OE-Vorlesung 2017
Einführung in Petrinetze

Dr. Daniel Moldt
Moldt@Informatik.Uni-Hamburg.de
Folien: Dr. Frank Heitmann

Fachbereich Informatik
MIN-Fakultät
Universität Hamburg

Informatik-OE 2017

Warum Modellierung?

Frage

Warum modellieren wir überhaupt?

- Gedanken machen, Dinge konkretisieren
- Kommunikation
- Schon am Modell Eigenschaften überprüfen

Was ist ein Modell?

Stachowiak (1973): Allgemeine Modelltheorie

Ein Modell hat drei Hauptmerkmale:

- Abbildungsmerkmal $\longrightarrow$ Analogie
- Verkürzungsmerkmal $\longrightarrow$ Abstraktion
- Pragmatisches Merkmal $\longrightarrow$ Nützlichkeit

Und warum Petrinetze?

Das Mittel der Wahl

Petrinetze sind eine *Modellierungstechnik*.

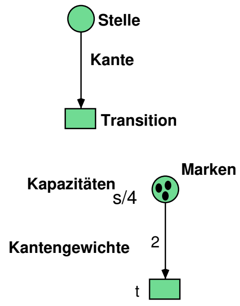
- Mit Petrinetzen ist *Nebenläufigkeit* modellierbar
- Petrinetze sind graphisch
  - $\Rightarrow$ Gut zu verstehen
- Petrinetze haben einen formalen Unterbau
  - $\Rightarrow$ Exaktheit
  - $\Rightarrow$ Eigenschaften formal (und automatisch) überprüfbar

Anmerkung

Im letzten Punkt steckt, warum eine (mathematische) *Theorie* sinnvoll ist/sein kann. Dinge sind *exakt* und *eindeutig* ausdrückbar. Dinge können exakt *nachgewiesen* oder *widerlegt* werden.

Petrinetze - Grundlagen

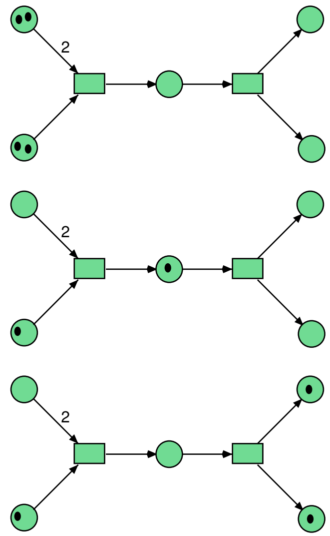
- **Stellen** (Plätze) als passive Komponenten
- **Transitionen** als aktive Komponenten
- gerichtete **Kanten** zwischen Stellen und Transitionen (und andersherum)



- Kante von Stelle s zu Transition t: s ist *Eingangs*stelle von t.
- Kante von Transition t zu Stelle s: s ist *Ausgangs*stelle von t.

Petrinetze: Dynamik

Marken auf den Stellen ermöglichen den Transitionen zu **schalten**:



Petrinetze: Dynamik

Definition (Aktiviertheit)

Eine Transition ist genau dann **aktiviert**, wenn

1. alle Eingangsstellen ausreichend Marken beinhalten
2. und die Kapazität jeder Ausgangsstellen ausreicht, um entsprechend viele zusätzliche Marken aufzunehmen.

Definition (Schalten)

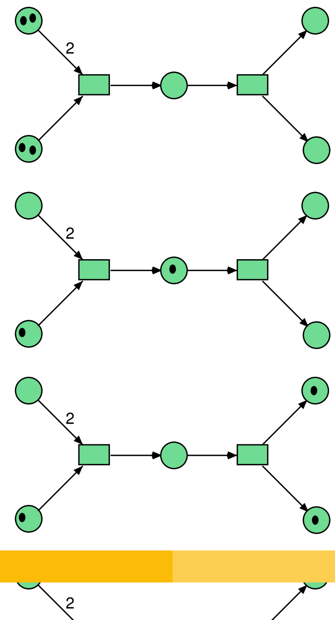
Ist eine Transition aktiviert, so *kann* sie **schalten**. Dabei

1. werden von allen Eingangsstellen Marken entfernt und
2. zu allen Ausgangsstellen Marken gelegt.

Dies geschieht entsprechend der Kantengewichtung.

Petrinetze: Dynamik

Marken auf den Stellen ermöglichen den Transitionen zu **schalten**:



Zwischenfazit

- Statische Struktur:
  - Stellen (Kreise)
  - Transitionen (Vierecke)
  - Kanten - gerichteter Pfeil
    - von Stelle zu Transition oder
    - von Transition zu Stelle
  - Kantengewichte
  - Kapazitäten
  - Marken auf den Stellen
- Dynamik:
  - Transitionen können aktiviert sein
  - aktivierte Transitionen können schalten

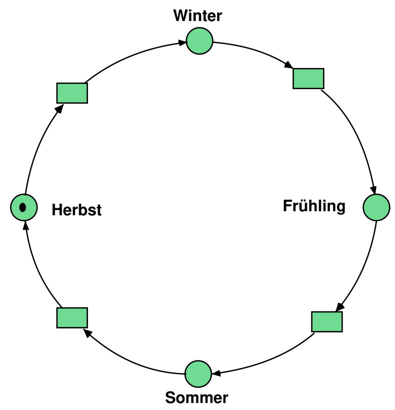
Petrinetze: Beispiel

Einfaches Modell der vier Jahreszeiten:

- Modellierung eines dynamischen Systems als S/T-Netz
- Marke als Zustandsmarker

Besonderheiten dieses einfachen Systems:

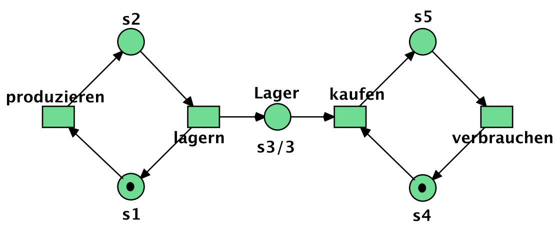
- Genau eine Marke im Netz (**Invariante**)
- Netz stellt **Kausalitäten** dar
- Transitionen schalten **sequentiell**



Nicht alle "Systeme" haben diese Eigenschaften.

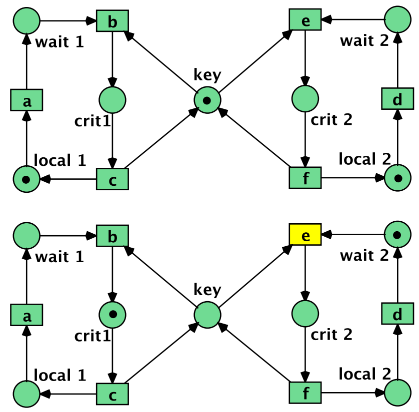
Problemstellung: Producer-Consumer

Ein *Prozess* **produziert** eine Ressource.
Ein (anderer) *Prozess* **konsumiert** eine Ressource.



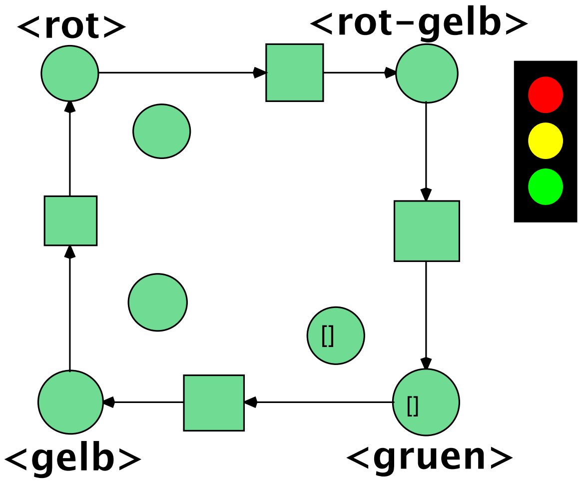
Problemstellung: Wechselseitiger Ausschluss

Zwei Prozesse, die in einem *kritischen Bereich* eine Ressource (alleine!) nutzen wollen.



Ampel-Beispiel: Verschiedene Lösungen

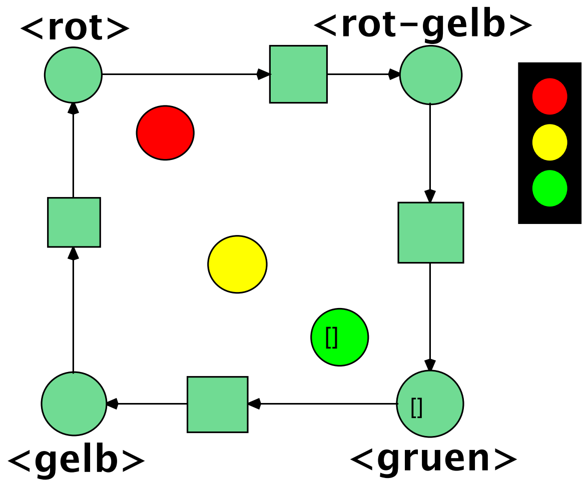
- Modellierung einer einfachen Ampel
- erste Idee



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

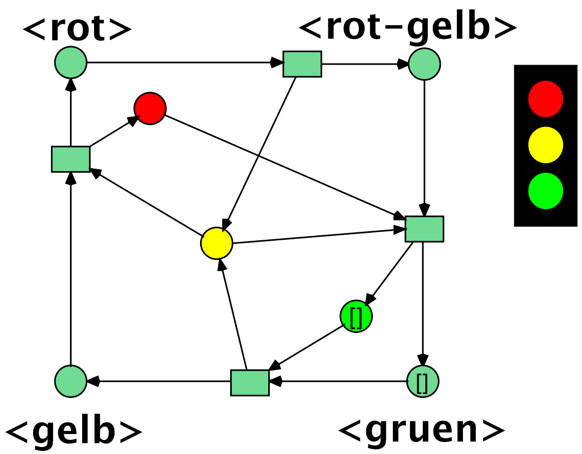
- Modellierung einer einfachen Ampel
- Ampelbezug



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

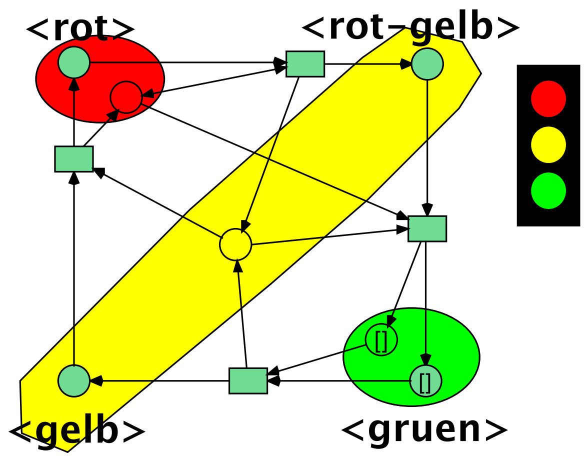
- Modellierung einer einfachen Ampel
- Lösung



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

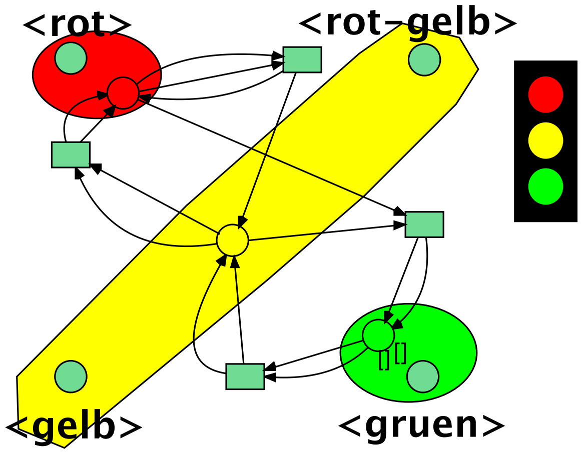
- Modellierung einer einfachen Ampel
- Lösung mit Vergröberung



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

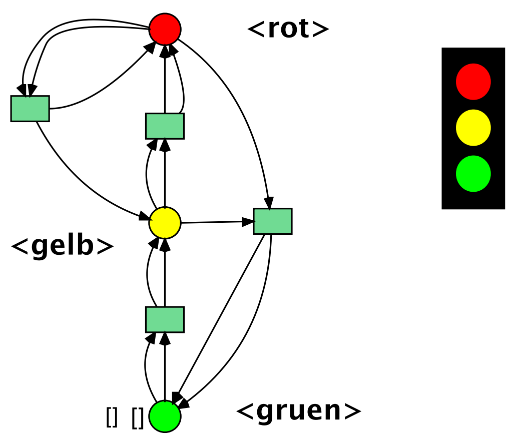
- Modellierung einer einfachen Ampel
- Lösung mit nur drei Stellen



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

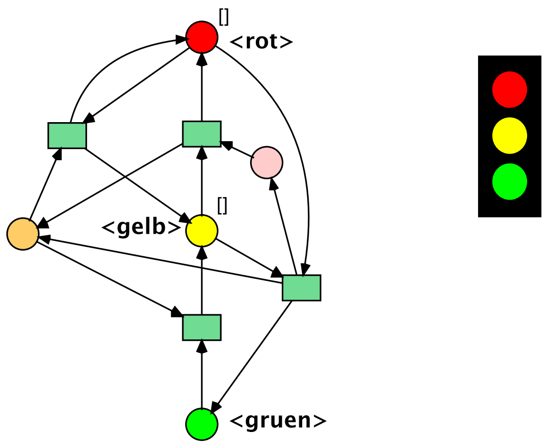
- Modellierung einer einfachen Ampel
- Lösung mit nur drei Stellen anders angeordnet



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

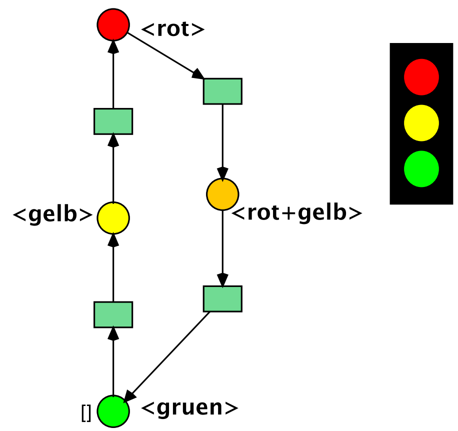
- Modellierung einer einfachen Ampel
- Lösung mit komplementärer Stelle (siehe Kapazitäten)



Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

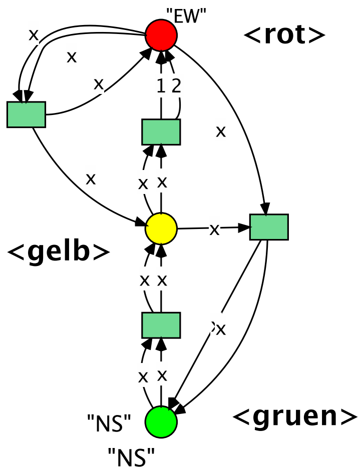
- Modellierung einer einfachen Ampel
- Lösung als endlicher Automat; erfordert andere technische Unterstützung

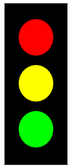


Petrinetze: Beispiel

Ampel-Beispiel: Verschiedene Lösungen

- Modellierung zweier Ampeln für eine Kreuzung
- Lösung für eine Kreuzung mit zwei Richtungen (NS, EW) als gefärbtes Netz





Grundlagen: Mengenlehre

Notation

- Eine Menge ist eine Ansammlung von Elementen z.B. $M_1 = \{1, 2, 3\}$ und $M_2 = \{2, \square\}$
- Enthaltensein eines Elementes: $1 \in M_1$, $1 \notin M_2$
- Teilmenge: $\{1, 2\} \subseteq M_1$, $M_1 \not\subseteq M_2$, $M_2 \not\subseteq M_1$
- Vereinigung: $M_1 \cup M_2 = \{1, 2, 3, \square\}$
- Schnitt: $M_1 \cap M_2 = \{2\}$
- Kartesisches Produkt: $M_1 \times M_2 = \{(1, 2), (1, \square), (2, 2), (2, \square), (3, 2), (3, \square)\}$

Grundlagen: Abbildungen

Notation

Eine Abbildung, notiert als

$$f : A \to B$$

weist einem Element $a \in A$ ein Element $f(a) \in B$ zu.

Beispiele:

- $f : \mathbb{N} \to \mathbb{N}$ mit $n \mapsto n^2$.
- $g : \mathbb{Z} \to \mathbb{N}$ mit $x \mapsto |x|$.
- $h : \{\square, \circ\} \to \{1, 2\}$ mit $h(\square) = h(\circ) = 1$.

P/T-Netz: Definition

Definition (P/T-Netz)

Ein *P/T-Netz* ist ein Tupel $N = (P, T, F, W, m_0)$ mit:

- Einer endlichen Menge P von **Plätzen** (Stellen)
- Einer endlichen Menge T von **Transitionen** mit $P \cap T = \emptyset$
- Einer **Flussrelation** $F \subseteq (P \times T) \cup (T \times P)$
- Einer **Kantenbewertung** $W : (P \times T) \cup (T \times P) \to \mathbb{N}$ mit $W(x, y) = 0 \Leftrightarrow (x, y) \notin F$
- Einer **Anfangsmarkierung** $m_0 : P \to \mathbb{N}$

P/T-Netz: Kapazitäten

Erweiterung um Kapazitäten:

Definition (P/T-Netz mit Kapazitäten)

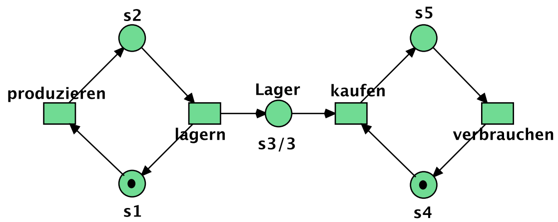
Ein *P/T-Netz mit Kapazitäten* ist ein Tupel $N = (P, T, F, W, K, m_0)$ mit:

- Einem *P/T-Netz* $N' = (P, T, F, W, m_0)$.
- Einer **Kapazitätsfunktion** $K : P \to \mathbb{N} \cup \{\omega\}$
- Zusätzlich gilt: $m_0(p) \leq K(p)$ für alle $p \in P$.

Anmerkung

Ist $K(p) = \omega$, so ist die Markenzahl auf Platz p nicht beschränkt.

Ein Beispiel



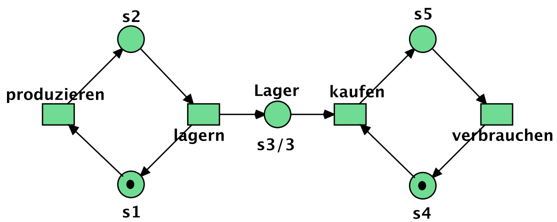
$$\begin{aligned} P &= \{s_1, s_2, s_3, s_4, s_5\} \\ T &= \{lagern, produzieren, kaufen, verbrauchen\} \\ F &= \{(lagern, s_1), (s_1, produzieren), \ldots\} \end{aligned}$$

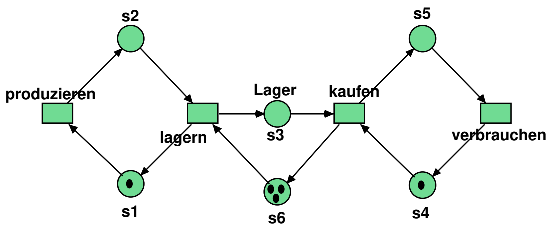
W ist gegeben durch $W(x, y) = 1$ für alle $(x, y) \in F$ und $W(x, y) = 0$ sonst.

m_0 ist gegeben durch $m(s_1) = m(s_4) = 1$ und $m(s_2) = m(s_3) = m(s_5) = 0$.

$K(s_1) = K(s_2) = K(s_4) = K(s_5) = \omega$.

Exkurs: Simulation von Kapazitäten





$\boxed{\text{P/T-Netze mit und ohne Kapazitäten sind äquivalent!}}$

Definition: Aktivierung

Gegeben:

- ein P/T-Netz $N = (P, T, F, W, m_0)$,
- eine Transition $t \in T$ und
- eine Markierung m_1.

Definition (Aktivierung)

Die Transition t ist **aktiviert** in m_1, falls für alle $p \in P$

$$m_1(p) \geq W(p, t)$$

gilt. Notation: $m_1 \xrightarrow{t}$

Definition: Aktivierung 2

Gegeben:

- ein P/T-Netz mit Kapazitäten $N = (P, T, F, W, K, m_0)$,
- eine Transition $t \in T$ und
- eine Markierung m_1.

Definition (Aktivierung (2))

Die Transition t ist **aktiviert** in m_1, falls für alle $p \in P$

$$m_1(p) \geq W(p, t)$$

gilt *und* für alle $p \in P$

$$m_1(p) - W(p, t) + W(t, p) \leq K(p)$$

gilt. Notation: $m_1 \xrightarrow{t}$

Definition: Schalten

Definition (Schalten)

Sei $N_1 = (P, T, F, W, K, m_0)$ ein P/T-Netz, $t \in T$ eine Transition und m_1, m_2 Markierungen. Die Transition t **schaltet** m_1 zu m_2, falls

1. t in m_1 aktiviert ist und
2. $\forall p \in P : m_2(p) = m_1(p) - W(p, t) + W(t, p)$ gilt.

Notation: $m_1 \xrightarrow{t} m_2$. m_2 heißt auch **Folgemarkierung**.

Anmerkung

Kapazitäten werden nicht besonders behandelt. Dies geschieht schon in der Definition der Aktivierbarkeit.

Definition: Schaltfolge

Eine **Schaltfolge** ist ein endliches Wort

$$w = t_1 t_2 t_3 \dots t_n$$

mit $t_i \in T$ für alle $i \in \{1, \dots, n\}$ und $n \in \mathbb{N}$.

Definition (Schaltfolge)

Schaltfolge w schaltet m zu m', falls

1. entweder $w = \lambda$ (leeres Wort mit $n = 0$) und $m = m'$
2. oder $w = (u \cdot t)$ für $u \in T^*$ und $t \in T$, so dass $m \xrightarrow{u} m_1$ und $m_1 \xrightarrow{t} m'$ für eine Markierung m_1 gilt.

Notation: $m \xrightarrow{w} m'$. Ist nur die *Existenz* der Schaltfolge wichtig, so notieren wir auch: $m \xrightarrow{*} m'$.

P/T-Netze: Eigenschaften

Definition (Wichtige Eigenschaften)

1. m ist *erreichbar*, wenn es eine Schaltfolge w gibt mit $m_0 \xrightarrow{w} m$. Die Menge aller erreichbaren Markierungen wird mit $R(N)$ bezeichnet.
2. $t \in T$ ist *aktivierbar*, wenn es eine erreichbare Markierung m gibt mit $m \xrightarrow{t}$.
3. $t \in T$ ist *tot*, wenn t nicht (mehr) aktivierbar ist.
4. $t \in T$ ist *lebendig*, wenn t immer wieder aktiviert werden kann, d.h. wenn es in jeder erreichbaren Markierung m eine Schaltfolge w gibt mit $m \xrightarrow{wt}$. Ein Netz ist lebendig, wenn alle Transitionen lebendig sind.

P/T-Netze: Eigenschaften 2

Definition (Wichtige Eigenschaften)

1. Ein Netz heißt *beschränkt*, wenn zu jedem Platz $p \in P$ eine natürliche Zahl n_p existiert, so dass in jeder erreichbaren Markierung nie mehr als n_p Marken auf p liegen. Ein Netz ist *k-beschränkt* oder *k-sicher*, wenn $n_p = k$ für jeden Platz $p \in P$.
2. Ein Netz ist *rücksetzbar*, wenn m_0 aus jeder erreichbaren Markierung heraus wieder erreichbar ist.

Zur Übung

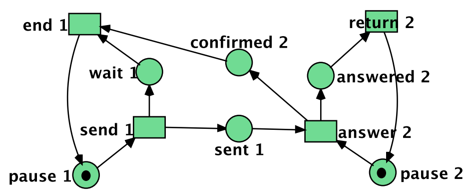
Netze überlegen, die jeweils einige der Begriffe lebendig, beschränkt und rücksetzbar erfüllen, andere nicht (8 Möglichkeiten).

Der Crosstalk-Algortihmus

Der Crosstalk-Algorithmus. Das Setting:

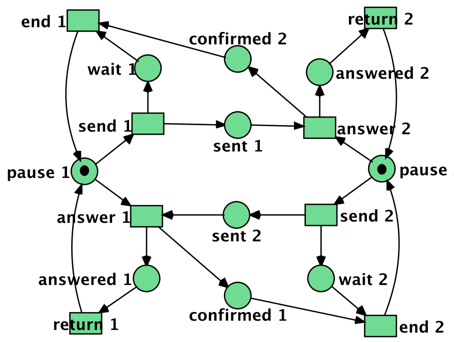
- Zwei Agenten kommunizieren über einen Kanal.
- Der Kanal ist nur bei *einer* Nachricht zuverlässig. Sonst kommt es zu *crosstalk* und die Nachrichten überschreiben sich.
- Der Algorithmus soll crosstalk nicht verhindern, *aber für die Agenten erkennbar machen*.
- Der Algorithmus arbeitet in Runden:
  - In einer Runde sendet entweder ein Agent eine Nachricht, die der andere korrekt empfängt, oder
  - beide Agenten erkennen crosstalk.

Lösungs(ansatz): Crosstalk-Algorithmus



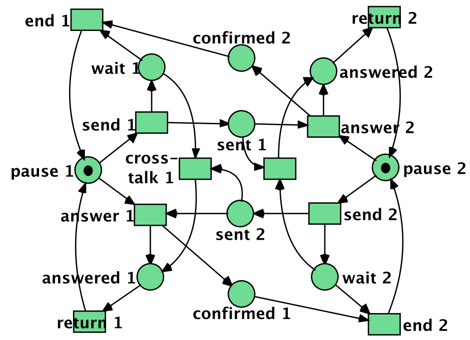
Senden mit Bestätigung.

Lösungs(ansatz): Crosstalk-Algorithmus



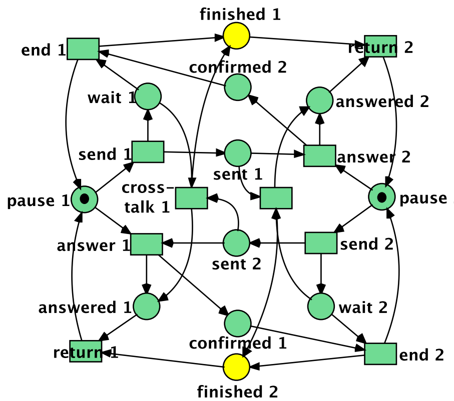
Wechselseitiges Senden mit Bestätigung.

Lösungs(ansatz): Crosstalk-Algorithmus



Crosstalk kann erkannt werden.

Lösungs(ansatz): Crosstalk-Algorithmus



Einführen einer zweiten Bestätigung.
Versehentliches Detektieren von Crosstalk durch nicht abgeholte Bestätigungen wird vermieden.

P/T-Netze: Eigenschaften prüfen

Der Sinn der Formalisierung:

- Eigenschaften sind **exakt ausdrückbar** und
- können formal (!) **nachgewiesen werden**.

Fehler sind weiterhin möglich! Aber die Zuversicht wächst!

Wiederholung und Zusammenfassung

- Petrinetze als graphisches Modell
- Petrinetze als mathematischer Formalismus
- Begriffe:
  - P/T-Netz, Platz, Transition, Kante, Kantengewicht, Kapazität, Startmarkierung
  - Aktiviert, Schalten, Schaltfolge, (Nach-)Folgemarkierung
  - Erreichbar, Aktivierbar, Tot, Lebendig
- Beispiele:
  - Producer-Consumer
  - Wechselseitiger Ausschluss
  - (Crosstalk-Algorithmus)

Literaturhinweis

Mehr zu Petrinetzen in:

- Wolfgang Reisig. *Petrinetze. Modellierungstechnik, Analysemethoden, Fallstudien.* Vieweg+Teubner Verlag, Wiesbaden, 2010.

Außerdem

Werkzeug-Unterstützung Renew (`http://www.renew.de`)

- Entwickelt am Fachbereich (Java Open Source)
- Implementierung / Modellierung (Petrinetze, UML, . . .)
- Verifikation / Validierung / Simulation

Vorlesung: FGI 2 Formale Grundlagen der Informatik II (3. Sem.)

- Petrinetze, Prozessalgebra, unendliche Sprachen
- Verifikation
- Geschäftsprozesse (Workflows)

AOSE Praktika / Projekt: Anwendungen realisieren

- Implementieren durch Modellieren
- Arbeiten im Team
- Reale Einsatzbedingungen / neuste Techniken & Werkzeuge

Schlusswort

Was noch?
School is over now.

Tipps und Tricks (unvollständig!)

- „Vokabeln lernen“
- Lesen, Lesen, Lesen
- Probieren geht über Studieren;-)
- Im Team arbeiten
- Nicht den Veranstalter „austricksen“ durch schummeln / vermeiden / ausweichen; damit trickst man sich nur selbst aus!
- Kernveranstaltungen zuerst; (Pro)-Seminare, Praktika, Projekte zuletzt Weniger ist mitunter mehr.
- Ausgeschlafen zur allen Veranstaltungen gehen
- Das BSc-Studium ist ein mindestens dreijähriges Projekt: Anwendung des informatischen Projektmanagements auf das eigene Studium
- Persönliche Einbindung in den Fachbereich: In der Fachschaft engagieren!

Viel Spaß im Studium!