine R-Komponente ist, dann ist $\Gamma_D[R]$ eine Komponente nach Definition 32.

Es ist $I^R \subseteq P_R$ zu zeigen: Es gilt $I^R \subseteq P_R$, denn $p^\bullet \subseteq T_R$ impliziert $p \in ({}^\bullet T_R \cup T_R{}^\bullet) = P_R$.

Analog folgt $O^R \subseteq P_R$, $P_i^R \subseteq P_R$ und $P_i^R \subseteq P_R$.

Wir zeigen $T_R{}^\bullet \cap I^R = \emptyset$: Sei $p \in I^R$, dann gilt nach Definition $R({}^\bullet p) \neq R(p^\bullet) \wedge p^\bullet \subseteq T_R$.

Da für Kommunikationsstellen stets $|R({}^\bullet p)| = |R(p^\bullet)| = 1$ gilt, ist $R({}^\bullet p) = \{r_1\}$ und $R(p^\bullet) = \{r_2\}$ für $r_1 \neq r_2$.

Da $p^\bullet \subseteq T_R$ gilt, muss $R(p^\bullet) \subseteq R$ sein, d.h. $r_2 \in R$ und $r_1 \notin R$.

Aus $r_1 \notin R$ folgt dann ${}^\bullet p \cap T_R = \emptyset$.

Dies ist gleichbedeutend mit $(\_F_R p) = \emptyset$.

Also gilt in $N[R]$ auch ${}^\bullet I^R = \emptyset$ und dies ist gleichbedeutend mit $T_R{}^\bullet \cap I^R = \emptyset$. Analog zeigt man $O^R \cap {}^\bullet T_R = \emptyset$.

Damit ist $\Gamma_D[R]$ eine Komponente. q.e.d.

Da für Komponenten eine Prozessdefinition existiert, können wir daher dies auch auf Rollenkomponenten übertragen:

$$Proc(N[R]) := Proc(\Gamma_D[R]) \tag{9}$$

Definition

Sei D ein Dienstnetz und $R \subseteq R(D)$ eine Rolle. Ein *Prozess von* $D[R]$ ist das Tupel (K, ϕ) mit $K = (B, E, \lessdot)$ und $\phi = (\phi^P, \phi^A)$, wobei gilt:

1. Der Prozess beschreibt den Anfangszustand M_0 von $D[R]$: $\phi^A(^\circ D[R])|_P = M_0$
2. $\phi^A : B \to A$ weist jeder Bedingung einen typgerechten Wert der Algebra zu: $\phi^A(p) \in [\![d(p)]\!]$ für alle $p \in \mathcal{P}$.
3. Für jedes Ereignis $e \in E$ gilt eine der beiden Bedingungen:
   1. Interne Aktion: Es existiert eine Transition $t \in T$ und eine Bindung α, so dass $\phi^P(^\bullet e) = {}^\bullet t$ und $\phi^P(e^\bullet) = t^\bullet$ und e beschreibt das Schalten von (t, α) mit $\alpha := \alpha$:

$$A, \alpha \models G(\phi^T(e)) \quad \wedge \quad \forall p \in {}^\bullet t : \phi^A(^\bullet e)(p) = \alpha(W(p,t)) \\ \wedge \quad \forall p \in t^\bullet : \phi^A(e^\bullet)(p) = \alpha(W(t,p))$$

   2. Externe Aktion: Für alle Stellen $x \in \mathcal{P}$ gilt $\phi(^\bullet e)(x) \leq 1$ und $\phi(e^\bullet)(x) \leq 1$. Aus $\phi(^\bullet e)(x) = 1$ folgt $x \in O \cup (\mathcal{P} \setminus P)$ und $\phi(e^\bullet)(x) = 1$ impliziert $x \in I \cup (\mathcal{P} \setminus P)$.

Theorem

Wenn $(K, (\phi^P, \phi^A))$ *ein Prozess von* $D[R]$ *ist, dann ist* (K, ϕ^P) *eine Komponentenprozess von* $\Gamma_D[R]$.

Beweis.

Direkt aus dem Vergleich von Definition 40 mit Definition 33. □

Verhaltensähnlichkeit I

Mit jeder Nachricht wird ein Ereignis assoziert:

Dazu verfeinern wir jeden Kanal p durch zwei Stellen p^{in} und p^{out} aufspalten, die durch die Transition t_p verbunden werden.

Verhaltensähnlichkeit II

Definition

Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz. Die *Kommunikationserweiterung von D* ist das Dienstnetz:

$$\hat{D} = (N', A, d', r', W', G', M_0')$$

$$\begin{aligned}
P' &= (P \setminus P_{KK}(N)) \cup \{p^{in}, p^{out} \mid p \in P_{KK}(N)\} \\
T' &= T \cup \{t_p \mid p \in P_{KK}(N)\} \\
F' &= F \cup \{(p^{in}, t_p), (t_p, p^{out}) \mid p \in P_{KK}(N)\} \\
d'(p^{in}) &= d'(p^{out}) = d(p) \\
r'(t_p) &= r_* \\
W'(p^{in}, t_p) &= x = W'(t_p, p^{out}) \qquad x \in X_{d(p)} \\
M_0'(p^{in}) &= M_0'(p^{out}) = \mathbf{0}
\end{aligned}$$

Bisimulation

$$\begin{array}{ccc} P & \sim & Q \\ a\downarrow & & \downarrow a \\ P' & \sim & Q' \end{array}$$

Kommunikationsäquivalenz: Bisimulation auf den Transitionen t_p.

Löschender Homomorphismus h:

$$h(t,\alpha) = \begin{cases} (t,\alpha), & \text{falls } t = t_p, p \in P_{KK}(D) \\ \lambda, & \text{sonst} \end{cases}$$

Die Übergangsrelation $\stackrel{\widehat{\cdot}}{\longrightarrow}$ erlaubt beliebig viele interne Schritte vor und nach einer Kommunikationstransition:

$$M_1 \xrightarrow[\widehat{D}]{\widehat{t,\alpha}} M_2 : \Longleftrightarrow \exists w : M_1 \xrightarrow[\widehat{D}]{w} M_2 \wedge h(w) = (t,\alpha)$$

Definition

Seien D_1 und D_2 zwei Dienstnetze mit gleichen Kommunikationskanälen, d.h. $P_{KK}(D_1) = P_{KK}(D_2)$.
Eine binäre Relation $\mathcal{B}$ auf heißt *Kommunikationsbisimulation*, falls für alle $(M_1, M_2) \in \mathcal{B}$ gilt:

1. Wenn $M_1 \xrightarrow[\hat{D}_1]{\widehat{t,\alpha}} M_1'$ gilt, dann existiert ein M_2', so dass $M_2 \xrightarrow[\hat{D}_2]{\widehat{t,\alpha}} M_2'$ und $(M_1', M_2') \in \mathcal{B}$ gilt.
2. Wenn $M_2 \xrightarrow[\hat{D}_2]{\widehat{t,\alpha}} M_2'$ gilt, dann existiert ein M_1', so dass $M_1 \xrightarrow[\hat{D}_1]{\widehat{t,\alpha}} M_1'$ und $(M_1', M_2') \in \mathcal{B}$ gilt.

Notation: $M_1 \approx M_2$, falls $(M_1, M_2) \in \mathcal{B}$ für ein $\mathcal{B}$ gilt.

Man erkennt leicht, dass $\approx$ eine Äquivalenzrelation ist. Man beachte, dass für die Definition der Simulation die erweiterten Netze $\hat{D}_i, i = 1, 2$ und nicht die Dienstnetze D_i selbst verwendet werden.

Rollensubstitution I

Rollensubstitution:

$$\langle D; \{(R_1, R_1'), \ldots, (R_n, R_n')\}; D' \rangle$$

Das Verhalten ändert sich nicht, wenn wir in D eine R_i-Komponente $D[R_i]$ gegen die korrespondierende R_i'-Komponente $D'[R_i']$ von D' substituieren.

Rollensubstitution II

Definition

Die *Verhaltensäquivalenz* $\langle D; \rho; D' \rangle$ gilt genau dann, wenn folgendes erfüllt ist:

1. $\mathcal{E} = \{R_1, \ldots, R_n\}$ ist eine Mengenpartition auf $R(D)$.
2. $\mathcal{E}' = \{R'_1, \ldots, R'_n\}$ ist eine Mengenpartition auf $R(D')$.
3. $\rho : \mathcal{E} \to \mathcal{E}'$ ist eine Bijektion.
4. Für jede Auswahl $A_i, B_i \in \{D[R_i], D'[\rho(R_i)]\}$ gilt $(A_1 \| \cdots \| A_n) \approx (B_1 \| \cdots \| B_n)$.

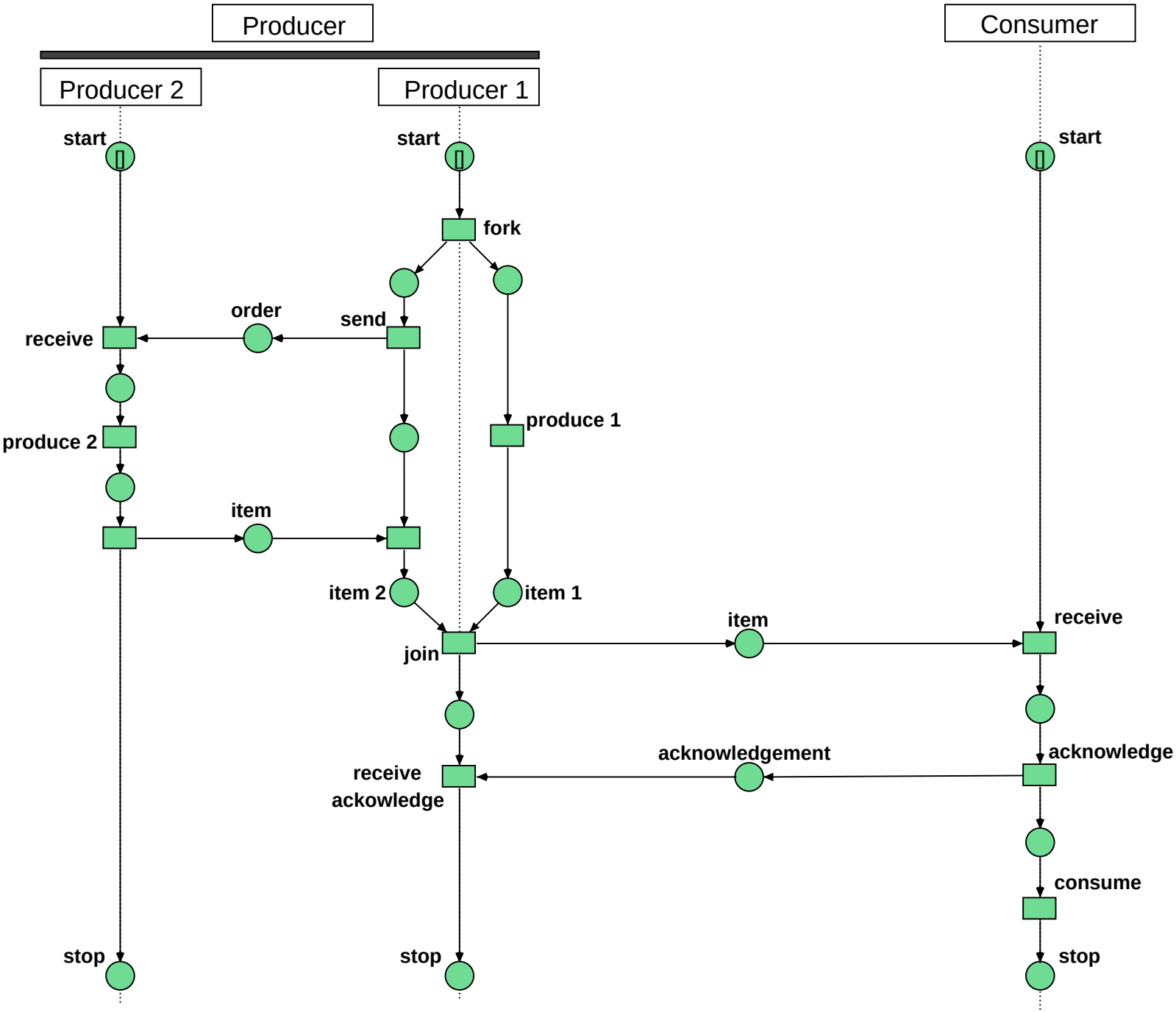
Statt $\langle D; \rho; D' \rangle$ notieren wir auch explizit: $\langle D; \{(R, \rho(R)) \mid R \in \mathcal{E}\}; D' \rangle$.

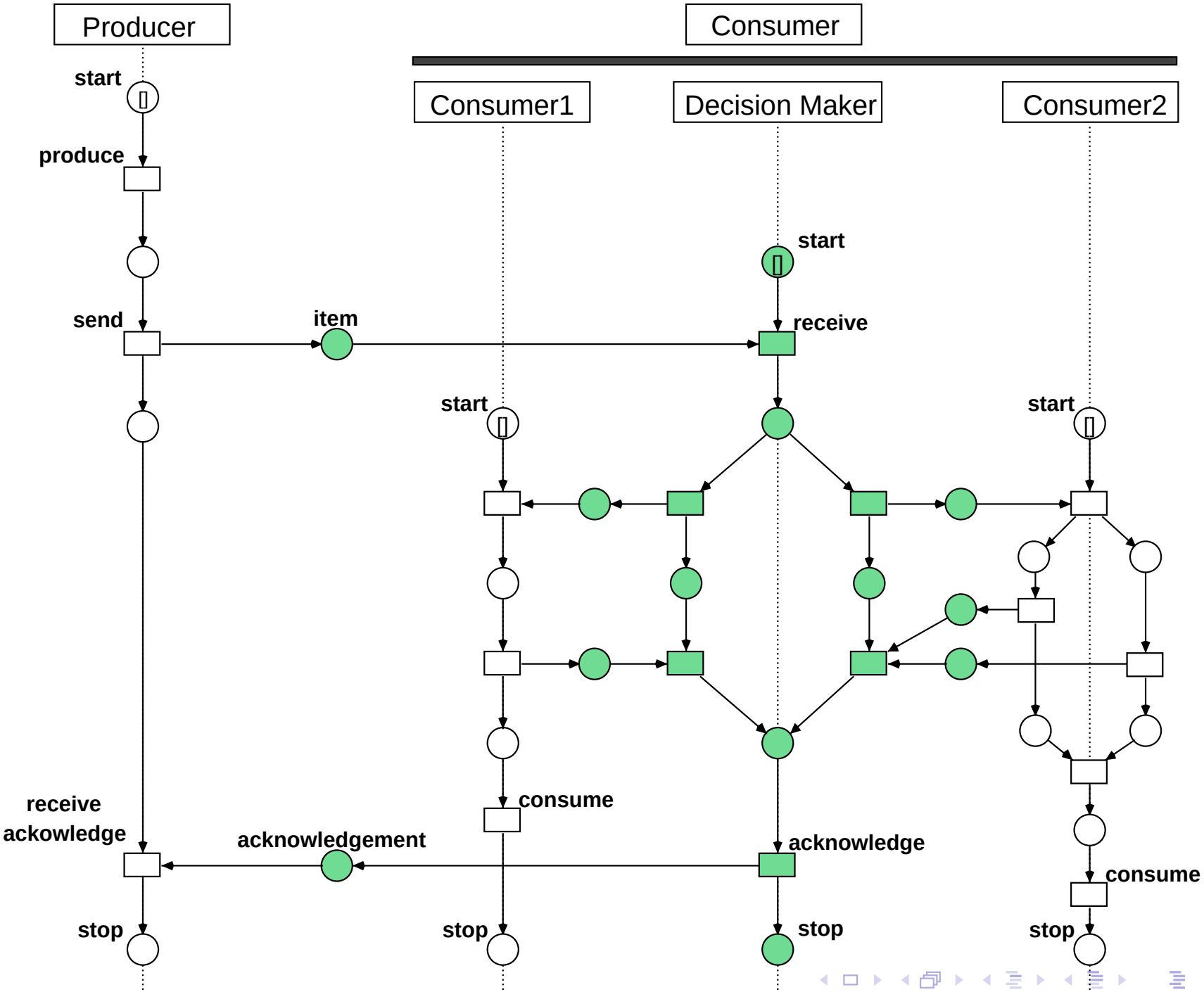
Im Falle $|\mathcal{E}| = 2$ wird $\langle D; \{(R, R'), (R(D) \setminus R, R(D') \setminus R)\}; D' \rangle$ kürzer notiert:

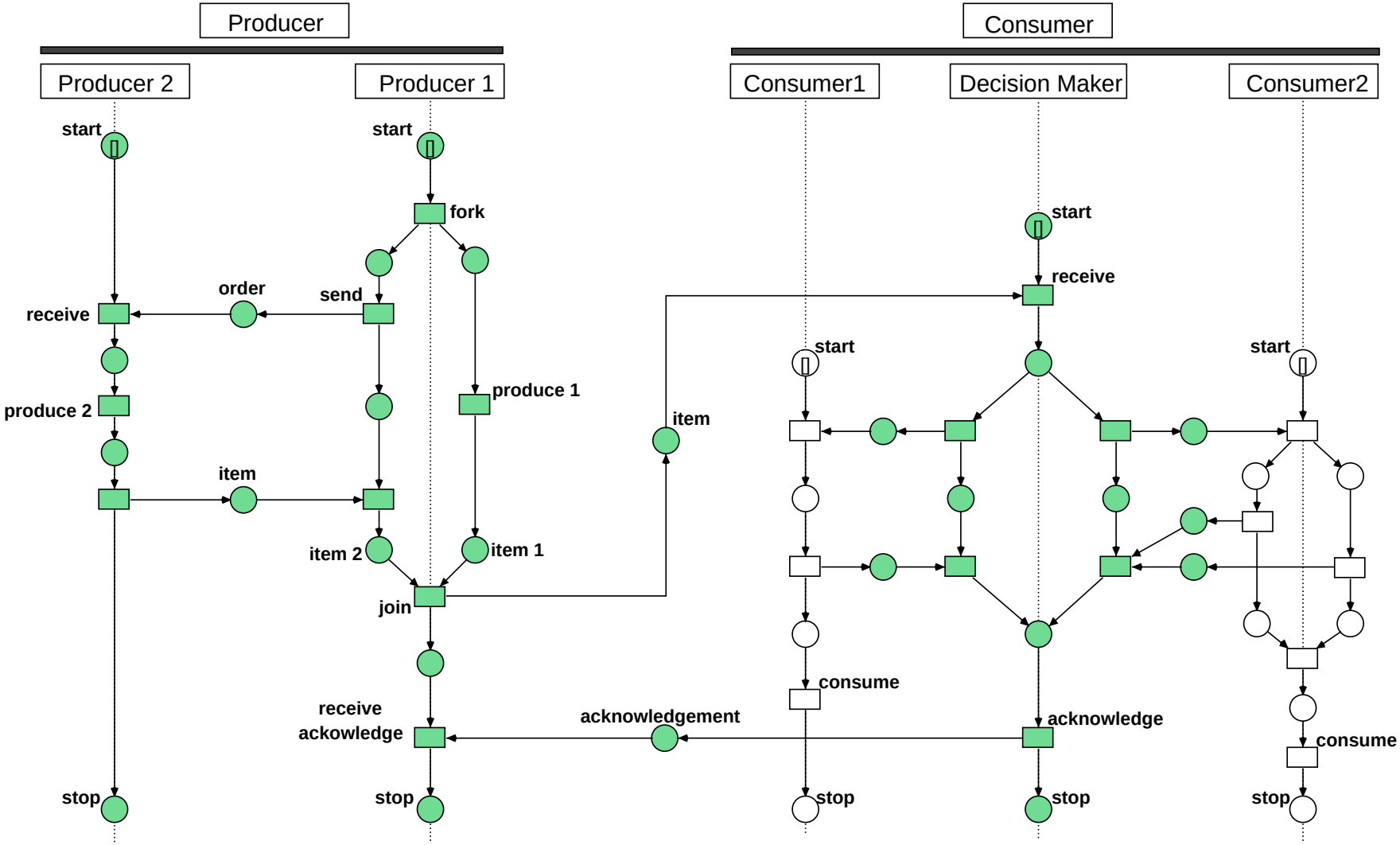
$$\langle\langle D; R, R'; D' \rangle\rangle$$

Rollensubstitution III

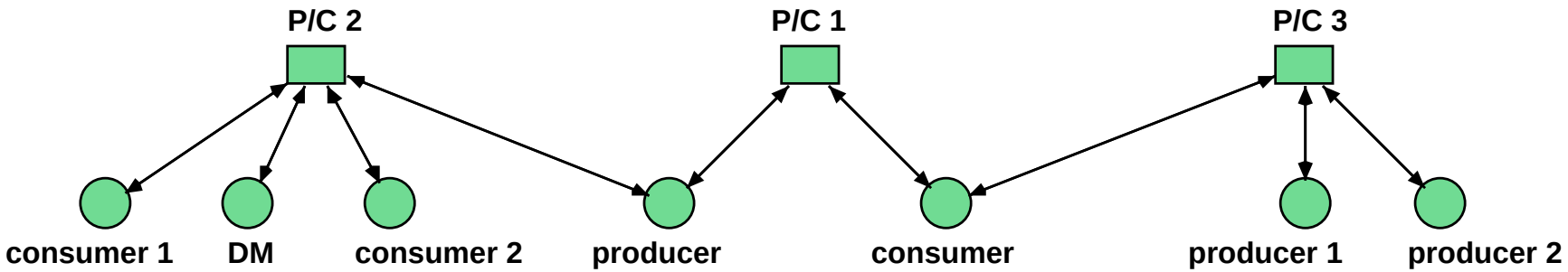
Die Definition von $\langle D; \rho; D' \rangle$ fordert, dass die Komponente $D[R]$ ohne Unterschied gegen $D'[\rho(R)]$ austauschbar ist. Mit $\langle\langle D; R, R'; D' \rangle\rangle$ drücken wir also speziell aus, dass die Interaktion der Rolle R mit seiner Umwelt nicht von R' unterschieden werden kann.


Producer
Consumer
Producer 2
Producer 1
start
start
start
fork
order
send
receive
produce 2
produce 1
item
item 2
item 1
item
receive
join
acknowledgement
acknowledge
receive
ackowledge
consume
stop
stop
stop


Producer
Consumer
Consumer1
Decision Maker
Consumer2
start
produce
send
item
start
receive
start
start
receive
ackowledge
acknowledgement
consume
acknowledge
consume
stop
stop
stop
stop


Producer
Consumer
Producer 2
Producer 1
Consumer1
Decision Maker
Consumer2
start
start
fork
start
receive
order
send
receive
start
start
produce 2
produce 1
item
item
item 2
item 1
join
consume
receive
ackowledge
acknowledgement
acknowledge
consume
stop
stop
stop
stop
stop

Dienst/Rollen-Beziehungen



Die Rollenäquivalenz ist eine parametrisierte Äquivalenzrelation.

Theorem

Seien D_1, D_2 und D_3 Dienstnetze. Sei $\mathcal{E}_1 = \{A_1, \ldots, A_n\}$ eine Mengenpartition auf $R(D_1)$, $\mathcal{E}_2 = \{B_1, \ldots, B_n\}$ eine auf $R(D_2)$ und $\mathcal{E}_3 = \{C_1, \ldots, C_n\}$ eine auf $R(D_3)$.

- *Die Relation ist reflexiv:*

$$\langle D_1; id_{\mathcal{E}_1}; D_1 \rangle$$

- *Die Relation ist symmetrisch:*

$$\langle D_1; \rho; D_2 \rangle \iff \langle D_2; \rho^{-1}; D_1 \rangle$$

- *Die Relation ist transitiv:*

$$\big(\langle D_1; \rho; D_2 \rangle \wedge \langle D_2; \tau; D_3 \rangle\big) \Longrightarrow \langle D_1; (\tau \circ \rho); D_3 \rangle$$

Direkt aus Definition 44.

Verhaltensäquivalente Rollenkomponenten können gegeneinander ausgetauscht werden.

Theorem

Die Relation $\langle\!\langle \cdot; \cdot; \cdot \rangle\!\rangle$ ist substitutiv:

$$\langle\!\langle D_1; A, (B \uplus B'); (D_2[B] \| D_2[\overline{B}]) \rangle\!\rangle \wedge \langle\!\langle D_2; B', C; D_3 \rangle\!\rangle \\ \Longrightarrow \langle\!\langle D_1; A, (B \cup C); (D_2[\overline{B}] \| D_3[C]) \rangle\!\rangle$$

Beweis.

Für die Substitution ist $N[R_1 \cup R_2] = N[R_1] \| N[R_2]$ auszunutzen:
$D_1[A] \approx D_2[B \cup B'] = D_2[B] \| D_2[B']$ und $D_2[B'] \approx D_3[C]$. □

Wir können die Rollenaufteilung auch vergröbern, indem wir Rollen zusammenfassen.

Theorem

Wenn $\langle D_1; \{(A_1, B_1), (A_2, B_2), \ldots, (A_n, B_n)\}; D_2\rangle$ *für* $n \geq 2$ *gilt, dann gilt auch*

$$\langle D_1; \{(A_1 \cup A_2, B_1 \cup B_2), (A_3, B_3), \ldots, (A_n, B_n)\}; D_2\rangle.$$

Sei die Rollenpartition $\{A_1, \ldots, A_n\}$ und $\{B_1, \ldots, B_n\}$ gegeben sowie die Vergröberungen $\{(A_1 \cup A_2), \ldots, A_n\}$ und $\{(B_1 \cup B_2), \ldots, B_n\}$ gegeben.

Wir betrachten nun eine Kommunikationsstelle p in D_1. Die Stelle p verbindet unterschiedliche Rollen: $R({}^\bullet p) = A_i \neq R(p^\bullet) = A_j$.

Diese Stelle verbindet auch in D_2 verschiedene Rollen: B_i und B_j, denn nach Definition 43 besitzen beide die gleichen Mengen an Kommunikationsstellen.

Die Stelle p wird in beiden Netzen durch die Kommunikationshandlung t_p, die die Marke von p^{in} nach p^{out} transportiert, verfeinert.

Das Schalten von t_p in D_1 kann von D_2 simuliert werden – und umgekehrt.

Gilt $R({}^\bullet p) = A_i$ und $R(p^\bullet) = A_j$ in D_1, dann gilt aufgrund der Substitutionseigenschaft in Definition 44 auch $R({}^\bullet p) = B_i$ und $R(p^\bullet) = B_j$ in D_2.

Wir unterscheiden folgende Fälle:

1. Gilt $R({}^\bullet p) = A_1$ und $R(p^\bullet) = A_2$ (oder umgekehrt) in D_1, dann ist $R({}^\bullet p) = B_1$ und $R(p^\bullet) = B_2$ in D_2 und in den beiden Rollenkomponenten $D_1[A_1 \cup A_2]$ und $D_2[B_1 \cup B_2]$ wird p zu einer internen Stelle, so dass es für die Rollenpaarung $\{(A_1 \cup A_2, B_1 \cup B_2), (A_3, B_3), \ldots, (A_n, B_n)\}$ keine zu simulierende Transition t_p gibt.
   Es ist also nicht zu zeigen.

2. Ist weder $R({}^\bullet p)$ noch $R(p^\bullet)$ gleich A_1 oder A_2 in D_1, dann sind auch B_1 und B_2 in D_2 nicht involviert.
   Die Simulation in der Vergröberung ergibt sich dann aus der ursprünglichen Simulationseigenschaft.
3. Es gilt $R({}^\bullet p) = A_1$ und $R(p^\bullet) \neq A_2$ (analog für A_2) in D_1.
   Dann kommuniziert die Rollenkomponente $D_1[A_1 \cup A_2]$ durch t_p mit einer der Rollenkomponente $D_1[A_j]$ mit $j \geq 3$.
   Dann kommuniziert aber auch $D_2[B_1 \cup B_2]$ durch t_p mit $D_2[B_j]$ und die Aktionen simulieren einander, da sie es auch in der ursprünglichen Form taten.

Die Betrachtung für aus der Perspektive von D_2 verläuft symmetrisch.

Definition

Die Menge der CTL^* -Formeln $CTL^*(X)$ über einer Menge atomarer Aussagen X ist die Menge aller Pfadformeln:

1. Jede atomare Aussage $x \in X$ ist eine Zustandsformel.
2. Sind p und q Zustandsformeln, so auch $\neg p$, $(p \vee q)$ und $\mathbf{E}p$.
3. Jede Zustandsformel ist eine Pfadformel
4. Sind P, Q Pfadformeln, dann auch $\neg P$, $(P \vee Q)$, $\bigcirc P$ und $(P\ \mathbf{U}\ Q)$.

Wir definieren $\textsc{True} := (x \vee \neg x)$ für ein beliebiges Atom und $\textsc{False} := \neg\textsc{True}$.

eventually-Operator: $\Diamond\psi := (\textsc{True}\ \mathbf{U}\ \psi)$

leads-to-Operator: $\psi \rightsquigarrow \phi := \Box(\psi \Longrightarrow \Diamond\phi)$.

Temporallogik II

Die bestehenden Operatoren besitzen duale Operatoren:

$$
\begin{aligned}
\neg(\phi \wedge \psi) &:= (\neg\phi \vee \neg\psi) \\
\neg\Box\phi &:= \Diamond\neg\phi \\
\neg(\phi \,\mathbf{U}\, \psi) &:= (\neg\psi \,\mathbf{R}\, \neg\phi) \\
\neg\mathbf{A}\psi &:= \mathbf{E}\neg\psi \\
\neg\bigcirc\psi &= \bigcirc\neg\psi
\end{aligned}
$$

Kripkemodell

Das *Kripkemodell* $M = (K, \alpha, z_s)$ dieser Logik besteht aus

- einer Kripke-Struktur $K = (Z, \rightarrow)$,
- Bewertungsfunktion $\alpha : Z \rightarrow 2^X$
- und dem Initialzustand $z_s \in Z$.

Gültigkeit I

Die Gültigkeit von Aussagen wird auf Aktionssequenzen von K definiert:

$$\sigma = z_0 z_1 z_2 \cdots \quad \text{mit} \quad \forall i \in \mathbb{N} : z_i \rightarrow z_{i+1}$$

Der Pfad σ_k ist definiert als der Pfad, der im k-ten Zustand von σ beginnt:

$$\sigma_k = z_k z_{k+1} \cdots$$

Sei $\Sigma(z)$ die Menge aller unendlichen Zustandssequenzen, die mit z beginnen.

Gültigkeit II

Definition

Sei $M = (K, \alpha, z_s)$ ein Modell. Die Gültigkeit einer CTL^*-Formel P in einem Pfad $\sigma = z_0 z_1 \cdots$ wird mit $M, \sigma \models P$. Sie ist folgendermaßen definiert:

$$
\begin{array}{lcl}
M, z_0 \models x & \iff & x \in \alpha(z_0) \\
M, z_0 \models \neg p & \iff & M, z_0 \not\models p \\
M, z_0 \models (p \vee q) & \iff & M, z_0 \models p \text{ oder } M, z_0 \models q \\
M, z_0 \models \mathbf{E}p & \iff & \exists \sigma \in \Sigma(z_0) : M, \sigma \models p \\
M, \sigma_k \models p & \iff & z_k \models p \\
M, \sigma_k \models \neg P & \iff & M, \sigma_k \not\models P \\
M, \sigma_k \models (P \vee Q) & \iff & M, \sigma_k \models P \text{ oder } \sigma_k \models Q \\
M, \sigma_k \models \bigcirc P & \iff & M, \sigma_{k+1} \models P \\
M, \sigma_k \models (P \ \mathbf{U} \ Q) & \iff & \exists n \geq k : M, \sigma_n \models Q \wedge \forall k \leq i < n : M, \sigma_i \models P
\end{array}
$$

Eine Aussage P ist gültig, notiert $M \models P$, wenn sie in z_0 gültig ist.

Bsp.: Korrektheit für Workflownetze

Schaltfolgen, die ψ erfüllen und gleichzeitig stets terminieren könne:

$$\forall M \in RS(D, M_0) : \psi \wedge \exists M' \in RS(D, M_0) : \pi(M') = m_f \tag{10}$$

Für $\psi = \pi(M') \geq m_f \Longrightarrow \pi(M') = m_f$ erhält man schwache Korrektheit.

in der Temporallogik $\mathrm{CTL}^*(X)$

$$\mathbf{A}\Box(\psi \wedge \mathbf{E}\Diamond(\pi(M) = m_f)) \tag{11}$$

Temporallogik für partielle Ordnungen I

Definition

Die Menge $LTL_{po}(X)$ der LTL-Formeln für partielle Ordnungen über einer Menge atomarer Aussagen X ergibt sich induktiv.

1. Jede atomare Aussage $x \in X$ ist eine $LTL_{po}(X)$-Formel.
2. Sind P, Q $LTL_{po}(X)$-Formeln, dann auch $\neg P$, $(P \vee Q)$, $\textcircled{\vee} P$ und $(P \mathbf{U} Q)$.

Semantik I

Diese Logik wird durch die durch Prozesse eines Petrinetzes interpretiert.

Das *Prozessmodell* $M = (Proc(N), \alpha)$ für $LTL_{po}(X)$ besteht aus

- der Menge aller Prozesse π eines Netzes N und
- einer Bewertungsfunktion $\alpha = (\alpha_\pi : \mathcal{C}_\pi \to 2^X \mid \pi \in Proc(N))$, die jedem Stellenschnitt C des Prozesses π die Menge der gültigen atomaren Aussagen zuweist.

Semantik II

Definition

Sei $M = (Proc(N), \alpha)$ ein Prozessmodell für $LTL_{po}(X)$ und sei $\pi = (K, \phi) \in Proc(N)$ ein Prozess. Die Gültigkeit einer temporalen Aussage P in einem Stellenschnitt $C \in \mathcal{C}_\pi$ ist induktiv definiert:

$$
\begin{array}{lcl}
\pi, C \models x & \iff & x \in \alpha_\pi(C) \\
\pi, C \models \neg P & \iff & \pi, C \not\models P \\
\pi, C \models (P \vee Q) & \iff & \pi, C \models P \text{ oder } \pi, C \models Q \\
\pi, C \models \bigcirc\!\!\!\!\vee P & \iff & \exists C' \in \mathcal{C}_K : C \to C' \wedge \pi, C' \models P \\
\pi, C \models (P \mathbf{U} Q) & \iff & \exists C' \in \mathcal{C}_K : C \xrightarrow{*} C' \wedge \pi, C' \models Q \wedge \\
 & & \forall C'' \in \mathcal{C}_K : C \xrightarrow{*} C'' \xrightarrow{+} C' \Longrightarrow \pi, C'' \models P
\end{array}
$$

Der Prozess $\pi = (K, \phi)$ erfüllt P, wenn P im initialen Schnitt $^\circ K$ gilt:

$$\pi \models P \iff \pi, {}^\circ K \models P$$

Ein Netz N erfüllt eine Aussage, wenn diese in all seinen Prozessen gilt.

Kompositionstheorem der Rollenkomponenten

Theorem

Sei $R_1, R_2 \in \mathcal{R}$ *mit* $R_1 \cap R_2 = \emptyset$.

$$Proc(D[R_1], \psi_1) \cap Proc(D[R_2], \psi_2) = Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$$

Sei $K \in Proc(D[R_1], \psi_1) \cap Proc(D[R_2], \psi_2)$, dann gilt $K \models \psi_1$ und $K \models \psi_2$ und damit auch $K \models (\psi_1 \wedge \psi_2)$.

Da $Proc(\Gamma, \psi) \subseteq Proc(\Gamma)$ gilt, folgt $K \in Proc(D[R_i])$ für $i = 1, 2$ und mit Theorem 34 auch $K \in Proc(D[R_1] \| R[R_2])$.

Insgesamt also $K \in Proc(D[R_1] \| R[R_2], \psi_1 \wedge \psi_2)$.

Sei $K \in Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$, dann gilt $K \models (\psi_1 \wedge \psi_2)$ und damit $K \models \psi_1$ und $K \models \psi_2$.

Wenn $K \in Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$, dann auch $K \in Proc(D[R_1] \| D[R_2]$ und auch $K \in Proc(D[R_1]) \cap Proc(R[R_2])$.

Insgesamt also $K \in Proc(D[R_i], \psi_i)$ für $i = 1, 2$.

Gesteuerte Dienste I

Wir bezeichnen mit $(D \times \psi)$ den durch ψ gesteuerten Dienst.

Gesteuerte Erreichbarkeitsmenge
$RS(D \times \psi, M_0) := \bigcup_{i \in \mathbb{N}} RS_i(D \times \psi, M_0)$ mit:

$$
\begin{aligned}
RS_{k+1}(D \times \psi, M_0) &:= \{M' \mid \exists M \in RS_k(D \times \psi, M_0) : M \xrightarrow{t,\alpha} M' \wedge M' \models \psi\} \\
RS_0(D \times \psi, M_0) &:= \begin{cases} \{M_0\}, & \text{falls } M_0 \models \psi \\ \emptyset, & \text{sonst} \end{cases}
\end{aligned}
$$

Die Menge $RS(D \times \psi, M_0)$ ist also genau dann leer, wenn M_0 die Eigenschaft ψ nicht besitzt.

Wir schränken den Erreichbarkeitsgraph $RG(D, M_0)$ eines Dienstnetz D auf solche Zustände M ein, die ψ erfüllen, indem wir die Knoten von $RG(D, M_0)$ auf $RS(D \times \psi, M_0)$ einschränken.

Korrektheit gesteuerter Dienste

Analog zu Diensten, nur $RS(D \times \psi, M)$ anstelle $RS(D, M)$.

Definition

Wenn D ein Dienst ist und $\psi \in \mathrm{CTL}^*(X)$ eine Formel, so bezeichnen wir mit $(D \times \psi)$ den durch ψ gesteuerten Dienst. Ein gesteuertes Dienstnetz $(D \times \psi)$ heißt *korrekt*, wenn gilt:

$$\begin{aligned} &\forall M \in RS(D \times \psi, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f \\ &\wedge \forall M \in RS(D \times \psi, M_0) : \exists M' \in RS(D \times \psi, M) : \pi(M') = m_f \\ &\wedge \forall t \in T : \exists M_0 : f(M_0) = m_0 \wedge \exists M \in RS(D \times \psi, M_0) : M \xrightarrow{t} \end{aligned}$$

Ein gesteuertes Dienstnetz $(D \times \psi)$ heißt *schwach korrekt*, wenn nur die ersten beiden Bedingungen gelten
Ein gesteuerte Dienstnetz $D \times \psi$ heißt *strukturell (schwach) korrekt*, wenn es für alle Anfangsmarkierungen M mit $\pi(M) = m_0$ (schwach) korrekt ist.

Theorem

Sei D ein Dienst, der in M_0 terminieren kann, d.h.
$\exists M \in RS(D, M_0) : \pi(M) = m_f$, dann existiert eine Steuerung ψ, so dass $D \times \psi$ schwach korrekt ist.

Beweis.

D kann terminieren. Also existiert eine Schaltfolge $M_0 \xrightarrow{t_1,\alpha_1} M_1 \xrightarrow{t_2,\alpha_2} \cdots M_n = M$ mit $\pi(M) = m_f$. Wählen wir speziell

$$\psi = \bigwedge_{k=0}^{n} \bigcirc^k (M = M_k) = \Big((M = M_0) \wedge \bigcirc (M = M_1) \wedge \bigcirc\bigcirc (M = M_2) \wedge \cdots \Big),$$

so erlaubt die Steuerung nur diese Schaltfolge, und der Dienst $D \times \psi$ ist korrekt. □

Dienstklassen I

- Dienstklassen sind Mengen von Dienstnetzen, die konsistent bezüglich der verwendeten Rollen sind.
- Die Rollen der Dienstnetze einer Dienstklasse sind nicht notwendigerweise disjunkt. Es ist möglich, dass die gleiche Rolle r in zwei Dienstnetzen D_1 und D_2 vorkommt.
- In diesem Fall fordern wir aber, dass die Rolle in beiden Netzen identische Prozesse beschreiben:

$$\langle\langle D_1; R, R; D_2 \rangle\rangle$$

- Dadurch wird ausgedrückt, dass Interaktion und Rollen zwei untrennbare Konzepte sind.

Dienstklassen II

Zu einer Menge an Diensten $\mathcal{D}$ bezeichnet $\mathcal{D}(R)$ die Menge aller Dienste, die die Rolle $R \in \mathcal{R}$ enthalten:

$$\mathcal{D}(R) = \{D \in \mathcal{D} \mid R(D) \supseteq R\} \tag{13}$$

Wir definieren eine Auswahlfunktion $D^{\mathcal{D}}$, die zu jeder Rolle R einen Repräsentanten aus der Menge $\mathcal{D}(R)$ auswählt. Diesen Repräsentanten $D^{\mathcal{D}}(R)$ bezeichnen wir als den *Referenzdienstnetz* von R. Für den Referenzdienst von R gilt offensichtlich:

$$R \subseteq R(D^{\mathcal{D}}(R))$$

Dienstklassen III

Definition

Sei $\mathcal{R}$ eine Rollenstruktur. Eine *Dienstklasse* $(\mathcal{D}, D^{\mathcal{D}})$ besteht aus einer Menge an Dienstnetzen $\mathcal{D}$ mit

$$\forall R \in \mathcal{R} : \forall D_1, D_2 \in \mathcal{D}(R) : \langle\langle D_1; R, R; D_2 \rangle\rangle$$

und einer Abbildung $D^{\mathcal{D}}(R) : \mathcal{R} \to \mathcal{D}$ mit

$$\forall R \in \mathcal{R} : D^{\mathcal{D}}(R) \in \mathcal{D}(R)$$

Für jede Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ generiert jede Rolle R die Klasse $\mathcal{D}(R)$, deren Elemente $D \in \mathcal{D}(R)$ sich nicht in ihrem Verhaltem bezüglich der Rolle R unterscheiden lassen.

Dienstklassen IV

Für eine Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ definieren wir die Menge aller Dienstkomponentenprozesse:

$$Proc(\mathcal{D}, \mathcal{R}) := \bigcup_{D \in \mathcal{D}} \bigcup_{R \in \mathcal{R}} Proc(D[R]) \tag{14}$$

Atome

Wir definieren im folgenden die Menge der Formeln für eine Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$.

Die Atome sind von der Form

1. $A_{k,q}$, wobei $k \in \mathbb{Q}^{|P_{KK}(D)|}$ und $q \in \mathbb{Q}$ ist, oder
2. $A_{p,c,n}$, wobei $p \in P_{KK}(D)$, $c \in [\![d(p)]\!]$ und $n \in \mathbb{N}$ ist.

Die Menge aller dieser Atome von D wird mit $\mathcal{A}_D$ bezeichnet.

Die Menge aller Atome ist $\mathcal{A}_{\mathcal{D}} = \bigcup_{D \in \mathcal{D}} \mathcal{A}_D$.

Bewertung

Zu einer Markierung M ergibt sich die Bewertungsfunktion α als die Familie der α_D:

$$\begin{aligned}\alpha_D(M) \quad = \quad & \{A_{k,c} \mid \textstyle\sum_{p \in P_{KK}(D)} (k(p) \cdot \pi(M)(p)) \leq c\} \cup \\ & \{A_{p,c,n} \mid M(p)(c) \leq n\}\end{aligned}$$

Zu einem Prozess $\pi \in \mathit{Proc}(D)$ erweitert sich dies zu einer Bewertung auf den Schnitten:

$$\alpha_{D,\pi}(C) = \alpha_D(\phi(C))$$

Damit ist dann $\mathit{PM}(\mathcal{D}) = (\bigcup_{D \in \mathcal{D}} \mathit{Proc}(D), \alpha)$ das von D erzeugte Standardprozessmodell der Logik $\mathrm{LTL}_{po}(\mathcal{A}_{\mathcal{D}})$.

Prozessformeln

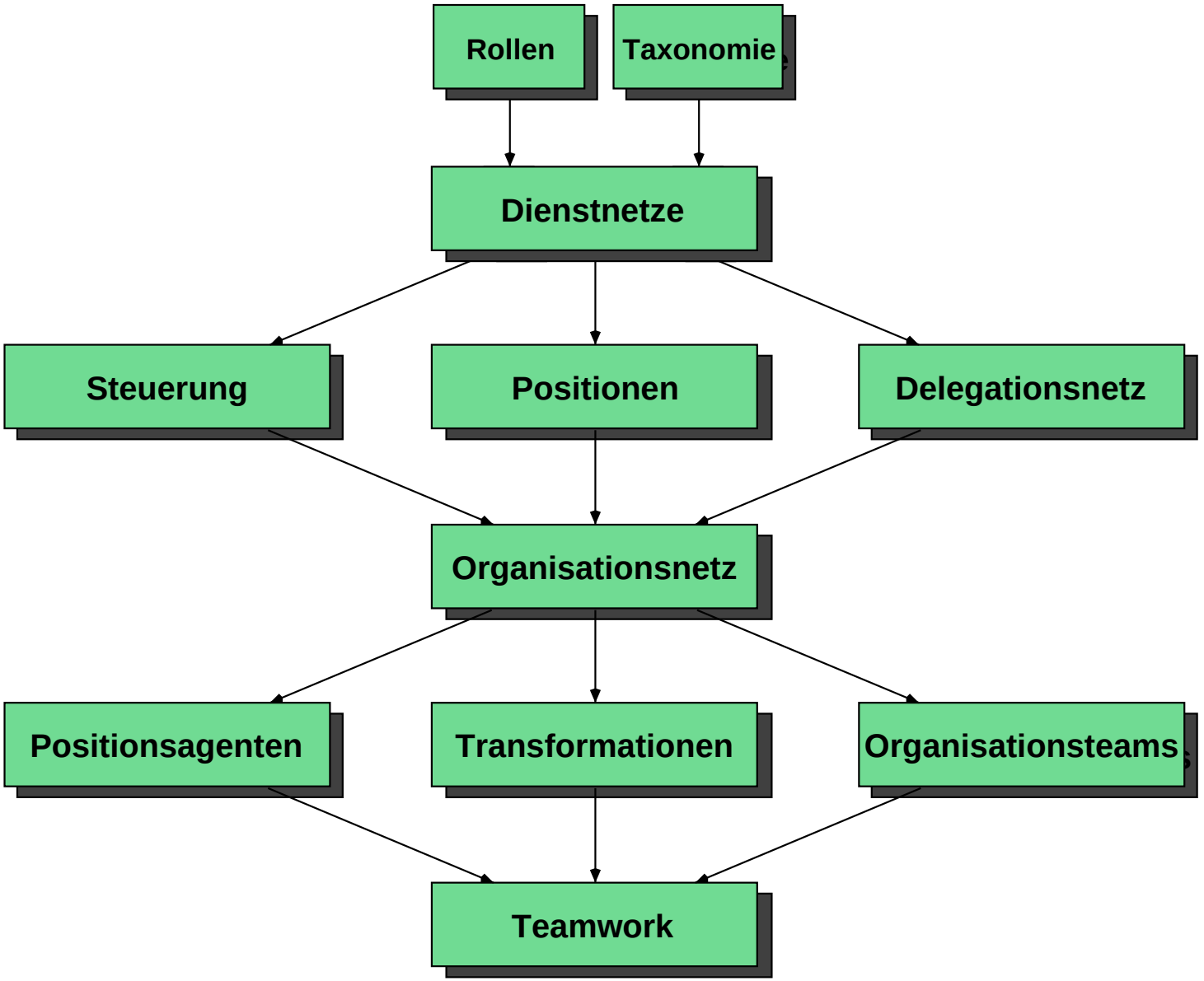
Um sowohl die Steuerung mittels des Erreichbarkeitgraphen als auch die Erfüllbarkeit im Prozessmodell betrachten zu können, schränken wir die folgenden Betrachtungen auf Formeln ein, die sowohl in $\mathrm{LTL}(X)$ als auch $\mathrm{LTL}_{po}(X)$ sind.

Definition

Sei $(\mathcal{D}, D^{\mathcal{D}})$ eine Dienstklasse. Die Menge der *Prozessformeln* ist definiert als die Menge:

$$PF_{\mathcal{D}} := \mathrm{LTL}(\mathcal{A}_{\mathcal{D}}) \cap \mathrm{LTL}_{po}(\mathcal{A}_{\mathcal{D}}) \tag{15}$$

Für die Steuerung konvertieren wir jede Prozessformel ψ implizit als eine CTL^*-Formel, indem wir den Pfadquantor **A** hinzufügen, d.h. wir betrachten $\mathbf{A}\psi$ anstelle von ψ.


Rollen
Taxonomie
Dienstnetze
Steuerung
Positionen
Delegationsnetz
Organisationsnetz
Positionsagenten
Transformationen
Organisationsteams
Teamwork

Literatur I

[Baader u. a. 2003] BAADER, F. (Hrsg.) ; CALVANESE, D. (Hrsg.) ; MCGUINNESS, D. (Hrsg.) ; NARDI, D. (Hrsg.) ; PATEL-SCHNEIDER, P.F. (Hrsg.): *Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003

[Kindler 1997] KINDLER, Ekkart: A Compositional Partial Order Semantics for Petri Net Components. In: AZEME, Pierre (Hrsg.) ; BALBO, Gianfranco (Hrsg.): *Application and Theory of Petri Nets* Bd. 1248, Springer-Verlag, 1997

[Schmidt-Schauß und Smolka 1991] SCHMIDT-SCHAUSS, M. ; SMOLKA, G.: Attributive concept descriptions with complements. In: *Articial Intelligence* 48 (1991), Nr. 1, S. 1–26