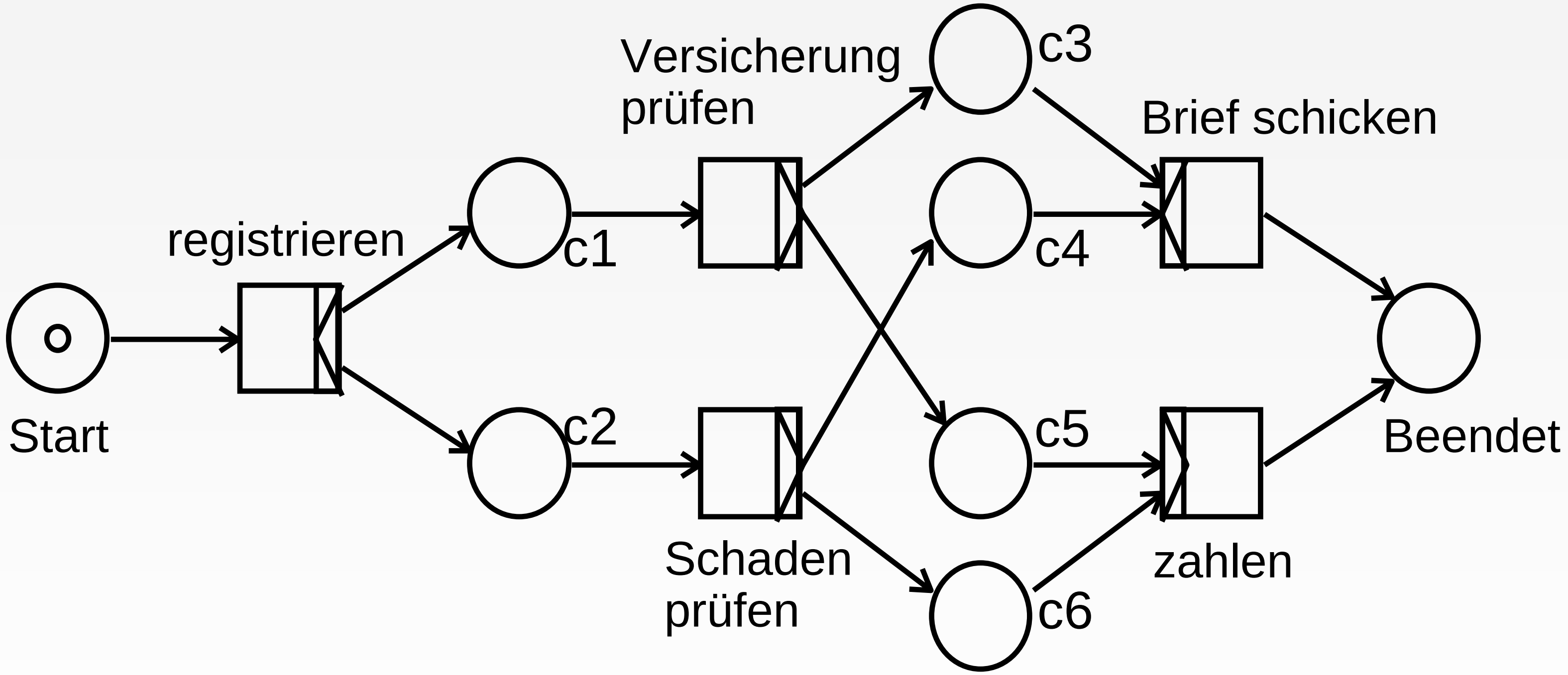
Kapitel 4: Analyse von Petrinetzen

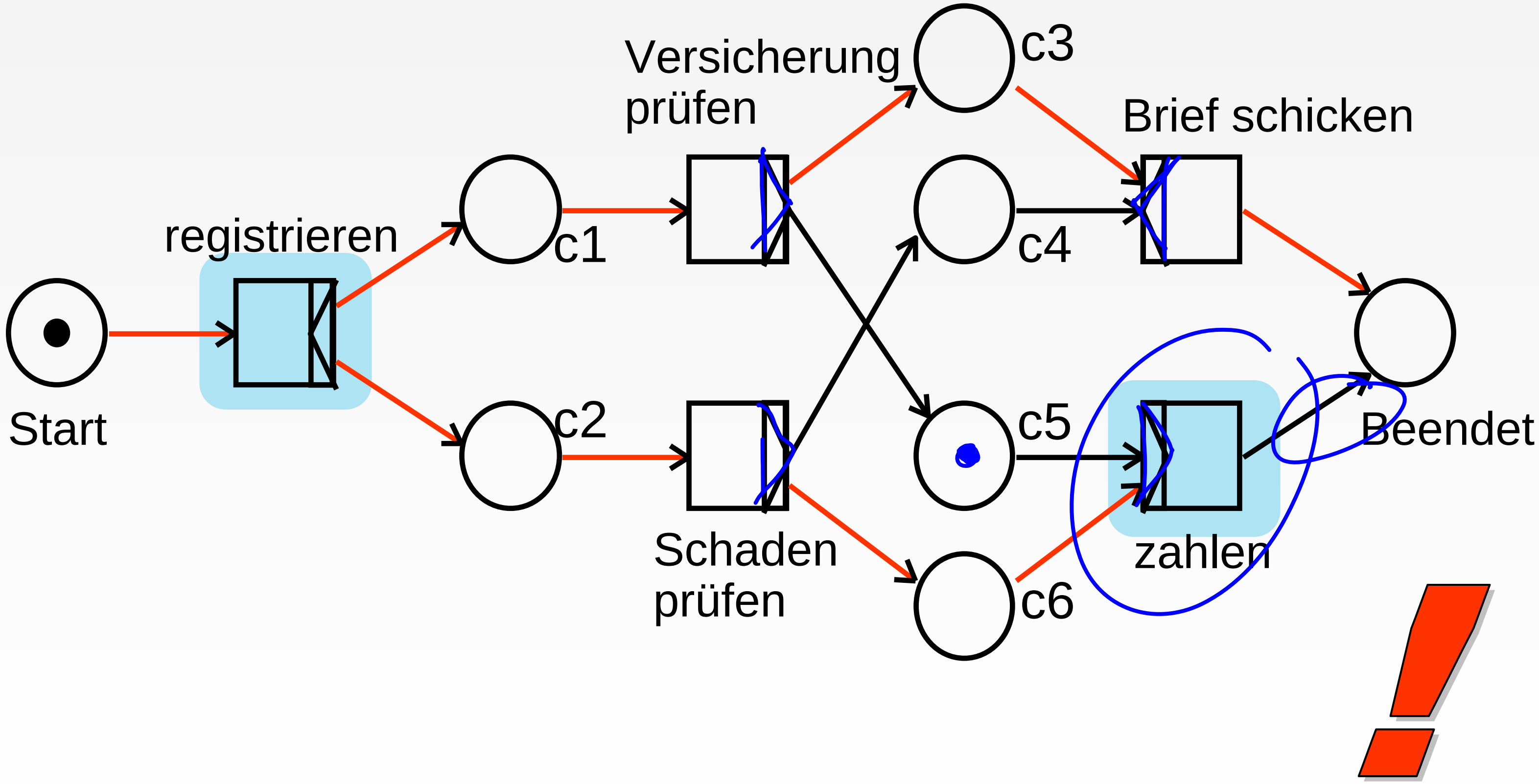
1. Beispiele
2. Analyseansatz
3. Markierungsgraph
4. Beschränktheit
5. State Space Explosion: Beispiel
6. Komplementbildung
7. Zusammenhängend
8. Tot, lebendig, verklemmungsfrei
9. Eigenschaften in Sätzen
10. Stelleninvarianten // Semantik des Systemverhaltens
11. Free Choice !

4.1 Ein Beispiel



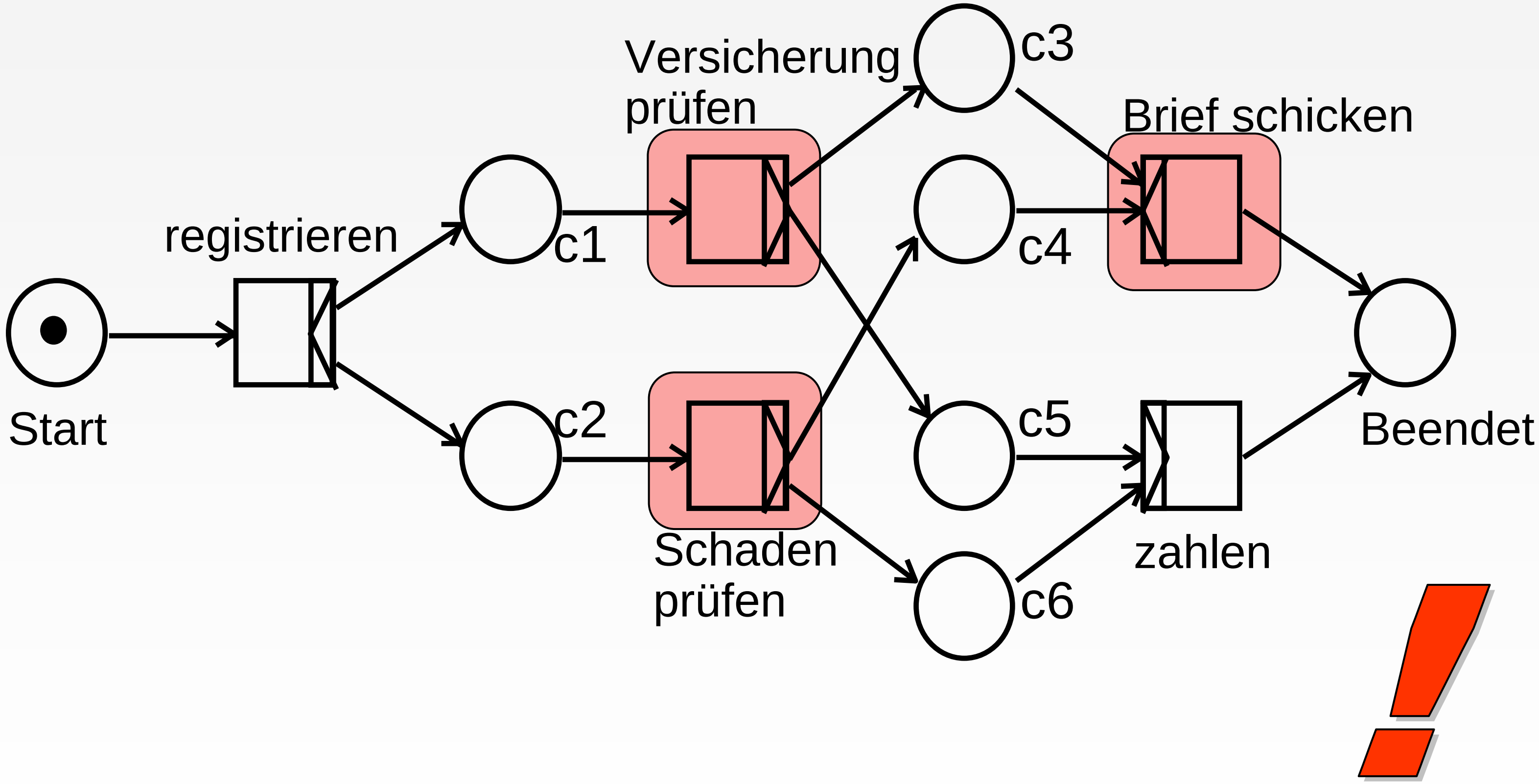
Ist die Prozessdefinition korrekt?

4.1 Ein Beispiel

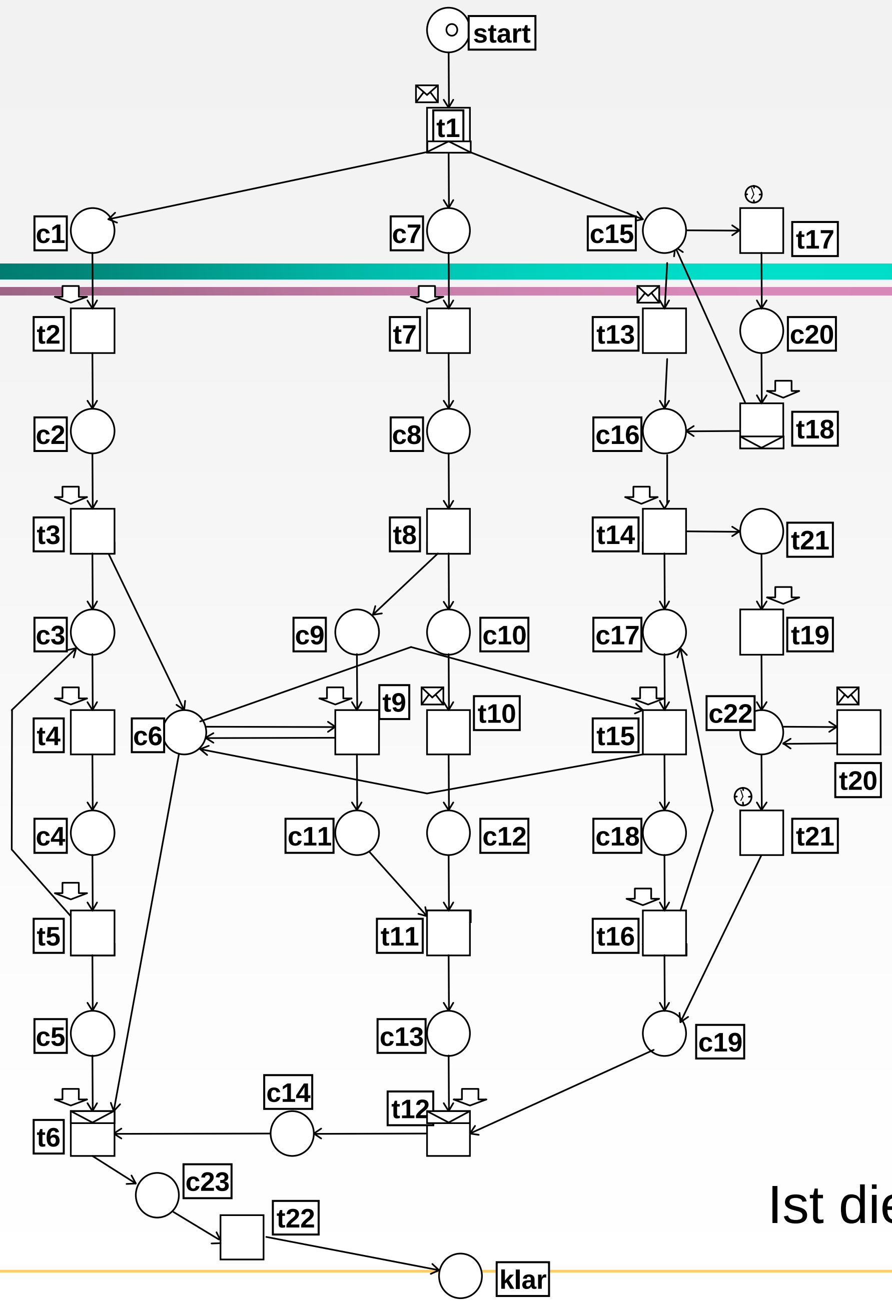


Ist die Prozessdefinition korrekt?

4.1 Ein Beispiel



Ist die Prozessdefinition korrekt?



4.1 Noch ein Beispiel

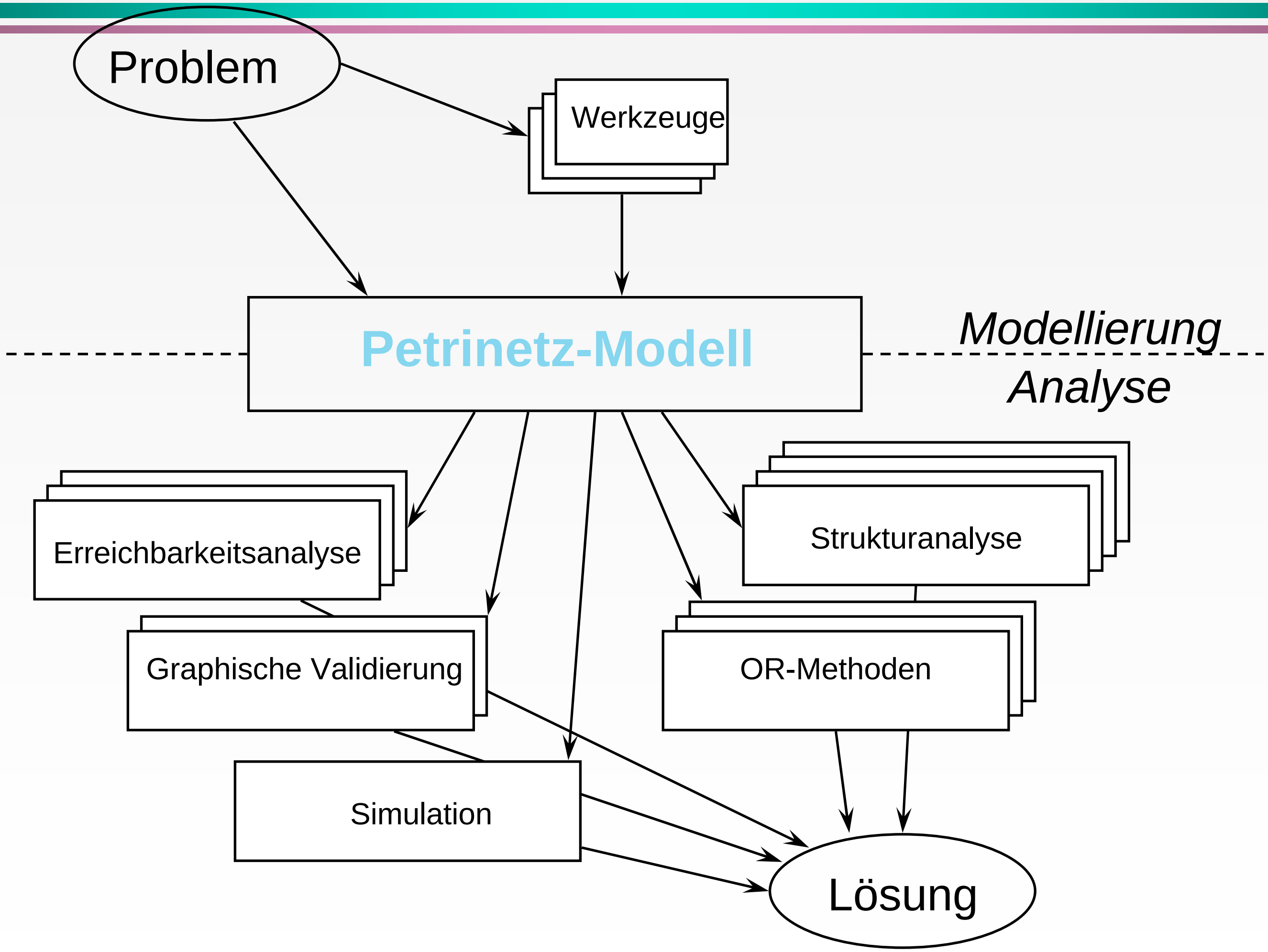
Ist die Prozessdefinition korrekt?

4.2 Analyse-Ansatz

- Prozessdefinitionen können nur für einfache Beispiele „von Hand“ überprüft werden

- Daher: Petrinetz-basierter Ansatz
  - » erfüllt Anforderungen an Modellierungssprache
  - » viele Methoden zur Analyse
  - » viele Werkzeuge zur „automatischen“ Verifikation

4.2 Analyse-Ansatz



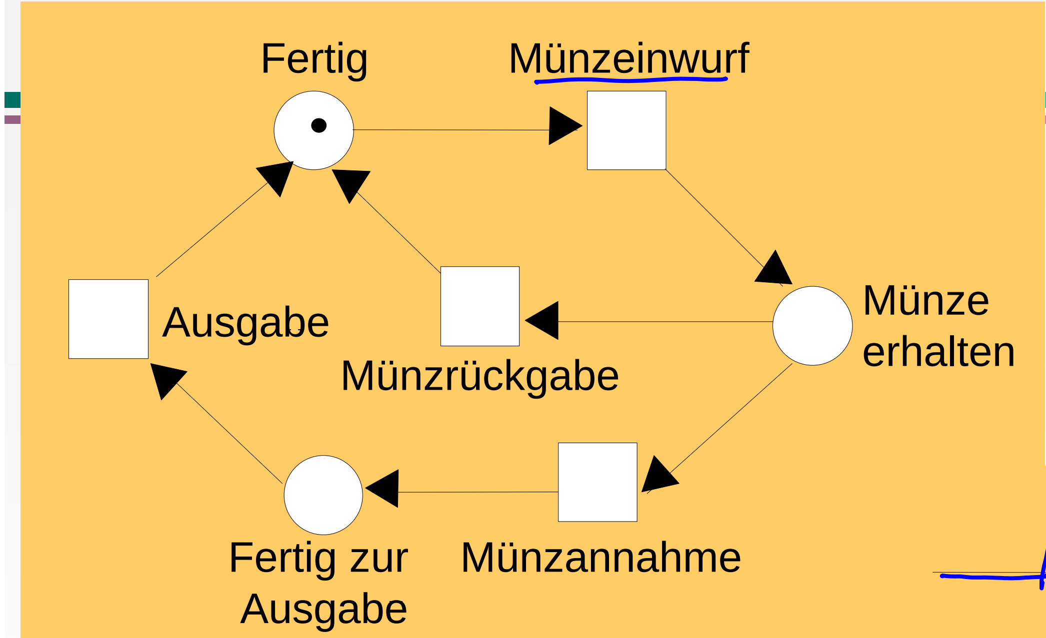
4.3 Markierungsgraph

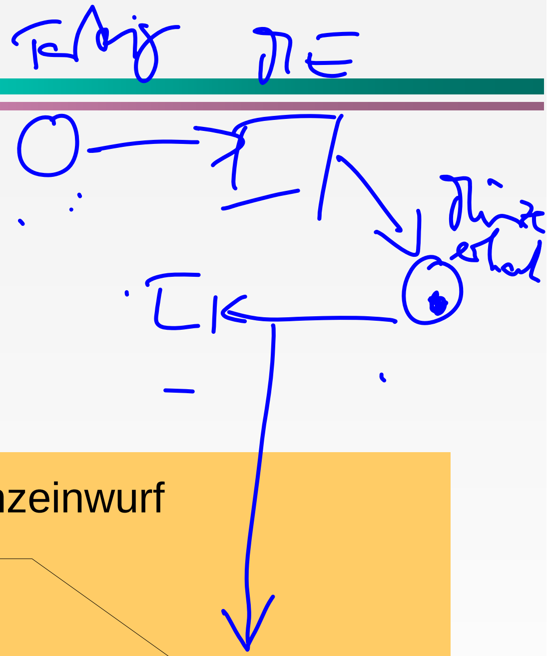
- Sei $N=(S, T, F)$ ein S/T-Netz und m_0 eine Markierung von N, genannt ***Anfangsmarkierung***.

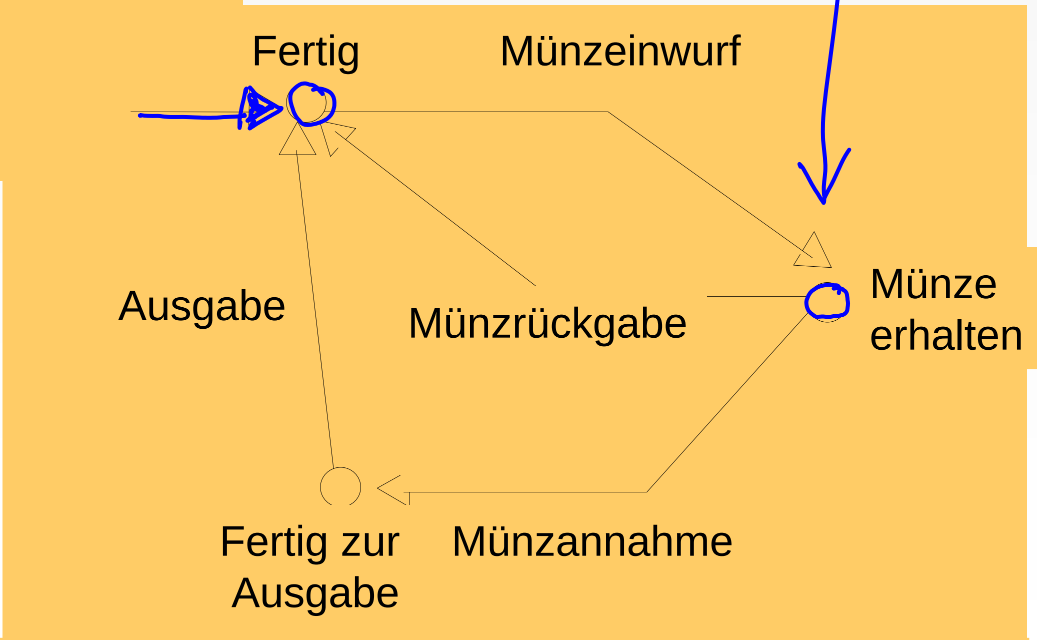
- Der Graph aus Knoten $[m_0>$ und mit Transitionen beschrifteten Kanten $\{(m, t, m') | m \xrightarrow{t} m'\}$ heißt ***Markierungsgraph*** von (N,m_0).

  Die Anfangsmarkierung wird im Markierungsgraph besonders dargestellt : $\rightarrow\bullet$.

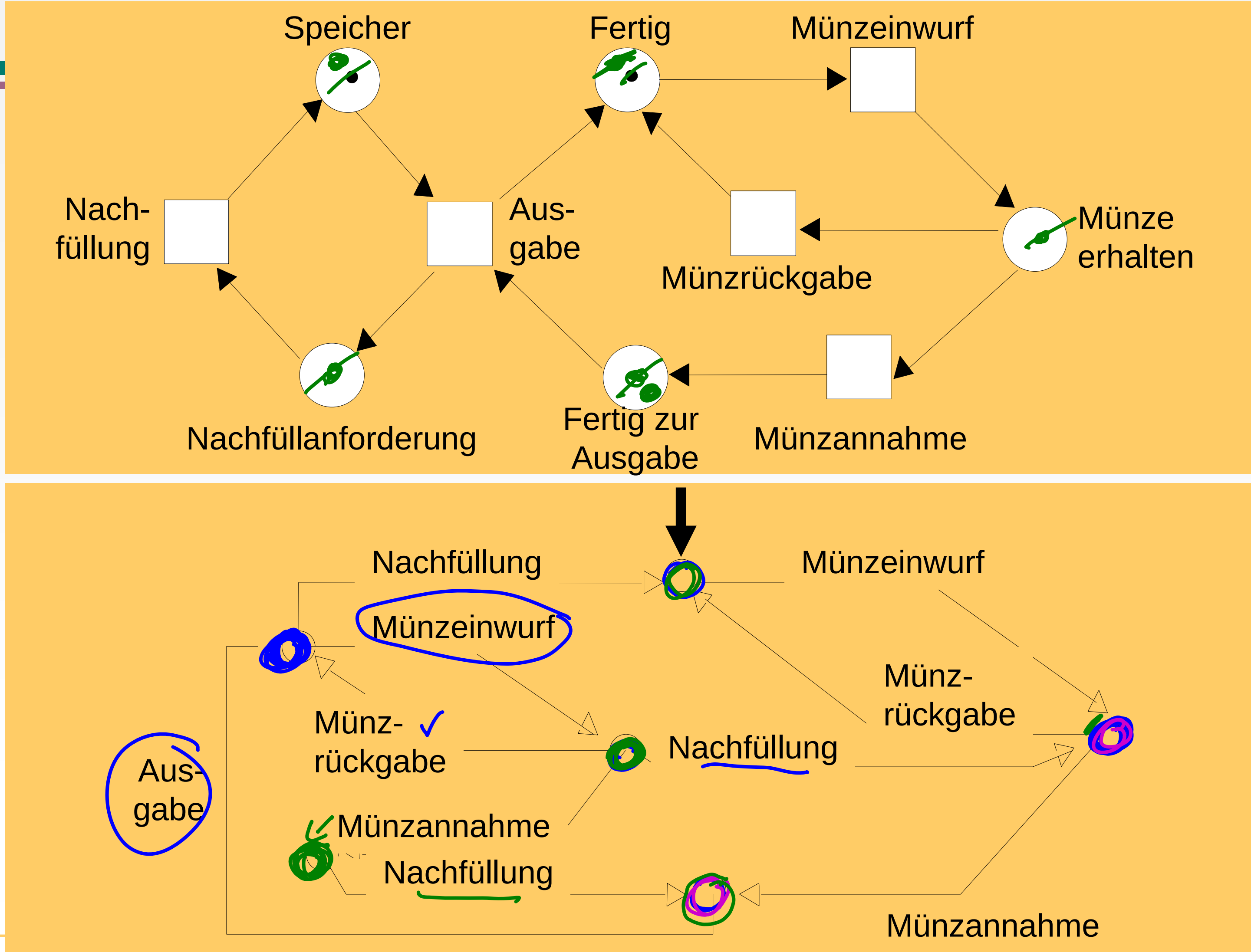
4.3 Markierungsgraph: Beispiel (I)



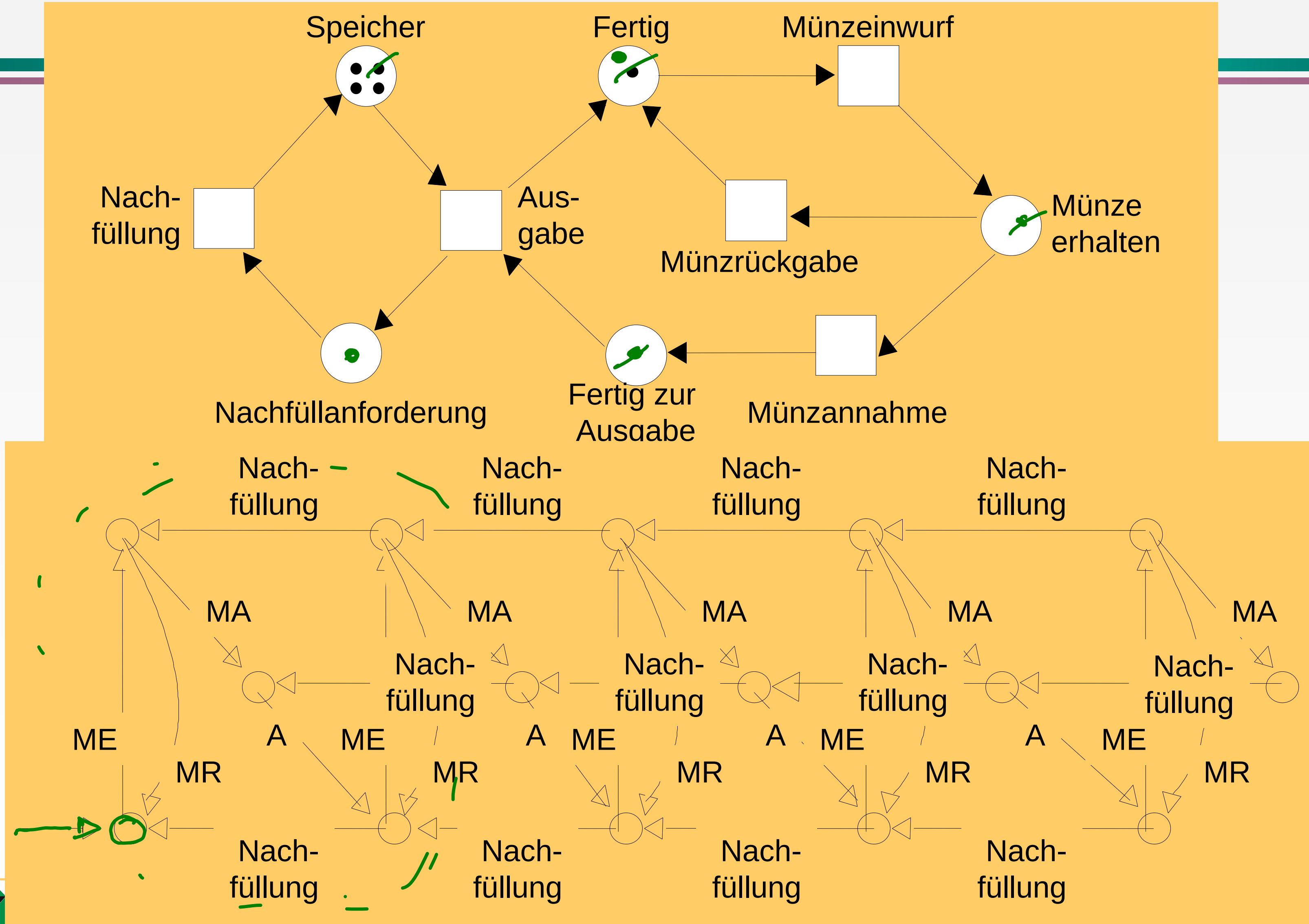




4.3 Markierungsgraph: Beispiel (II)

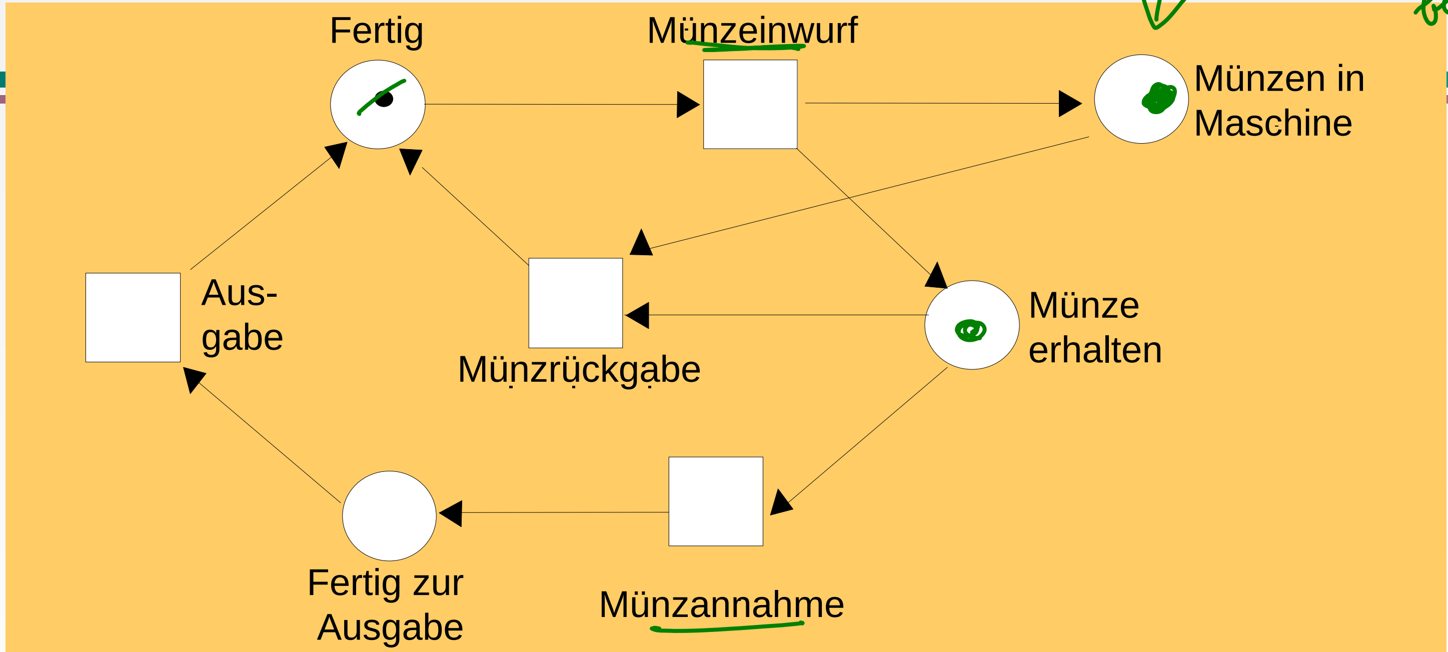


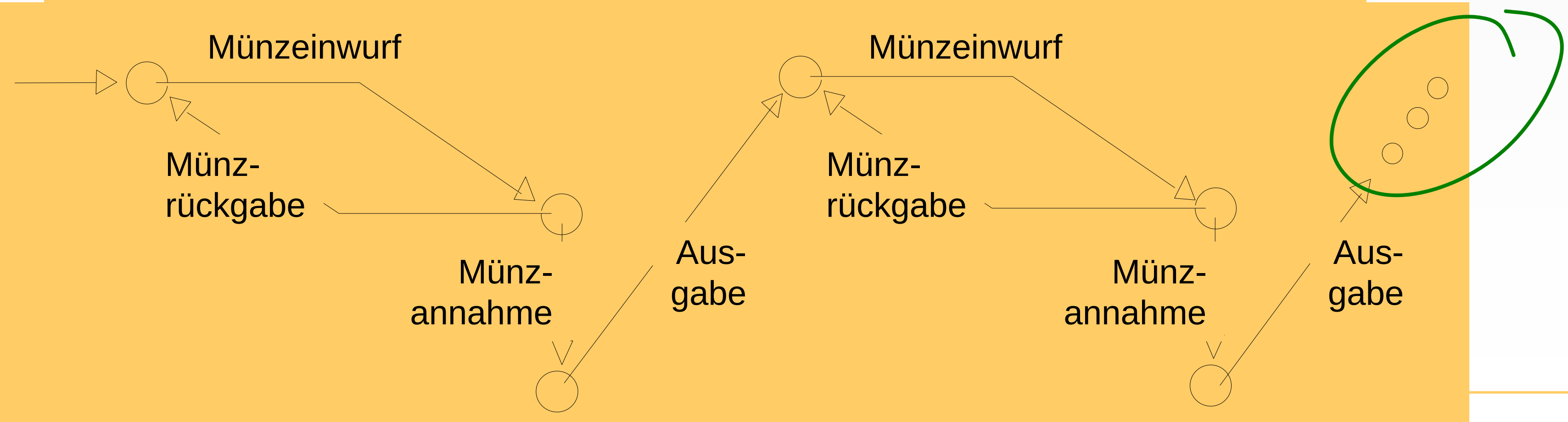
4.3 Markierungsgraph: Beispiel (III)



4.3 Markierungsgraph: Beispiel (IV)

S ist nicht beschränkt





4.4 Beschränktheit

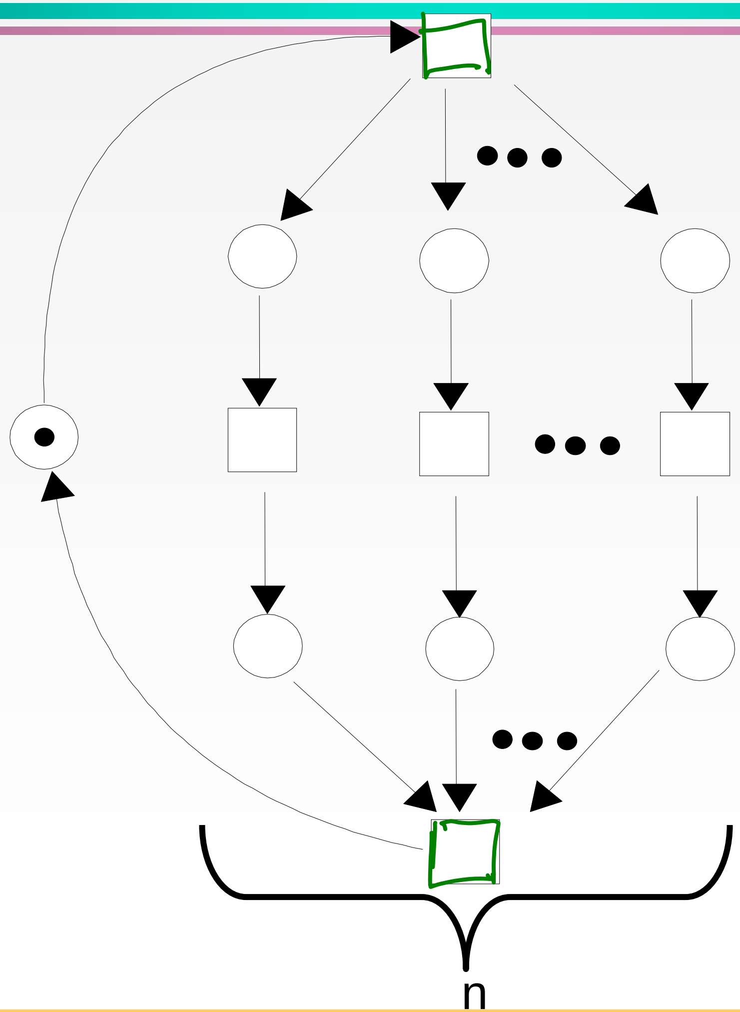
- (N,m_0) heißt ***beschränkt***, wenn eine Schranke b existiert, so dass $m(s) \leq b$ für alle $m \in [m_0>$.

- (N,m_0) heißt ***1-beschränkt***, wenn b=1.

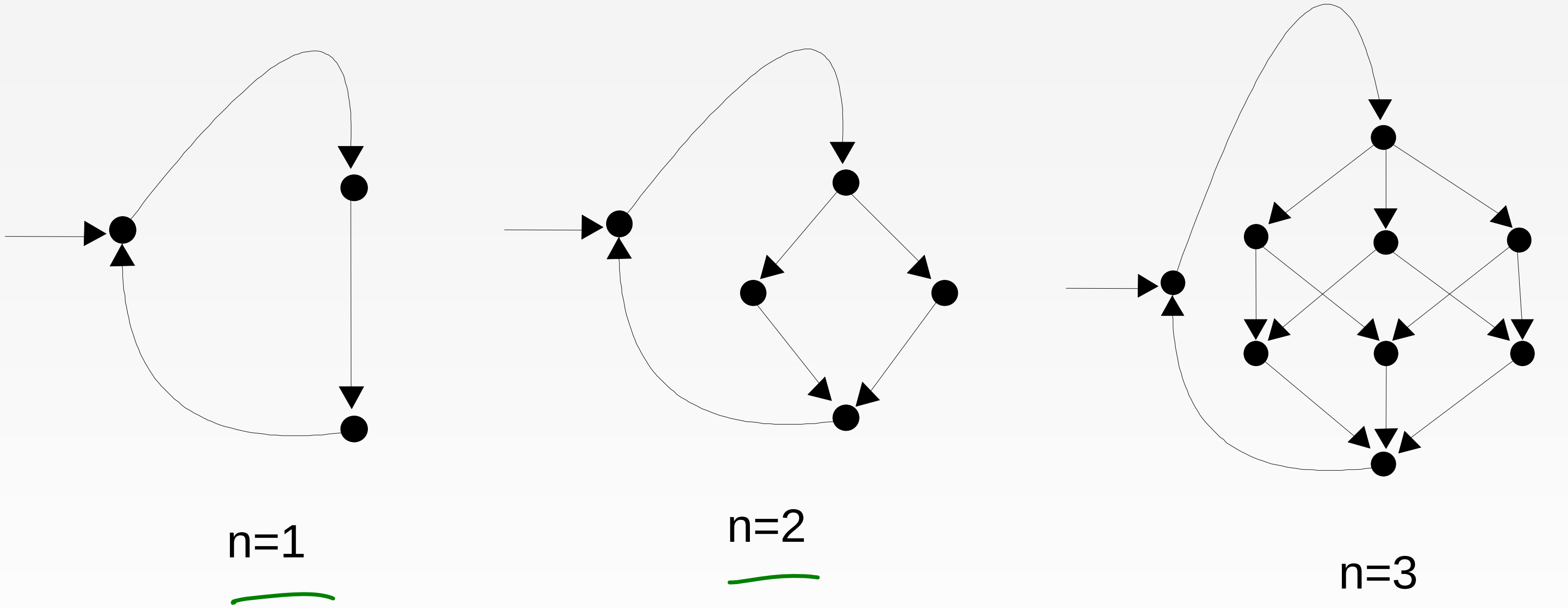
- Für beschränkte S/T-Netze wächst die Zahl erreichbarer Markierungen exponentiell in der Größe des Netzes (Anzahl der $S \cup T$) (***State Space Explosion***).

  Eine Analyse des Verhaltens mittels Konstruktion des Markierungsgraphen ist deshalb praktisch unmöglich.

4.5 State Space Explosion: Beispiel



4.5 State Space Explosion: Beispiel



Für n=k hat der Markierungsgraph 2^k+1 Knoten, entsprechend viele Markierungen sind erreichbar.

4.6 Komplementbildung

Vgl. Beispiel 4.3 (IV):
Stelle „Münzen in Maschine“ nicht beschränkt
➔ Markierungsgraph nicht endlich.
Oft sinnvoll: Beschränkung der Kapazität einer Stelle s durch Schranke k(s).
Hilfsmittel: „Komplementstelle“

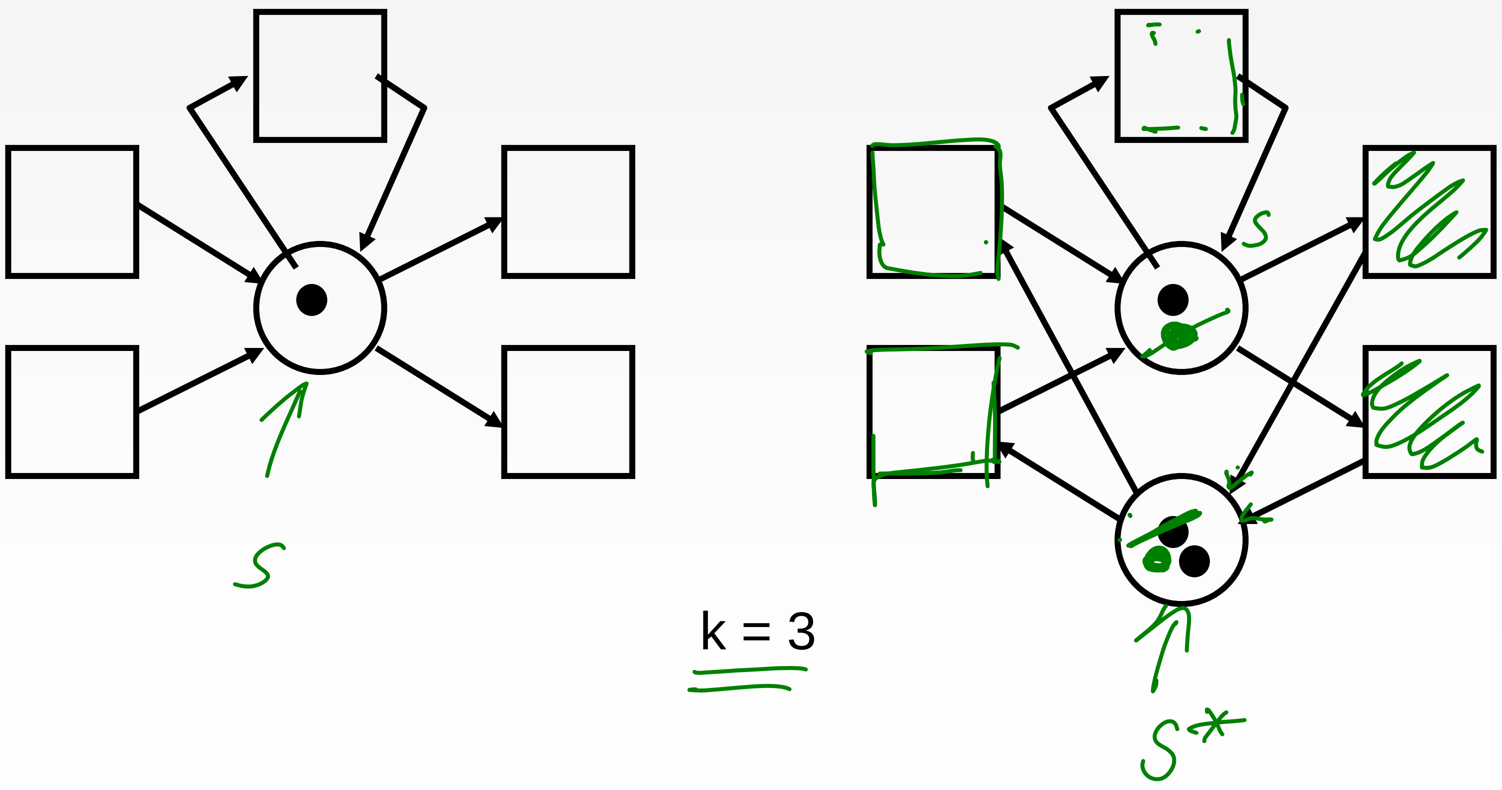
Sei s eine Stelle; eine neue Stelle s* mit

– $s^* \bullet \quad = \quad \bullet s \setminus s \bullet$
– $\bullet s^* \quad = \quad s \bullet \setminus \bullet s$
– $m_0(s^*) \quad = \quad k(s) - m_0(s)$,

wird als das ***Komplement*** (die Komplementstelle) von s bezeichnet.

k(s) ist die gewünschte Schranke für s.

4.6 Komplementbildung: Beispiel



4.7 Zusammenhängend

- Ein Netz $N = (S,T,F)$ ist ***zusammenhängend***, wenn keine Zerlegung $S \cup T = X_1 \cup X_2$, $X_1, X_2 \neq \emptyset$, $X_1 \cap X_2 = \emptyset$ mit $F \subseteq (X_1 \times X_1) \cup (X_2 \times X_2)$ existiert.

$X_1 \cap S$ $X_1 \cap T$

$N'(X_1, X_1, \ldots)$

Stell. Tras.

- N heißt ***stark zusammenhängend***,

  wenn für je zwei Elemente $x, y \in S \cup T$ mit $x \neq y$ eine Sequenz $(z_1,z_2), (z_2,z_3), \ldots, (z_{n-1},z_n) \in F$ existiert $(n \geq 2)$, so dass $x = z_1$ und $y = z_n$.

4.8 Tot, lebendig, verklemmungsfrei

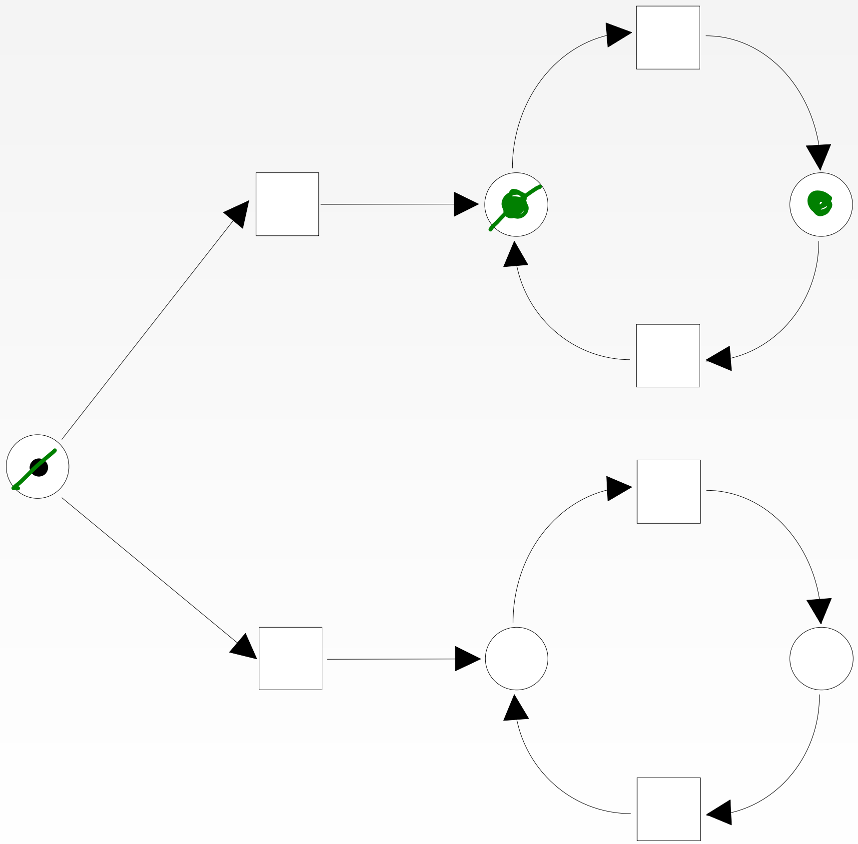
- Eine Transition t heißt ***tot*** unter einer Markierung m, wenn kein m’ $\in$ [m> die Transition t aktiviert.
- Eine Markierung heißt ***tot***, wenn alle Transitionen tot sind.
- Eine Markierung heißt ***verklemmungsfrei***, wenn keine tote Markierung erreichbar ist.
- Eine Transition heißt ***lebendig*** unter einer Markierung m,wenn sie unter keiner Folgemarkierung m’ tot ist.
- Eine Markierung heißt ***lebendig***, wenn alle Transitionen lebendig sind.

Noch mehr

- Ein markiertes Petrinetz heißt ***reversibel***, falls m_0 von jeder erreichbaren Markierung aus erreichbar ist.

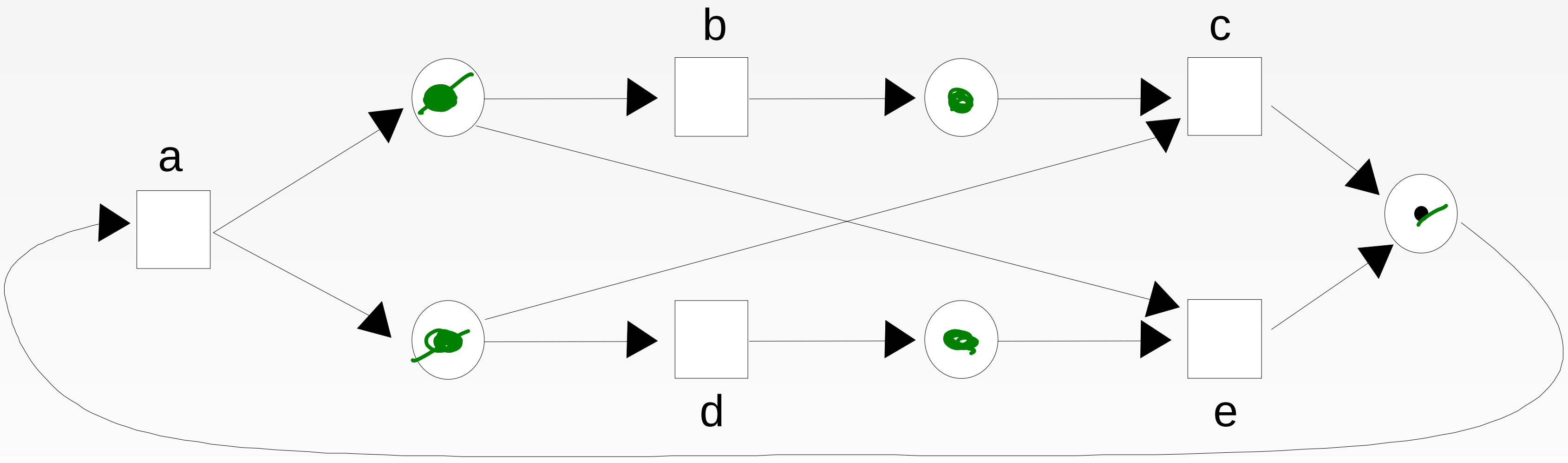
- Ein markiertes Petrinetz ***terminiert***, wenn die Menge der Schaltfolgen endlich ist.

4.8 Beispiel



verklemmungsfrei, aber nicht lebendig

4.8 Beispiel



beliebig lange Schaltfolgen, aber nicht verklemmungsfrei

4.9 Eigenschaften in Sätzen

- Ein markiertes Petrinetz ist genau dann verklemmungsfrei,
  wenn der dazugehörige Markierungsgraph keinen Knoten ohne Nachfolger besitzt.

- Ein markiertes Petrinetz terminiert genau dann,
  wenn der dazugehörige Markierungsgraph zyklenfrei ist.

- Es gibt kein markiertes Petrinetz, das terminiert und verklemmungsfrei ist.

4.9 Eigenschaften (II)

- Jedes lebendige markierte Petrinetz mit mindestens einer Transition ist verklemmungsfrei.

- Ein markiertes Petrinetz ist genau dann lebendig, wenn unter keiner erreichbaren Markierung eine tote Transition existiert.

- Ein markiertes Petrinetz ist genau dann beschränkt, wenn die Menge der erreichbaren Markierungen endlich ist.

4.9 Eigenschaften (III)

- Falls (N,m_0) beschränkt ist mit Schranke b, dann sind höchstens $(b+1)^{|s|}$ Markierungen erreichbar.

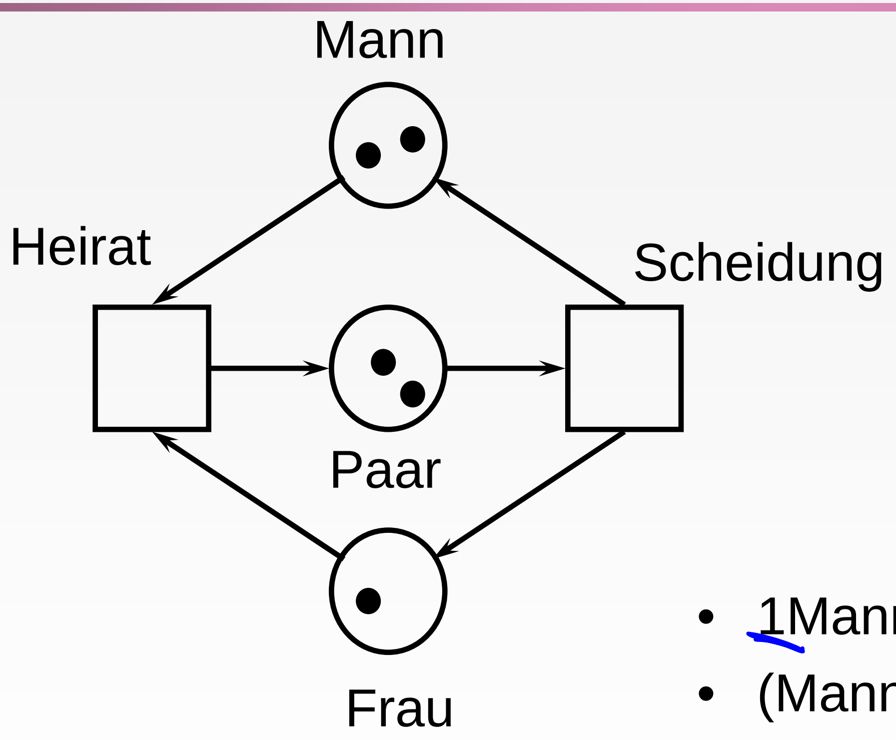
- Ein markiertes Petrinetz ist genau dann reversibel,
  wenn der dazugehörige Markierungsgraph stark zusammenhängend ist.

4.10 Stelleninvarianten

- Ein Stelleninvariante weist jeder Stelle eine ***Gewichtung*** zu, derart, dass die ***gewichtete Summe*** der Marken für alle Markierungsübergänge konstant bleibt.

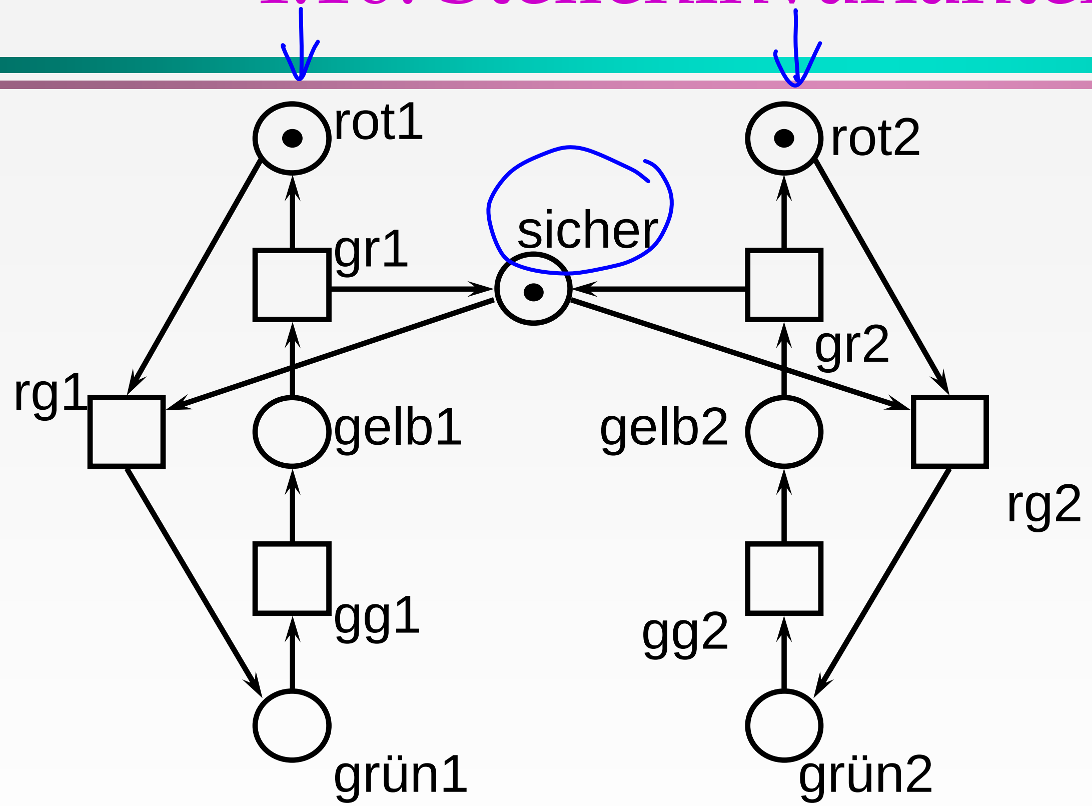
- Für jede erreichbare Markierung entspricht die gewichtete Summe der Marken der gewichteten Markensumme der Anfangsmarkierung (eine Konstante).

4.10. Stelleninvarianten: Beispiel



- 1Mann + 1Frau + 2Paar = const.
- (Mann + Frau + 2Paar = 7)
- Frau + Paar
- Mann + Paar
- Mann - Frau

4.10. Stelleninvarianten: Beispiel



rot1 + gelb1 + grün1 = 1

rot2 + gelb2 + grün2 = 1

sicher + grün1 + grün2 + gelb1 + gelb2 = 1

rot1 + rot2 - sicher = 1

4.11. Free Choice

- Ein ***Free-Choice-Netz*** ist ein Netz bei dem gilt:

  $\forall (s,t) \in F \cap (S \times T): \quad \bullet t \times s \bullet \subseteq F$

  d.h. die Transitionen einer vorwärts verzweigten Stelle dürfen nicht rückwärts verzweigt sein

- Daraus folgt:

  $s \rightarrow \boxed{t}$

  » $\forall t_1, t_2 \in T: \quad \bullet t_1 \cap \bullet t_2 \neq \emptyset \quad \Rightarrow \quad \bullet t_1 = \bullet t_2$

  » $\forall s_1, s_2 \in S: \quad s_1\bullet \cap s_2\bullet \neq \emptyset \quad \Rightarrow \quad s_1\bullet = s_2\bullet$

  d.h., zwischen den Transitionen kann frei und unabhängig von anderen als den beteiligten Stellen ausgewählt werden

Alternative Definition: Free Choice Netz

Ein ***Free-Choice-Netz*** ist ein Netz bei dem gilt:

für je zwei Transitionen t1 und t2 mit $\bullet t1 \cap \bullet t2 \neq \emptyset$

gilt $\bullet t_1 = \bullet t_2$

Daraus folgt:

- zwischen den Transitionen kann frei und unabhängig von anderen Stellen ausgewählt werden.
- Ein Konflikt zwischen mehreren Transitionen soll also nur dadurch verschwinden können, dass er zu Gunsten einer der beteiligten Transitionen gelöst wird (und nicht durch Schalten einer unbeteiligten weiteren Transition).

Soundness-Eigenschaft und Zusammenhang mit Finite-State-Netzen

Soundness-Eigenschaft - informell

Von einem Prozess sollen minimal die folgenden Eigenschaften erfüllt werden:

Transitionen

- Ein Prozess enthält keine unnötigen Tasks (Aktivitäten) und jeder Fall (Case, Anfangsmarkierung) der an den Prozess zur Abarbeitung geschickt wird, muss vollständig fertig gestellt werden und es sollen keine Markierungen übrig bleiben.
- Definiert man den Prozess als Petrinetz, dann gilt auch dass es eine einzige Start- und eine einzige Endestelle besitzt.

Ein solcher Prozess wird als **intakt (sound)** oder auch einwandfrei bezeichnet.

Ein Petrinetz, dass einen intakten Prozess modelliert, wird als **Workflow-Netz (WF-Netz)** bezeichnet und zur Modellierung von Workflows genutzt.

Soundness-Eigenschaft - formal

Ein Workflow-Netz wird als intakt (sound) definiert wenn es die folgenden drei Anforderungen erfüllt:

1. Für jede Markierung, die in die Startstelle gelegt wird, erscheint eventuell eine (und höchstens eine) Markierung in der Endestelle.
2. Wenn die Markierung in der Endestelle erscheint, dann sind alle anderen Stellen des Netzes unmarkiert.
3. Für jede Transition (Task) ist es möglich vom initialen Zustand in einen Zustand zu kommen, in dem die Transition aktiviert ist, also schalten kann.

Soundness-Eigenschaft - Ergänzung

Zu den drei Eigenschaften für intakte WF-Netze kommt noch die Fairness-Annahme:

Wenn eine Task ausgeführt werden kann, dann wird sie nicht auf unbestimmte Zeit verschoben.

Z.B. bei iterativem Routing könnte man sich im Prinzip vorstellen, dass die Iteration unendlich oft durchgeführt wird, dann wäre die Soundness-Eigenschaft verletzt.

Genauso wird davon ausgegangen, dass zwei Tasks nicht eine dritte Task auf unbestimmte Zeit „aushungern" lassen.

Ohne diese Annahmen wäre ein Prozess mit selektivem oder iterativem Routing nicht intakt (sound).

Soundness-Eigenschaft - Analyse

Im Prinzip sind die drei geforderten Eigenschaften nachzuweisen.

- Strukturbetrachtung
- Markierungsgraph analysieren

Allerdings ist das sehr aufwändig und bietet keine guten Anhaltspunkte für „Reparatur“möglichkeiten.

Daher wird folgendes Vorgehen für die Analyse verwendet:

Das WF-Netz wird um eine Transition t* erweitert, wobei t* die Endestelle als Eingabe und die Startstelle als Ausgabe erhält. (Resultat: ein „short-circuited“ Netz.)

Hiermit wird die Soundness des WF-Netzes (also ohne t*) mit den Eigenschaften Lebendigkeit und Beschränktheit des Netzes mit t* nachgewiesen. Hierfür gibt es effiziente Algorithmen.

Für einige wichtige Unterkategorien, einschließlich der sog. Free-Choice-Petri-Netzen, kann Lebendigkeit und Beschränktheit mit polynomialem Zeitaufwand nachgewiesen werden.

(Routingkonstrukte geordnet (paarweise) wirken)

Exemplarische Fragen – Kapitel 4

- Welche Analysemöglichkeiten bieten Petri-Netze?
- Was ist ein Markierungsgraph und wozu wird er verwendet?
- Was bedeutet, dass ein Petri-Netz terminiert, …? (informell)
- Was ist die Soundness-Eigenschaft?
- Was bieten Free-Choice-Netze für Vorteile und wie werden sie in der Workflow-Modellierung eingesetzt?