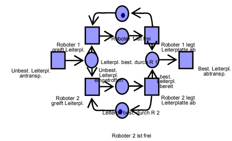
Petrinetze dienen primär der logischen Modellierung von Verhalten



Verhaltensmodellierung:
Beschreibung der Dynamik eines Informationssystems

- mögliche Aktivitäten
- Vor- und Nachbedingungen einer Aktivität
- Zustände (aller Bedingungen)
  (mögliche Ausprägung für jede einzelne Vor- bzw. Nachbedingung: Zustand einer Bedingung ist die Verteilung der Marken auf den Vor- bzw. Nachbereich)
- Anfangszustand
- sequentielle Abläufe (mögliche Folgen von Aktivitäten)
- nichtsequentielle Abläufe
- erreichbare Zustände (konkret auftretende Zustände in einem möglichen Ablauf)
- dynamische Eigenschaften (Invarianten, Erfüllung von Zielen)
- Geschäftsprozesse (mit Anfang, Ende, Varianten, ...)

Grundlagen

Eignung:

Modellierung, Analyse und Simulation von
dynamischen Systemen mit
nebenläufigen und nicht-deterministischen Vorgängen

Systemmodell für Vorgänge, Organisationen und Geräte mit geregelten Flüssen

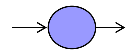
Beschreibungsmittel für Folgeprozesse (dynamische Systeme) mit fester Grundstruktur

Beschreibung von parallelen und zu synchronisierenden Prozessen

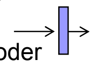
Grundlagen

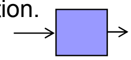
Ein Petri-Netz ist ein bipartiter (d.h. die Menge der Knoten besteht aus zwei disjunktenTeilmengen), gerichteter Graph mit Markierungen in Viertupelform: PN = (S, T, K, M)

s ε S: **Stellen** (oder auch Plätze), zur Beschreibung von Zuständen und/oder Bedingungen, Puffer, Speicher oder Lager, im Graph kreisförmig. Sie dienen der Ablage von Information.



t εT: **Transitionen**, beschreiben Zustandsübergänge, Ereignisse, Aktionen oder Tätigkeiten und sind im Graph strich-, balken- oder quaderförmig. Ihr Zweck ist die Verarbeitung von Information.





k ε K: **Kanten** sind ggf. gewichtete (d.h. mit Zahl versehene) Verbindungen zwischen Stellen und Transition (und zwar nur zwischen S und T, daher ist der Graph bipartit), im Graph pfeilförmig. Sie zeigen den Verlauf der Transitionen an.

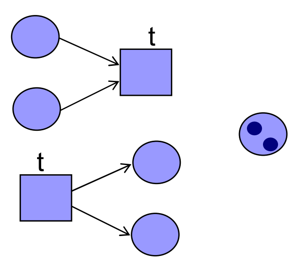
m ε M: **Marken** (Tokens), die den aktuellen Zustand des Petri-Netzes angeben, im Graph als kleiner gefüllter Kreis.

Grundlagen

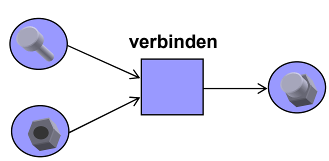
Kanten jeweils nur von Stelle zu Transition bzw. Transition zu Stelle

Eingabestellen von t: Stellen, von denen Kanten zu einer Transition t laufen

Ausgabestellen von t: werden mit Marken durch Weitergabe mittels Transition belegt



Schaltregel: Bewegungsablauf der Marken
- a) Transition t kann schalten, wenn in jeder Eingabestelle von t mindestens eine Marke vorhanden
- b) Schaltet Transition t, Entfernen einer Marke aus jeder Eingabestelle, Hinzufügen einer Marke zu jeder Ausgabestelle

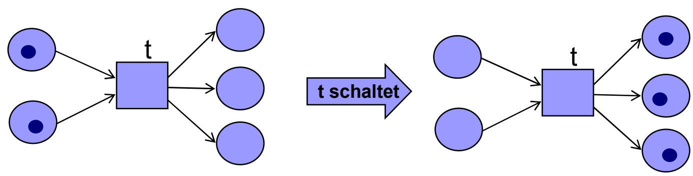


Einfachste Petri-Netze: Bedingungs/Ereignis- Netze

Marken sind vom Typ boole´sch

B/E-Systeme besitzen auf jeder Stelle maximal eine Marke und alle Marken sind gleich.

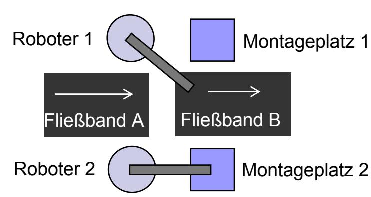
Schaltregel: Transition t kann schalten, wenn jede Eingabestelle von t eine Marke enthält und wenn jede Ausgabestelle von t leer ist (t ist aktiviert).

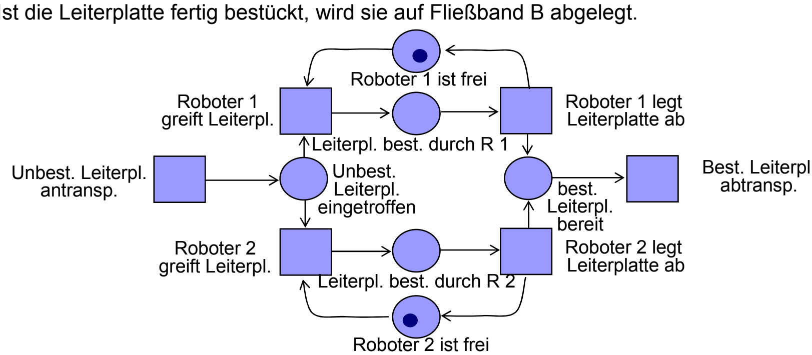


Bedingungs/Ereignis- Netze

Beispiel Bestückungsroboter:

Zwei Roboter bestücken Leiterplatten, die von Fließband A antransportiert werden.
Ist ein Roboter frei, kann er die Leiterplatte nehmen und am Montageplatz bestücken.
Bearbeitet werden unterschiedliche Leiterplatten (unterschiedliche Zeitdauer).
Ist die Leiterplatte fertig bestückt, wird sie auf Fließband B abgelegt.





Bedingungs/Ereignis- Netze

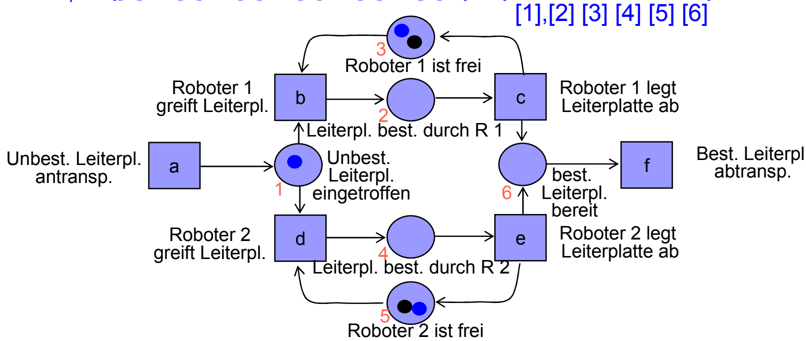
Beispiel Bestückungsroboter

S = {1,2,3,4,5,6}; T = {a,b,c,d,e,f}

Initialzustand: „R1 frei“ und „R2 frei“; keine Transition kann schalten, da keine Vorbedingung erfüllt ist. m_0 = {[1] 0, [2] 0, [3] 1, [4] 0, [5] 1, [6] 0}

Nächster Zustand: Leiterplatte antransportiert; „Transitionen R1 greift Leiterpl.“ und „R2 greift Leiterpl.“ **aktiviert**, Ereignisse können eintreten.

m_1 = {[1] 1, [2] 0, [3] 1, [4] 0, [5] 1, [6] 0} = { 1, 0, 1, 0, 1, 0 }
[1],[2] [3] [4] [5] [6]



Bedingungs/Ereignis- Netze

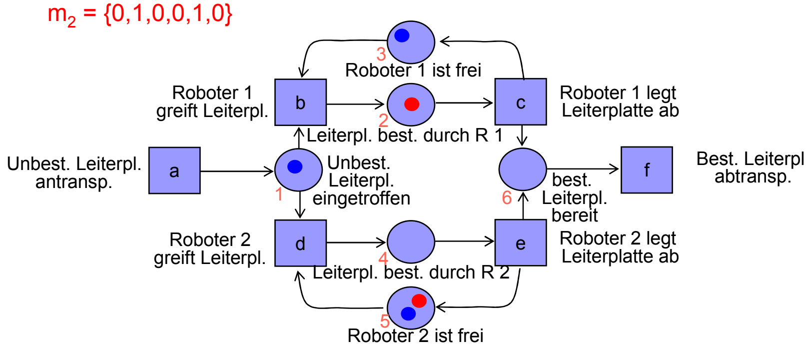
Beispiel Bestückungsroboter

m_1 = {1,0,1,0,1,0}

Konflikt zwischen den gegensätzlichen Transitionen (mind. eine gemeinsame Eingangsstelle), d.h. konkurrierende Ereignisse: Tritt eines der Ereignisse ein, ist die Transition des anderen nicht mehr aktiviert. Eintreten zufällig => Netzverhalten nicht-deterministisch.

Nächster Zustand: Ereignis b: d inaktiviert, da 1 leer; c aktiviert, da 2 belegt.

m_2 = {0,1,0,0,1,0}



Bedingungs/Ereignis- Netze: Dynamik

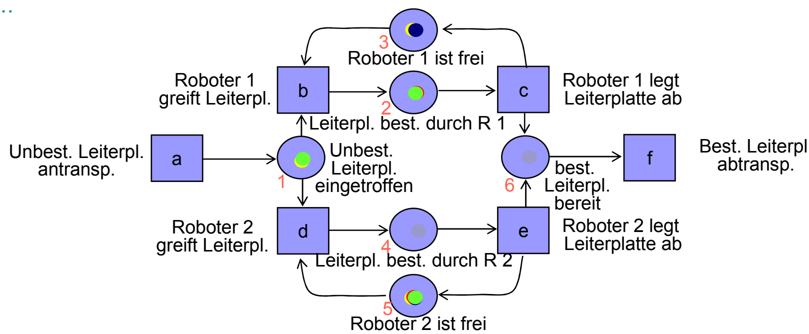
Beispiel Bestückungsroboter

Anfangsmarkierung m_0 = {0,0,1,0,1,0}

Transition a schaltet m_1 = {1,0,1,0,1,0}

Transition b schaltet m_2 = {0,1,0,0,1,0}

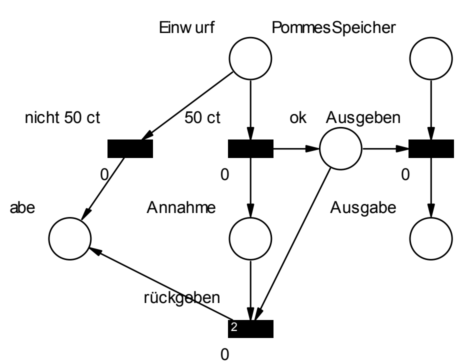
...



Petri-Netze: Modellierung nicht-deterministischer Vorgänge
Sicherstellung, dass nur ein konkurrierender Prozess läuft.

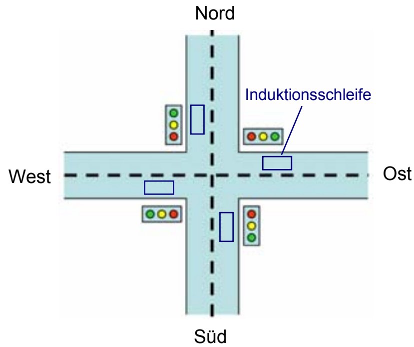
Bedingungs/Ereignis- Netze: Übung 1

Konstruieren Sie ein Petri-Netz eines (rudimentären) Verkaufsautomaten, der eine Portion Pommes frites gegen eine 50-Cent-Münze ausgibt: Der Automat habe einen Münzeinwurf, eine Münzrückgabe, eine Münzannahme und eine Warenausgabe. Nach Annahme einer 50-Cent-Münze kann der Kunde die Rückgabe der Münze oder die Ausgabe der Ware veranlassen.



Bedingungs/Ereignis- Netze: Übung 2

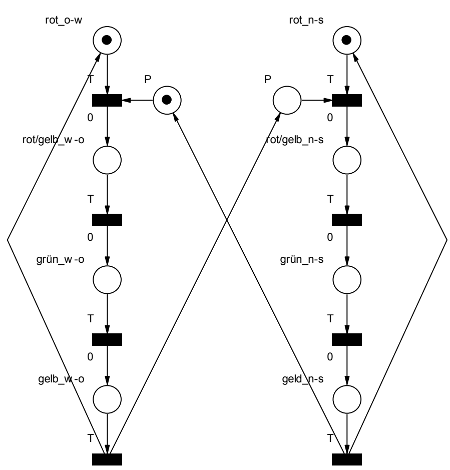
a) Modellieren Sie eine Ampelsteuerung mittels B/E-Netz jeweils mit den Phasen „rot“, „rot-gelb“, „grün“ und „gelb“ für eine Kreuzung mithilfe eines Petri-Netzes, welches gewährleistet, dass jeweils nur eine Straße (Ost-West oder Nord-Süd) freigegeben wird.



b) Erweitern Sie das Petri-Netz so, dass die Induktionsschleifen mit berücksichtigt werden.

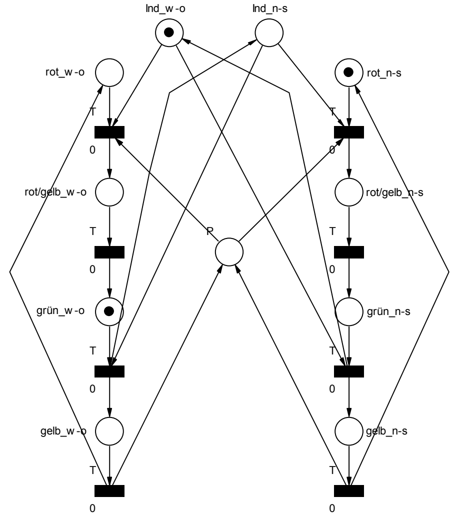
Bedingungs/Ereignis- Netze: Übung 2

Lösung a)



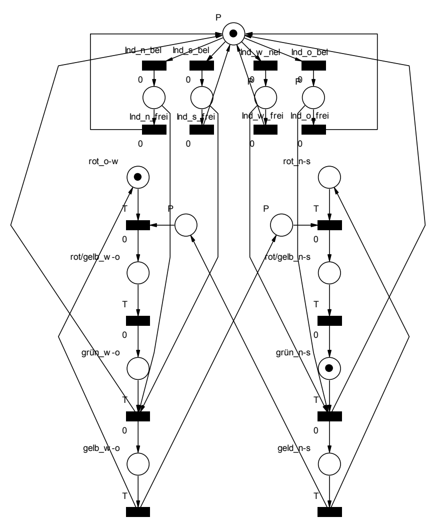
Bedingungs/Ereignis- Netze: Übung 2

Lösung b) Variante 1



Bedingungs/Ereignis- Netze: Übung 2

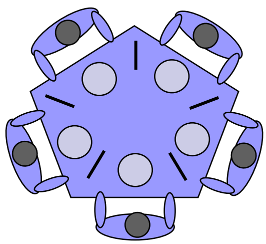
Lösung b) Variante 2



Bedingungs/Ereignis- Netze: Übung 3

Ein philosophisches (filosofisches ?) Problem

Problem der speisenden Philosophen von Dijkstra zur Demonstration von Semaphoren bei nebenläufigen Prozessen:
Tischrunde von 5 Philosophen: Denken nach und müssen ab und an etwas essen. Vor jedem Philosophen ein Teller Reis, zwischen zwei Tellern je ein Stäbchen. Ein Philosoph, der sein Denken unterbricht, um zu essen, benötigt dazu je ein Stäbchen auf beiden Seiten seines Tellers.



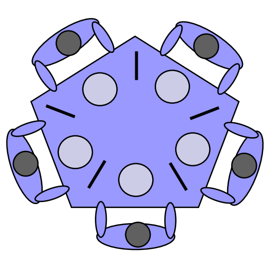
Petri-Netz-Modellierung
1. Definition der Stellen
2. Definition der Transitionen
3. Definition der Beziehungen
4. Vollständiges Netzwerk
5. Plausible Anfangsmarkierung
6. Dynamik
   Terminierung ?
   Lebendigkeit ?
   Verklemmungen ?

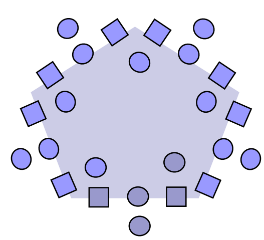
Bedingungs/Ereignis- Netze: Übung 3

Lösung Petri-Netz-Modellierung des Problems der speisenden Philosophen

1. Definition der Stellen
   Philosoph denkend $S1_X$
   Philosoph speisend $S2_X$
   Stäbchen $S3_X$

2. Definition der Transitionen
   Philosoph nimmt Stäbchen und beginnt zu speisen. $T1_X$
   Philosoph legt Stäbchen weg und beginnt zu denken. $T2_X$



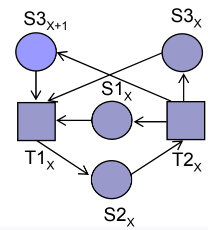


Bedingungs/Ereignis- Netze: Übung 3

Lösung Petri-Netz-Modellierung des Problems der speisenden Philosophen

3. Definition der Beziehungen

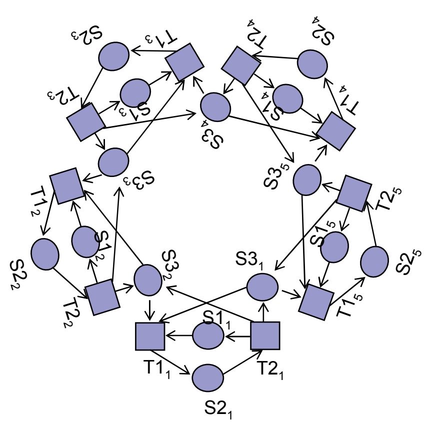
Der Philosoph kann nur zu speisen beginnen ($T1_X$ kann nur schalten), wenn beide benachbarten Stäbchen vorhanden ($S3_X$ und $S3_{X+1}$ belegt) sind und sich der Philosoph im denkenden Zustand befindet ($S1_X$ belegt). Dabei werden $S3_X$, $S3_{X+1}$ und $S1_X$ geleert und $S2_X$ belegt.
Wenn der Philosoph zu denken beginnt ($T2_X$ kann nur schalten, wenn er sich im speisenden Zustand befindet, d.h. $S2_X$ belegt ist), legt er beide benachbarten Stäbchen zurück ($S3_X$ und $S3_{X+1}$ belegt) sind und $S2_X$ wird geleert und $S1_X$ belegt.



Bedingungs/Ereignis- Netze: Übung 3

Lösung Petri-Netz-Modellierung des Problems der speisenden Philosophen

4. Vollständiges Netzwerk

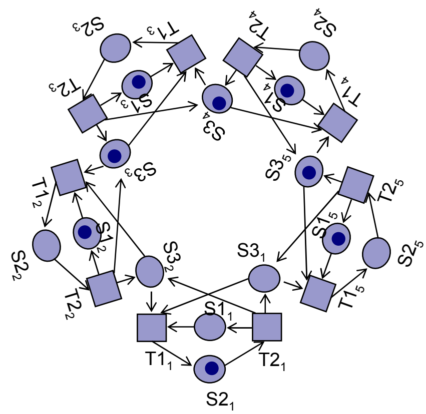


$S1_X$ Philosoph denkend
$S2_X$ Philosoph speisend
$S3_X$ Stäbchen
$T1_X$ Philosoph nimmt Stäbchen und beginnt zu speisen.
$T2_X$ Philosoph legt Stäbchen weg und beginnt zu denken.

Bedingungs/Ereignis- Netze: Übung 3

Lösung Petri-Netz-Modellierung des Problems der speisenden Philosophen

5. Plausible Anfangsmarkierung (eine Möglichkeit)



$S1_X$ Philosoph denkend
$S2_X$ Philosoph speisend
$S3_X$ Stäbchen
$T1_X$ Philosoph nimmt Stäbchen und beginnt zu speisen.
$T2_X$ Philosoph legt Stäbchen weg und beginnt zu denken.

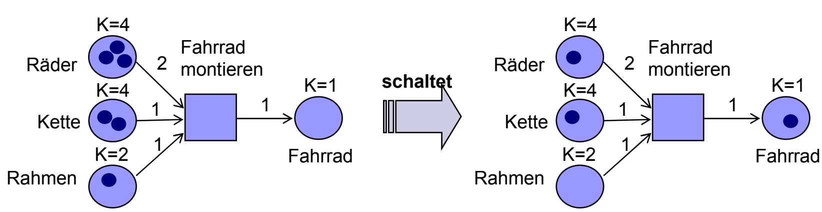
Terminierung ? nein

Lebendigkeit ? ja

Verklemmungen ? nein

Stellen/Transitions-Netze (Markentyp unsigned integer)

- **Auf jeder Stelle mehrere gleiche Marken möglich bis zur Kapazität K der Stelle**
- **Kanten mit Zahlen gewichtet: Transition muss Eingabestellen so viele Marken entnehmen bzw. Ausgabestellen hinzufügen, wie der jeweilige Gewichtswert angibt.**
- **Bei einem Schaltvorgang ggf. mehrere Marken betroffen**
- **Sind alle Eingangszustände einer Transition jeweils mit mindestens der Menge an Marken belegt, die das Gewicht der korrespondierenden Kante angibt, dann ist die Transition "aktiviert".**
- **Nur eine aktivierte Transition kann (muss aber nicht) schalten.**
- **Transition schaltet (zugehöriges Ereignis tritt ein) ➔ Folgemarkierung:**
  - **- Marken werden gemäß Kantengewichten aus Eingabestellen entfernt.**
  - **- Ausgabestellen werden gemäß Kantengewichten gefüllt.**



Stellen/Transitions-Netze

Ein S/T-Netz ist ein bipartiter (d.h. die Menge der Knoten besteht aus zwei disjunktenTeilmengen), gerichteter Graph mit Markierungen in Sechstupelform:

PN = (S, T, K, C, W, M)

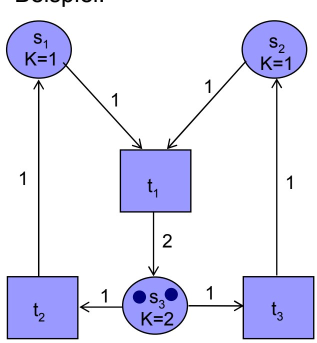
s ε S: endliche, nicht-leere Menge **Stellen** $S = \{s_1, s_2, \ldots, s_n\}$
t ε T: endliche, nicht-leere Menge **Transitionen** $T = \{t_1, t_2, \ldots, t_n\}$
k ε K: nicht-leere Menge **Kanten** mit
$Pre \subseteq S \times T$ Prekanten
$Post \subseteq T \times S$ Postkanten

C: S ➔ N Abbildung, die jeder Stelle eine Kapazität zuordnet
W: K ➔ N Abbildung, die jeder Kante ein Gewicht zuordnet
M: S ➔ N_0 Marken der Anfangsmarkierung

$N = \{1, 2, 3, \ldots\}$
$N_0 = \{0, 1, 2, 3, \ldots\}$

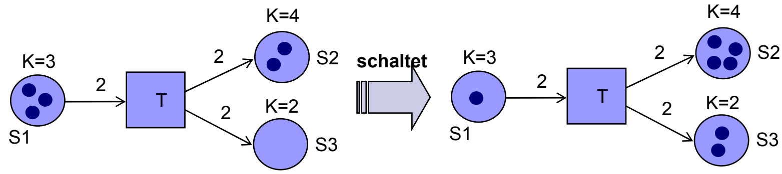
Stellen/Transitions-Netze

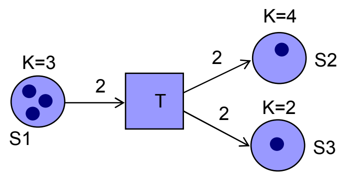
Ein S/T-Netz ist ein bipartiter (d.h. die Menge der Knoten besteht aus zwei disjunktenTeilmengen), gerichteter Graph mit Markierungen in Sechstupelform: PN = (S, T, K, C, W, M)
Beispiel:



$S = \{s_1, s_2, s_3\}$
$T = \{t_1, t_2, t_3\}$
$Pre = \{(s_1, t_1), (s_2, t_1), (s_3, t_2), (s_3, t_3)\}$
$Post = \{(t_1, s_3), (t_2, s_1), (t_3, s_2)\}$
K = Pre U Post
$C(s_1) = 1; C(s_2) = 1; C(s_3) = 2;$
$W(s_1, t_1) = 1; W(s_2, t_1) = 1; W(s_3, t_2) = 1;$
$W(s_3, t_3) = 1; W(t_1, s_3) = 2; W(t_2, s_1) = 1;$
$W(t_3, s_2) = 1;$
$M_0(s_1) = 0; M_0(s_2) = 0; M_0(s_3) = 2;$

Stellen/Transitions-Netze





Kann nicht schalten, da sich in S3 drei Marken ergäben, die Kapazität von S3 aber nur K=2 ist.

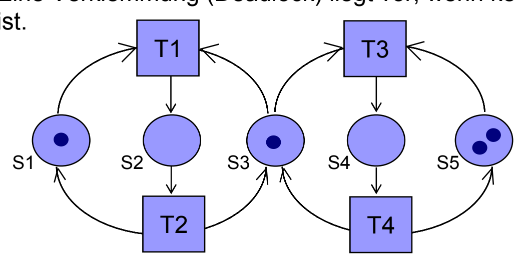
S/T-Netze können B/E-Netze darstellen, wenn Gewichte und Kapazitäten der S/T-Netze gleich eins gesetzt werden.

Stellen/Transitions-Netze

Dynamik

Bei dynamischen Vorgängen kann es zu Nebenläufigkeiten kommen.

1. Aktivierte Transitionen heißen gegensätzlich, wenn sie mindestens eine gemeinsame Eingangsstelle haben.
2. Bei mehreren gegensätzlichen aktivierten Transitionen kann immer nur eine schalten.
3. Aktivierte Transitionen können, falls sie nicht gegensätzlich sind, gleichzeitig schalten.
4. Eine Verklemmung (Deadlock) liegt vor, wenn keine Transition aktiviert ist.



Erzeuger-Verbraucher-System-Modell

S3 stellt Ausschließlichkeit zwischen T1 und T3 sicher.

Stellen/Transitions-Netze

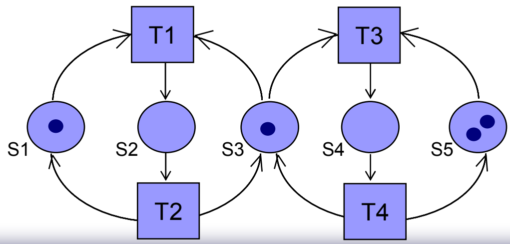
Dynamik

Im Beispiel repräsentiert die Stelle S3 eine Semaphore.

Semaphoren: Signalzähler, mit genau so vielen Einheiten (Marken), wie Prozesse gleichzeitig einen kritischen Abschnitt betreten können.

- Schließen sich die Prozesse gegenseitig aus: Anzahl der Marken = 1.
- Bei Ressourcen, die n Prozesse gleichzeitig bedienen können: Anzahl der Marken = n.

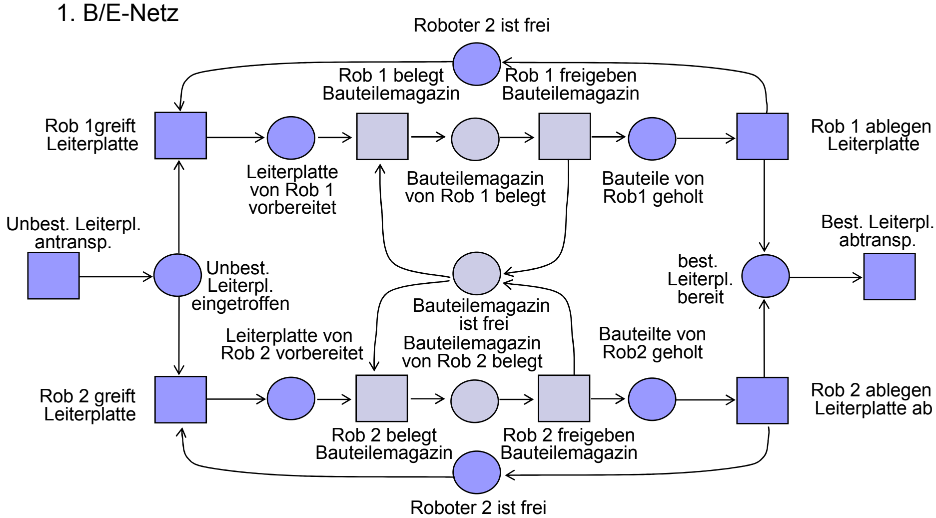
Prozesseintritt in einen kritischen Abschnitt: Verringerung der Semaphore um 1. Zählerwert negativ => Prozess in eine Warteschlange, andernfalls Prozesseintritt. Verlassen des kritischen Abschnitts durch einen Prozess: Erhöhung Zählerwert Semaphore um 1. Prozess in Warteschlange wird durch ein Signal verständigt.



Stellen/Transitions-Netze

Beispiel Erweiterung der Bestückungsanlage um ein Bauteilemagazin

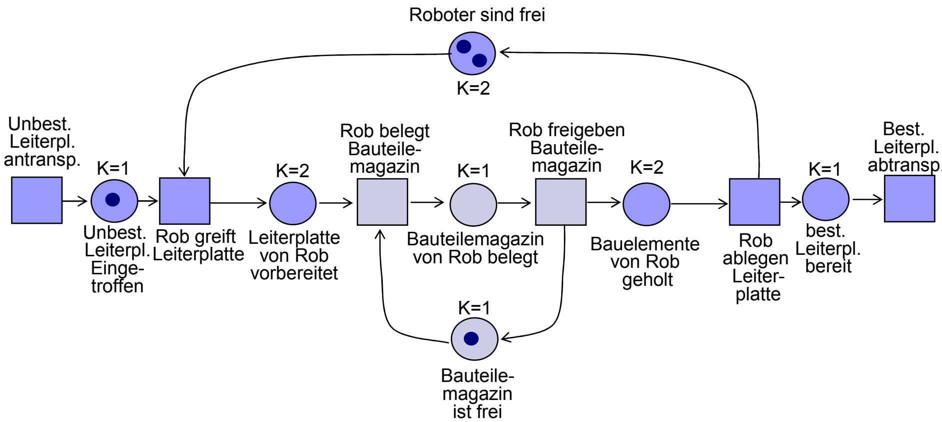
1. B/E-Netz



Stellen/Transitions-Netze

Beispiel Erweiterung der Bestückungsanlage um ein Bauteilemagazin

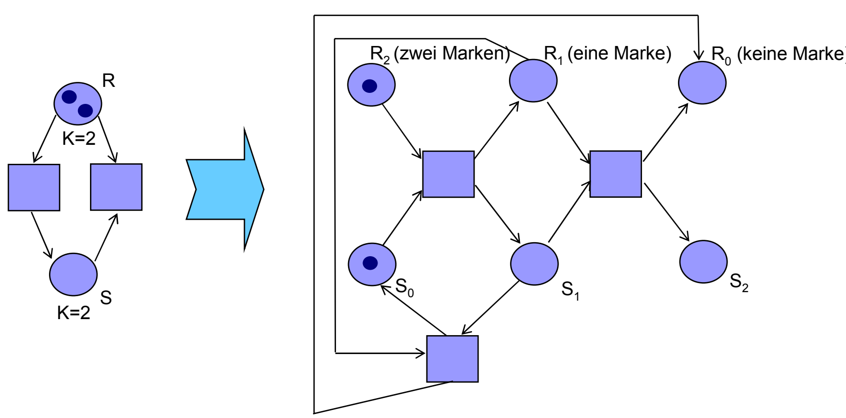
2. S/T-Netz



Stellen/Transitions-Netze

Modellierung von S/T-Netzen durch B/E-Netze

Ersetze Stelle s des S/T-Netzes mit Kapazität K durch K Stellen s_0, s_1,…, s_K eines S/T-Netzes, jeweils für die Belegung der S/T-Stelle durch k=0,…,K Marken.

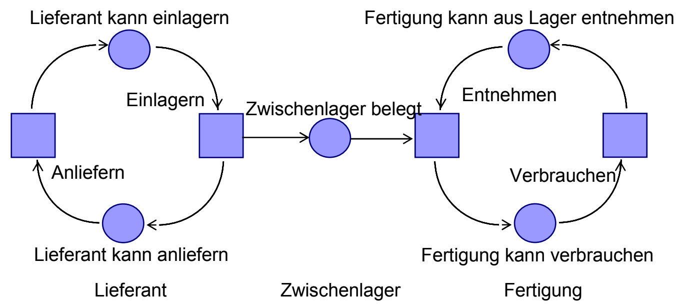


Übungsbeispiele Petri-Netze

Modellierung Erzeuger/Verbraucherproblem durch B/E- und S/T-Netze

Zwischenlagerbeispiel aus dem allgemeinen Erzeuger/Verbraucherproblem: Ein Lieferant liefert Teile an und lagert sie im Zwischenlager ein. Die Fertigung entnimmt die Teile aus dem Zwischenlager und verbraucht sie. Aufgrund der begrenzten Kapazität des Zwischenlagers sind Lieferant und Fertigung miteinander gekoppelt.

B/E-Netz:



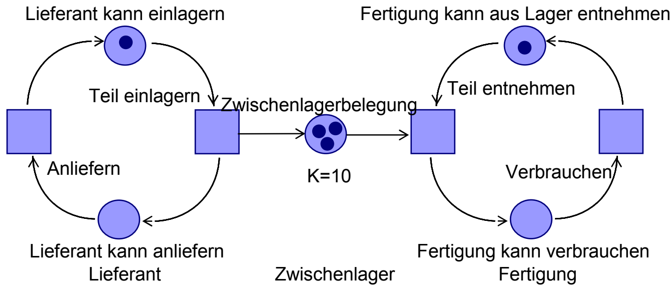
Übungsbeispiele Petri-Netze

Die Lagerkapazität soll vergrößert werden. Geben Sie ein entsprechendes S/ T-Netz mit einer Zwischenlagerkapazität = 10 an.

Übungsbeispiele Petri-Netze

Die Lagerkapazität soll vergrößert werden. Geben Sie ein entsprechendes S/ T-Netz mit einer Zwischenlagerkapazität = 10 an.

Lösung



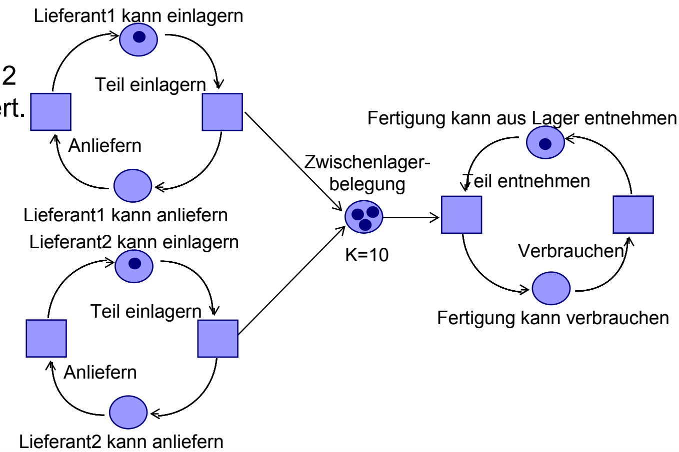
Übungsbeispiele Petri-Netze

Lieferengpässe führen zum Vertragsabschluß mit einem zweiten Liefe-ranten. Geben Sie ein entsprechendes S/T-Netz mit einer Zwischen-lagerkapazität = 10 und zwei Lieferanten an, wobei unterschieden werden kann,
ob Lieferant 1
oder Lieferant 2
ein Teil anliefert.

Übungsbeispiele Petri-Netze

Lieferengpässe führen zum Vertragsabschluß mit einem zweiten Liefe-ranten. Geben Sie ein entsprechendes S/T-Netz mit einer Zwischen-lagerkapazität = 10 und zwei Lieferanten an, wobei unterschieden werden kann, ob Lieferant 1 oder Lieferant 2 ein Teil anliefert.

Lösung



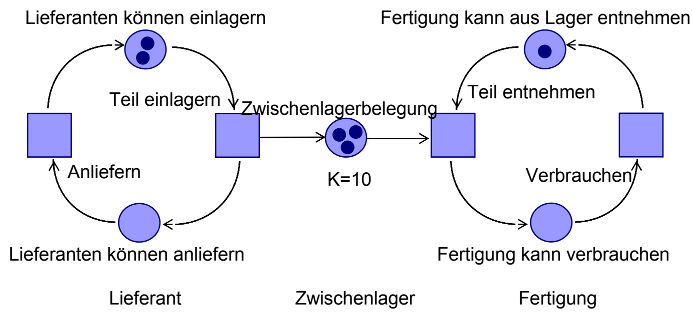
Übungsbeispiele Petri-Netze

Vereinfachen Sie das S/T-Netz von oben, indem Sie nicht mehr unterscheiden, welcher der beiden Lieferanten ein Teil anliefert.

Übungsbeispiele Petri-Netze

Vereinfachen Sie das S/T-Netz von oben, indem Sie nicht mehr unterscheiden, welcher der beiden Lieferanten ein Teil anliefert.

Lösung



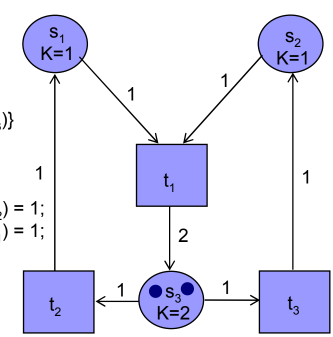
Stellen/Transitions-Netze

Vektorielle Darstellung

PN = (S, T, K, C, W, M)

Beispiel:

$S = \{s_1, s_2, s_3\}$
$T = \{t_1, t_2, t_3\}$
$Pre = \{(s_1, t_1), (s_2, t_1), (s_3, t_2), (s_3, t_3)\}$
$Post = \{(t_1, s_3), (t_2, s_1), (t_3, s_2)\}$
$K = Pre \cup Post$
$C(s_1) = 1; C(s_2) = 1; C(s_3) = 2;$
$W(s_1, t_1) = 1; W(s_2, t_1) = 1; W(s_3, t_2) = 1;$
$W(s_3, t_3) = 1; W(t_1, s_3) = 2; W(t_2, s_1) = 1;$
$W(t_3, s_2) = 1;$
$M_0(s_1) = 0; M_0(s_2) = 0; M_0(s_3) = 2;$



Transitionsindex Stellenindex

Transitionsvektoren $\vec{t}_j = \begin{pmatrix} t_{j1} \\ \vdots \\ t_{jn} \end{pmatrix}$ $\quad mit \quad t_{ji} = \begin{cases} -W(s_i, t_j) \text{ für Prekante} \\ +W(t_j, s_i) \text{ für Postkante} \\ 0 \text{ sonst} \end{cases} \quad und \quad \begin{matrix} i = 1,\dots,n \\ j = 1,\dots,m \end{matrix}$

Netzmatrix $\vec{N} = (\vec{t}_1, \vec{t}_2, \dots, \vec{t}_m) = \begin{pmatrix} t_{11} & t_{21} & \cdots & t_{m1} \\ t_{12} & t_{22} & \cdots & t_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ t_{1n} & t_{2n} & \cdots & t_{mn} \end{pmatrix}$

Kapazitätsvektor $\vec{c} = (c_i) = \begin{pmatrix} C(s_1) \\ \vdots \\ C(s_n) \end{pmatrix}$

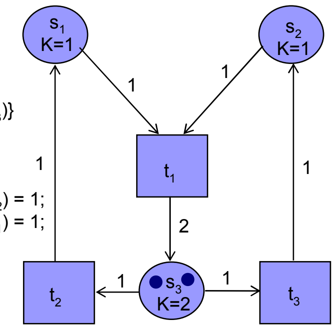
Anfangsmarkierungsvektor $\vec{m}_0 = (m_{0i}) = \begin{pmatrix} M_0(s_1) \\ \vdots \\ M_0(s_n) \end{pmatrix}$

Stellen/Transitions-Netze

Vektorielle Darstellung

PN = (S, T, K, C, W, M)

Beispiel 1:



$S = \{s_1, s_2, s_3\}$
$T = \{t_1, t_2, t_3\}$
$Pre = \{(s_1, t_1), (s_2, t_1), (s_3, t_2), (s_3, t_3)\}$
$Post = \{(t_1, s_3), (t_2, s_1), (t_3, s_2)\}$
K = Pre U Post
$C(s_1) = 1; C(s_2) = 1; C(s_3) = 2;$
$W(s_1, t_1) = 1; W(s_2, t_1) = 1; W(s_3, t_2) = 1;$
$W(s_3, t_3) = 1; W(t_1, s_3) = 2; W(t_2, s_1) = 1;$
$W(t_3, s_2) = 1;$
$M_0(s_1) = 0; M_0(s_2) = 0; M_0(s_3) = 2;$

$$\overleftrightarrow{N} = \begin{pmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \\ 2 & -1 & -1 \end{pmatrix} \quad \vec{C} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix} \quad \vec{m}_0 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$$

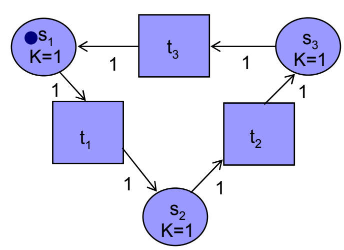
Petri-Netze

Stellen/Transitions-Netze

Vektorielle Darstellung

PN = (S, T, K, C, W, M)

Beispiel 2:



$S = \{s_1, s_2, s_3\}$
$T = \{t_1, t_2, t_3\}$
$Pre = \{(s_1, t_1), (s_2, t_2), (s_3, t_3)\}$
$Post = \{(t_1, s_2), (t_2, s_3), (t_3, s_1)\}$
K = Pre U Post
$C(s_1) = 1; C(s_2) = 1; C(s_3) = 1;$
$W(s_1, t_1) = 1; W(s_2, t_2) = 1; W(s_3, t_3) = 1;$
$W(t_1, s_2) = 1; W(t_2, s_3) = 1; W(t_3, s_1) = 1;$
$M_0(s_1) = 1; M_0(s_2) = 0; M_0(s_3) = 0;$

$$\vec{t}_1 = \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} \quad \vec{t}_2 = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} \quad \vec{t}_3 = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \quad \overleftrightarrow{N} = \begin{pmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{pmatrix} \quad \vec{C} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \quad \vec{m}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Stellen/Transitions-Netze

Schaltregeln und Markenfluss in vektorieller Darstellung

Veränderung des Markierungsvektors durch Schalten: $\vec{m} = \begin{pmatrix} M(s_1) \\ \vdots \\ M(s_2) \end{pmatrix}; \quad \vec{m}' = \vec{m} + \vec{t}_j$

Transition ist aktiviert (schaltfähig), wenn durch Schalten eine zulässige Folgemarkierung erzeugt werden kann: Markenanzahl darf nirgends negativ werden (Vorbedingung), Kapazität darf nirgends überschritten werden: $\vec{0} \le \vec{m} + \vec{t}_j \le \vec{C}; \quad \le gilt\ für\ jede\ Komponente$

Beispiel 1:

$$\vec{N} = \begin{pmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \\ 2 & -1 & -1 \end{pmatrix} \quad \vec{m}_0 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \quad \vec{C} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

t_2 und t_3 sind aktiviert.

$$\vec{m}' = \vec{m}_0 + \vec{t}_2 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}; \quad \vec{m}'' = \vec{m}' + \vec{t}_3 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}; \quad \vec{m}''' = \vec{m}'' + \vec{t}_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$$

$$\vec{m}''' = \vec{m}_0$$

Stellen/Transitions-Netze: Realisierung in C

```
int transition, num_stellen=3, num_transitions=3;
int M[num_stellen], C[num_stellen], N[num_transitions, num_stellen];
bool activity_status(transition)
        {
         int i;
         for (i=0; num_stellen-1; i++)
             {
              if ((M[i]+N[transition,i] < 0) || ((M[i]+N[transition,i] > C[i]))
                { break; }
             }
         if (i==num_stellen-1)
            { return 1; }
         else
            { return 0; }
        }
bool trans_switch(transition)
        {
         for (i=0; num_stellen-1; i++)
             { M[i]=M[i]+N[transition,i] }
        }
do
   {
    for (transition=0; num_transitions-1, transition++)
        {
         if ( acitivty_status(transition) && event(transition))
           {trans_switch(transition);}
        }
   }
```

Stellen/Transitions-Netze
Schaltregeln und Markenfluss in vektorieller Darstellung

Transitionsfolge $\tau = \vec{t}_1', \vec{t}_2', \ldots, \vec{t}_f' \quad mit \quad \vec{t}_i' \in T \;\; für\ alle\ i \in \{1,2,\ldots,p\}$
heisst Schaltsequenz.

Schaltsequenz ist anwendbar, wenn $\forall i \in \{1,2,\ldots,p\} : \vec{0} \leq \vec{m} + \sum_{k=1}^{i} \vec{t}_k' \leq \vec{C}$

Beispiel 1:

$$\vec{N} = \begin{pmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \\ 2 & -1 & -1 \end{pmatrix} \quad \vec{m}_0 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \quad \vec{C} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix} \qquad \tau = \vec{t}_2, \vec{t}_3, \vec{t}_2 \qquad \text{ergibt}$$

$$\vec{m}''' = \vec{m}_0 + \vec{t}_2 + \vec{t}_3 + \vec{t}_2 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} nicht \leq \vec{C} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

=> Schaltsequenz τ nicht anwendbar.

Stellen/Transitions-Netze
Erreichbarkeit

Eine Markierung $\vec{m}$ eines Petri-Netzes heißt erreichbar, wenn eine anwendbare Schaltsequenz τ existiert, die $\vec{m}_0$ in $\vec{m}$ überführt.

Die Erreichbarkeitsmenge ist dann

$$R(\vec{m}_0) = \{\vec{m} \mid \vec{m}\ ist\ erreichbar\}$$

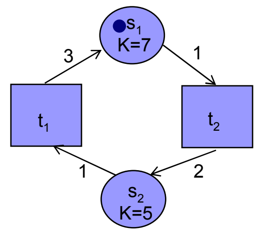
Stellen/Transitions-Netze

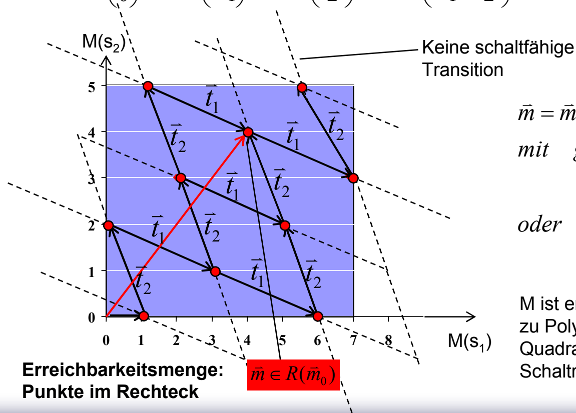
Erreichbarkeit

Geometrische Bedeutung der Erreichbarkeitsmenge

Beispiel:

$$\vec{m}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; \quad \vec{t}_1 = \begin{pmatrix} 3 \\ -1 \end{pmatrix}; \quad \vec{t}_2 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}; \quad \vec{\vec{N}} = \begin{pmatrix} 3 & -1 \\ -1 & 2 \end{pmatrix}; \quad \vec{C} = \begin{pmatrix} 7 \\ 5 \end{pmatrix};$$





Erreichbarkeitsmenge: Punkte im Rechteck

$$\vec{m} = \vec{m}_0 + g_1 \cdot \vec{t}_1 + g_2 \cdot \vec{t}_2 + \ldots + g_m \cdot \vec{t}_m$$

$$mit \quad g_1, g_2, \ldots, g_m \in N_0$$

$$oder \quad \vec{m} = \vec{m}_0 + \vec{\vec{N}}\vec{g} \quad mit \ \vec{g} = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}$$

M ist erreichbar, wenn Transitionsvektoren zu Polygon kombiniert, Ecken im ersten Quadranten und innerhalb Grenzen der Schaltregel.

Stellen/Transitions-Netze

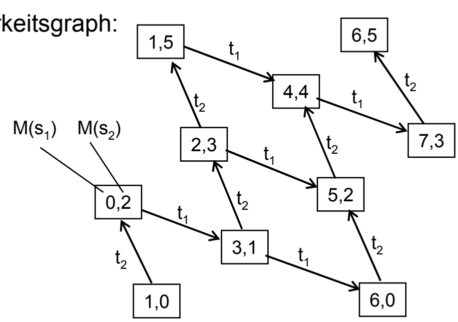
Erreichbarkeit

Reversibles Petri-Netz: $\forall \vec{m}_1, \vec{m}_2 \in R(\vec{m}_0): \ \vec{m}_2 \in R(\vec{m}_1)$

Jeweils zwei Markierungen der Erreichbarkeitsmenge können ineinander übergeführt werden.

$\Rightarrow$ In einem reversiblen Petri-Netz ist die Anfangsmarkierung erreichbar.

Erreichbarkeitsgraph:



Analysemöglichkeiten:

Vektoralgebra
Graphentheorie

Stellen/Transitions-Netze

Analyse

$$\vec{m} = \vec{m}_0 + g_1 \cdot \vec{t}_1 + g_2 \cdot \vec{t}_2 + \ldots + g_m \cdot \vec{t}_m$$
$$mit \quad g_1, g_2, \ldots, g_m \in N_0$$
$$oder \quad \vec{m} = \vec{m}_0 + \vec{N}\vec{g} \quad mit\ \vec{g} = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}$$

Analyse-Grundlage: Gleichungssystem $\vec{N}\vec{g} = \vec{m} - \vec{m}_0$

Notwendige Bedingung für Erreichbarkeit: positive, ganzzahlige Lösung

Notwendige Bedingung für Reversibilität: $\vec{N}\vec{g} = \vec{0}$

Lebendiges und beschränktes Netz: $\exists \min\ ein\ \vec{i} : \vec{N}\vec{i} = \vec{0} \quad mit \quad i_j > 0, j = 1,\ldots,m$

Koeffizienten >0: Alle Transitionen werden berücksichtigt.
Bei jeder Markierung der Erreichbarkeitsmenge ist eine Schaltsequenz vorhanden, die alle Transitionen berücksichtigt und zur Anfangsmarkierung zurückführt.

Analyse und Simulation von Petri-Netzen

Fragen:

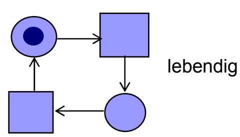
1. Terminiert das Netz?
   Ausgehend von einer Ausgangsmarkierung, können nur endlich viele Transitionen schalten?
2. Ist jede Transition lebendig?
   Ausgehend von einer Ausgangsmarkierung, können die Transitionen stets so schalten, dass eine vorgegebene Transition t im weiteren Verlauf nochmals schalten kann?
3. Treten vermeidbare Verklemmungen auf?
   Gibt es Situationen, in denen keine Transition schalten kann, die aber bei anderer Schaltreihenfolge hätten vermieden werden können?

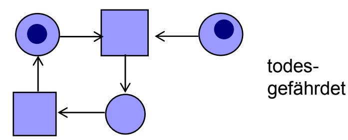
Lebendigkeit von Petri-Netzen

Grade der Lebendigkeit in S/T-Netzen:

1. Todesgefahr: Zustand möglich, bei dem aufgrund leerer Eingangsstellen und voller Ausgangsstellen keine Transition mehr schaltfähig ist. → S/T-Netz mit Markierung M ist lebendig, wenn mindestens eine Transition schalten kann und das Netz mit der Folgemarkierung wieder lebendig ist.
2. Jede Transition muss immer oder immer wieder schalten können.

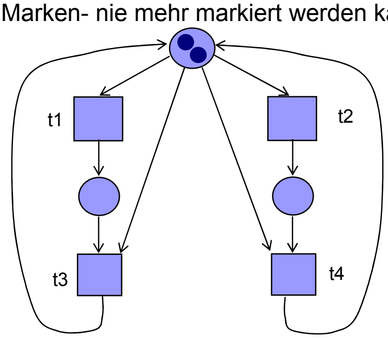
Lebendige Netze: Weder Mangel, noch Überfluss an Marken.
Ein System im Endlosbetrieb kann nur durch ein grad-2-lebendiges Netz beschrieben werden.





Verklemmung in Petri-Netzen

Eine Menge von Stellen S heißt Verklemmung, wenn sie –einmal ohne Marken- nie mehr markiert werden kann.



Verklemmung,
wenn t1 und t2 gleichzeitig schalten.

Totale Verklemmung:
Keine Transition kann mehr schalten.

Partielle Verklemmung:
Nur noch ein Teil aller Transitionen kann aktiviert werden.

Lebendigkeit von Petri-Netzen

Tote Transition

Eine Transition t_j heißt *tot*, wenn sie bei keiner Markierung der Erreichbarkeitsmenge aktiviert ist:

$$\exists \text{kein } M \in R_N(M_0): \quad t_j \text{ aktiviert durch } M$$

Tote Markierung

Eine Markierung *M* heißt *tot*, wenn sie keine Transition aktiviert:

$$\exists \text{kein } t_j \in T: \quad t_j \text{ aktiviert durch } M$$

Eine *tote Transition* führt zu einer *partiellen Verklemmung*.

Eine *tote Markierung* entspricht einer *totalen Verklemmung*.

Lebendigkeit von Petri-Netzen

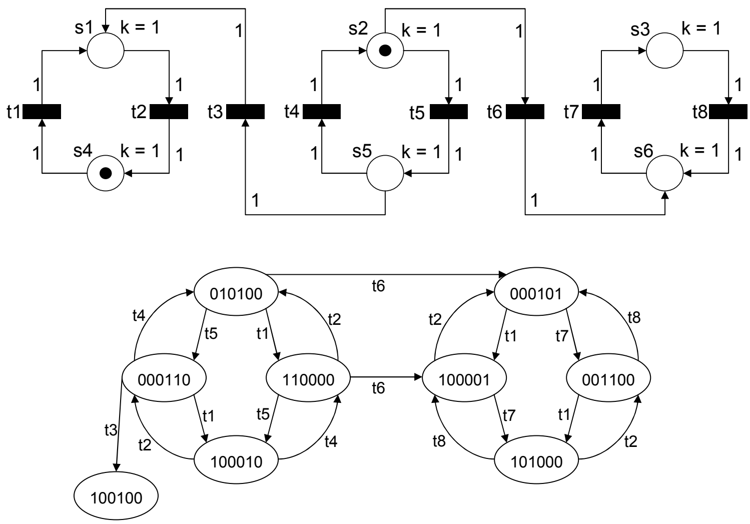
Lebendigkeit

Eine *Transition* t_j heißt *lebendig*, wenn t_j bei jeder Folgemarkierung von M_o aktivierbar ist.

$$\forall M \in R_N(M_0) \ \exists \text{ein } M' \in R_N(M): \quad t_j \text{ aktiviert durch } M'$$

Ein *Netz* heißt *lebendig*, wenn alle Transitionen t_j von *N* lebendig sind.

Analyse mit Erreichbarkeitsgraphen



Analyse mit Erreichbarkeitsgraphen

knotenmenge = {} // Transponierter Anfangsmarkierungsvektor

anzahlTransitionen = n

Anfangsmarkierungsvektor in Ellipse zeichnen

while (knotenmenge.hasNext ())

for (int i = 1; i <= n; i++)

i-te Transition schaltfähig bei aktuellem Markierungsvektor?

Nein | Ja

Schaltergebnis in Knotenmenge?

Nein | Ja

Neue Ellipse mit Schaltergebnis zeichnen

Schaltergebnis in Knotenmenge hinzufügen

Kante zeichnen vom aktuellen Markierungsvektor zum Schaltergebnis und mit j-ter Transition beschriften

Analyse mit Erreichbarkeitsgraphen

Erreichbarkeit einer Markierung
Jede erreichbare Markierung M: Knoten im Erreichbarkeitsgraphen.
Anwendbare Schaltsequenz τ: Weg von der Anfangsmarkierung M_0 zu M im Graphen durch Traversierung in Pfeilrichtung bis zur gewünschten Markierung.

Existenz einer toten Transition
Kanten des Erreichbarkeitsgraphen geben anwendbare Schaltsequenzen wieder. => *Nicht in der Kantenmenge enthaltene Transitionen sind tot.*

Existenz einer toten Markierung
Tote Markierungen sind *Knoten im Erreichbarkeitsgraph, die keine auslaufende Kante besitzen*, also keine Transitionen aktivieren: totale Verklemmung

Konfliktsituation
Konfliktsituation: Gleichzeitig sind mehrere Transitionen aktiviert und durch das Schalten einer Transition wird mindestens einer anderen Transition die Aktiviertheit entzogen. Im Erreichbarkeitsgraphen: notwendige Bedingung, dass von einem Knoten mehr als eine auslaufende Kante existiert. Hinreichende Bedingung, dass nach dem Schalten der Transition in der Folgemarkierung nicht mehr alle anderen Transitionen, die vorher auch aktiviert waren, noch aktiviert sind. Überprüfung im Graph: *Vergleich der Menge der auslaufenden Kanten beim jetzigen Knoten mit den auslaufenden Kanten beim vorigen Knoten verglichen. Fehlt eine auslaufende Kante, außer natürlich die eben traversierte Kante, dann ist die Konfliktsituation erfüllt.*

Beschränktheit und Sicherheit in Petri-Netzen

Ein Stelle S heißt beschränkt bezüglich einer Markenanzahl Z, wenn

$$Z \in N, \forall\, M \in R(\vec{m}_0): \quad M(s_i) \le Z$$

Ein Netz heißt beschränkt bezüglich Z, wenn alle Stellen s_i ε S beschränkt bezüglich Z sind.

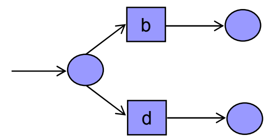
Ein Netz heißt sicher, wenn es beschränkt bezüglich Z=1 ist.

Gefährdung der Lebendigkeit

Gefährdung der Lebendigkeit

durch Verzweigung: Konflikt

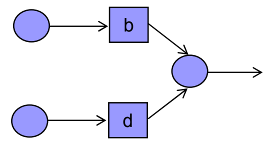
Schalten einer Transition entzieht der anderen ihre Aktiviertheit.



Gefährdung der Sicherheit

durch Zusammenführung: Kontakt

Schalten einer Transition durch die Markierung und Kapazität des Nachbereichs verhindert.

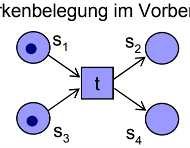


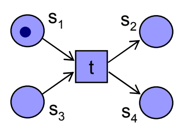
Kontaktfreies Netz:

$$\forall M \in R_N(M_0),\ \forall t_j \in T : \{\vec{m} + \vec{t}_j \geq 0 \Rightarrow \vec{m} + \vec{t}_j \leq \vec{C}\}$$

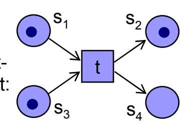
Je nach Markenbelegung im Vorbereich kann dasselbe Netz kontaktfrei oder –behaftet sein:

Kontaktfrei:





Kontakt-behaftet:

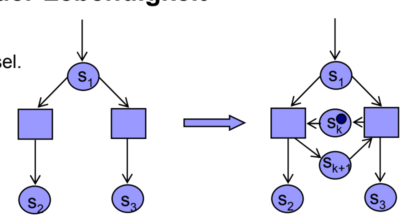


Beseitigung von Gefährdung der Lebendigkeit

Konfliktbeseitigung

Transitionen t_1 und t_2 schalten im Wechsel.

t_1 schaltet zuerst.

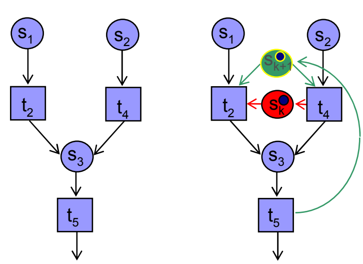


konfliktbehaftet konfliktfrei

Kontaktbeseitigung

Stelle s_{k+1} wandelt Kontakt in Konflikt.

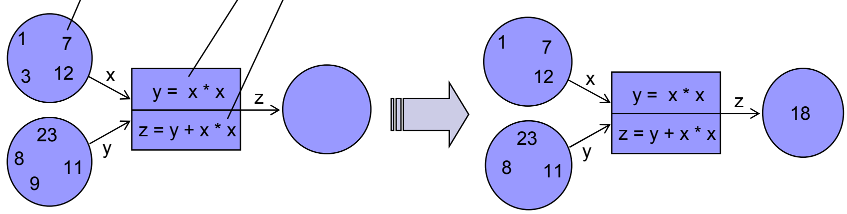
Stelle s_k löst Konflikt durch Priorisierung; t_2 hat Vorrang vor t_4.



kontaktbehaftet kontaktfrei

Prädikat/Transitions-Netze

- Unterscheidbare Marken (repräsentiert z.B. durch Zahl)
- Transitionen nur bei Vorliegen bestimmter Marken schaltfähig
  Den Transitionen werden Schaltbedingungen mit Variablen zugeordnet, die die von den Eingabestellen erhaltenen Marken repräsentieren.
  Symbole: Variablenname an Pfeil
  Schaltbedingung im oberen Teil der Transition
- Schaltvorgang: Verbrauch eingehender Marken der Eingabestellen, Erzeugung neuer Marken in Ausgabestellen
  Symbole: Variablenname an Pfeil
  Berechnungsvorschrift im unteren Teil der Transition



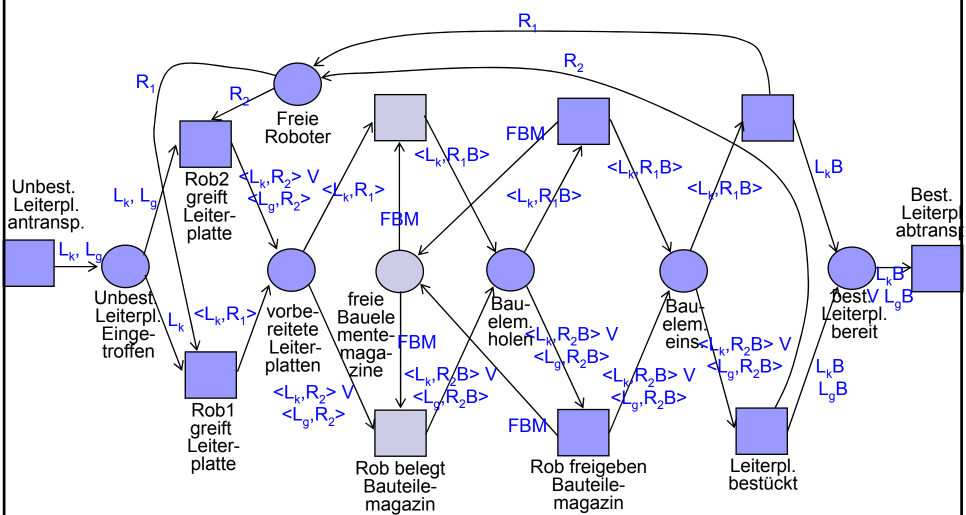
Prädikat/Transitions-Netze

- Schalten einer Transition nicht von bestimmten Marken abhängig: Schaltbedingung entfällt.
- Marken durch Transition nicht verändert: Schaltwirkung entfällt.
- Konstante Pfeilbeschriftung einer Transition: nur entsprechend bezeichnete Marken werden übergeben.
- Variable Pfeilbeschriftung: Weitergegebene Marken müssen zum Wertebereich der angetragenen Variablen gehören

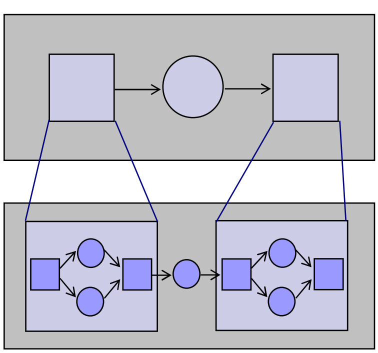
Beispiel Bestückungsroboter:

6 Leiterplattentypen: 3 kleine $L_k = \{L_1, L_2, L_3\}$, 3 große $L_g = \{L_4, L_5, L_6\}$, alle mit gleichen Bauelementen B bestückt.
2 Roboter: R1 für kleine Leiterplatten, R2 für kleine und große Leiterplatten.
Anzahl Bauelementemagazine
Marken: L_1, L_2, L_3, L_4, L_5, L_6, B, R1, R2, FBM

Prädikat/Transitions-Netze: Beispiel



Hierarchische Petri-Netze

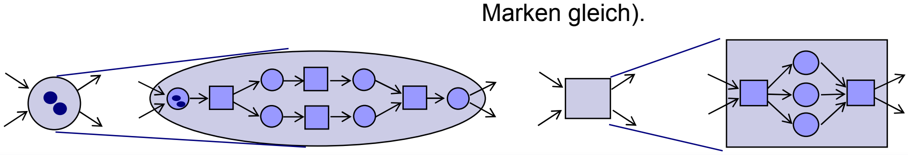


Hierarchische Modellierung durch

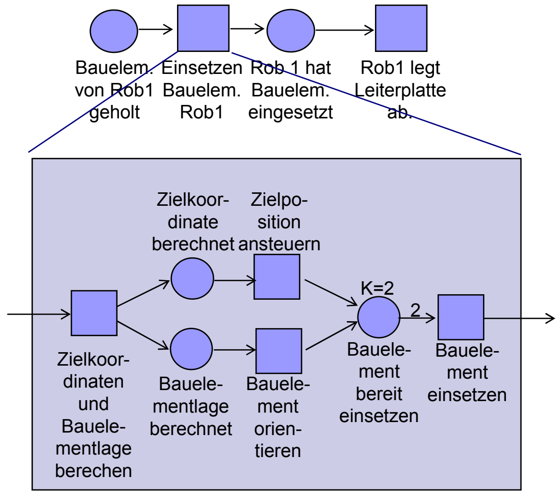
Zusammenfassung von Netzknoten, Verfeinerung von Kanten

Regeln zur Konsistenzerhaltung

- Zusätzl. Pfeile nur **innerhalb** eines Baumknotens (zw. Baumknoten auf tieferen Ebenen keine neuen Pfeile)
- Pfeile zu und von Unternetzen dürfen durch Verfeinerung nicht verändert werden.
- Verfeinerung muss markengetreu erfolgen (Abgabe und Verbrauch von Marken gleich).



Hierarchische Petri-Netze: Beispiel

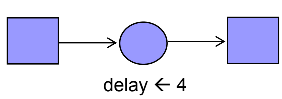


Petri-Netze

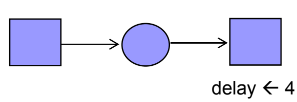
Zeitbehaftete Petri-Netze

Bisher: Alle Transitionen zum gleichen Zeitpunkt schaltfähig, an dem Bedingungen erfüllt sind.

Für System-Leistungsmessung und –Simulation: Dauer von Aktion oder Ereignis (Verharren der Marken auf einer Stelle vor Verbrauch durch Transition).



Marke steht nach Verzögerung von vier Einheiten zur Verfügung.



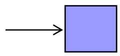
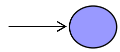
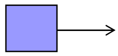
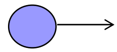
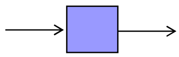
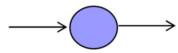
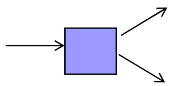
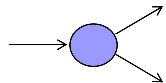
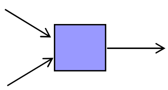
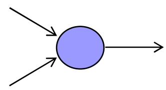
Transition schaltet vier Einheiten nach Eintreffen einer Marke in Eingangsstelle.

Festlegung der Zeitintervalle in Stellen oder Transitionen, Verharren der Marken in den Stellen.

Zeitattributierte Netze: Transitionen mit deterministischem Zeitverbrauch
Stochastische Netze: Transitionen mit stochastischem Zeitverbrauch

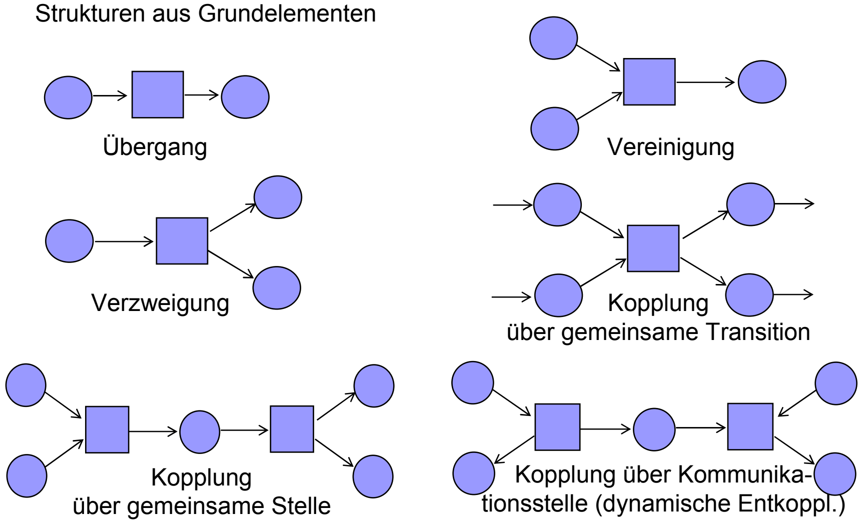
Elemente und Strukturen von Petri-Netzen

10 Strukturelemente

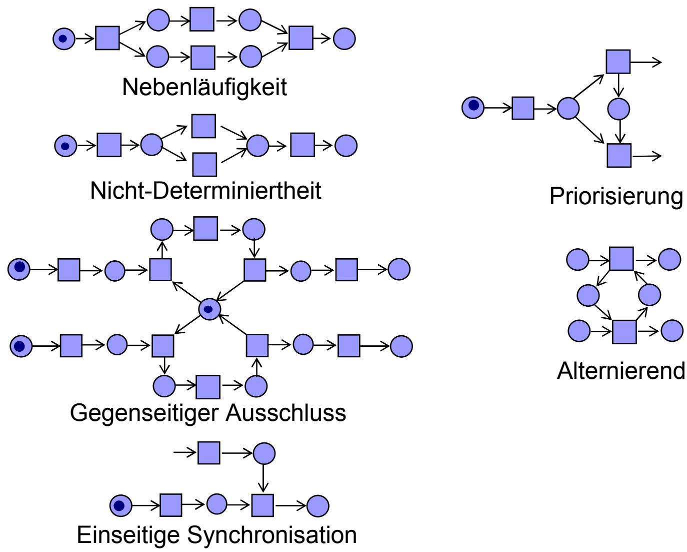
	Objekt löschen		Tote Stelle, ablegen in Archiv
	Objekt erzeugen		Quelle, Reservoir von Objekten
	Weitergabe/ Verarbeitung von Obj.		Zwischenablage, -speicher
	Aufspalten/ Vervielfachen Obj., Beginn Nebenläufigk.		Willkürl. Verzweigung, nicht-det. Fortsetzung eines Prozesses und/oder Beginn Nebenläufigk.
	Verschmelzen Objekte, Ende Nebenläufigk., Synchronisa-tionspunkt		Gemeinsamer Speicher für Objekte, Synchronisationsstelle

Elemente und Strukturen von Petri-Netzen

Strukturen aus Grundelementen



Anwendungsmuster von Petri-Netzen



Petri-Netze

Anwendung von Petri-Netzen

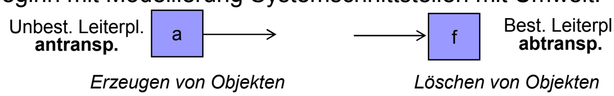
Vorgehen (Überblick)

1. Aktive (Transitionen) und passive (Stellen) Komponenten identifizieren
2. Beziehungen zwischen den Komponenten ermitteln
3. Verfeinerung und Ergänzung
4. Festlegung der Objekte
5. Überlegungen zu Schaltregeln und Schaltwirkungen
6. Netztyp festlegen
7. Anfangsmarkierung festlegen
8. Analyse, Simulation

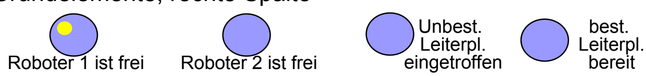
Anwendung von Petri-Netzen

1. Aktive (Transitionen) und passive (Stellen) Komponenten identifizieren

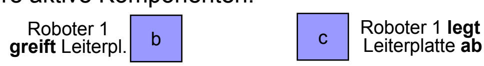
Aktiv: Erzeugung, Transport oder Veränderung von Objekten;
Beschreibung durch Verben (siehe Grundelemente, linke Spalte);
Beginn mit Modellierung Systemschnittstellen mit Umwelt:



Passiv: Komponenten in bestimmten Zuständen zur Lagerung, Speicherung oder Sichtbarmachung von Objekten; Beschreibung durch Substantive oder Zustandsaussagen (Ja/nein-Antwort); siehe Grundelemente, rechte Spalte



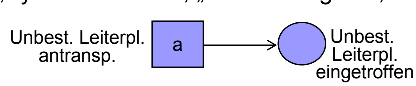
Weitere aktive Komponenten:

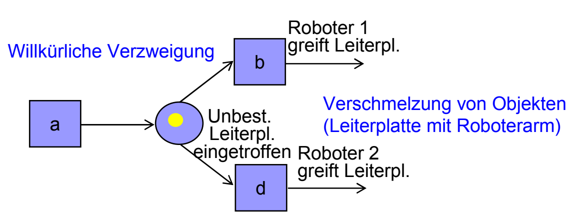


Anwendung von Petri-Netzen

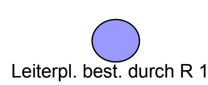
2. Beziehungen ermitteln

Pfeile beschreiben abstrakte, gedankliche Beziehung zwischen aktiven und passiven Komponenten (siehe Tabelle Strukturen), wie z.B. „nicht-deterministische Fortsetzung", „Vereinigung", „Verzweigung", „Synchronisation", „Nebenläufigkeit", ...



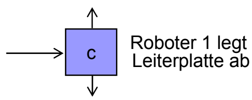


Bestückungsprozess wird nicht betrachtet:

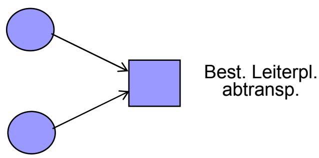


Anwendung von Petri-Netzen

2. Beziehungen ermitteln



Aufspalten von Objekten



Vereinigung von Objekten

Anwendung von Petri-Netzen

3. Verfeinerung und Ergänzung
   Verfeinerung von Transitionen und Stellen
   Ergänzung des Netzes
4. Festlegung der Objekte
   Welche konkreten Objekte können die Transitionen und Stellen enthalten? Anonyme Objekte oder individuelle Objekte?
5. Schaltregeln und Schaltwirkungen
   Ergeben sich aus Überlegungen zu Objekten.
6. Netztyp festlegen
   Netztyp ergibt sich aus 5.
   Entsprechend Objekt- und Pfeilbeschriftungen durchführen.
7. Anfangsmarkierung festlegen
8. Analyse und Simulation