Vorlesung „ Methoden des Software Engineering“

Block C „Formale Methoden“

Petri-Netze

Martin Wirsing

Einheit C.5, 14.12.2006

Ziele

- Petri-Netze zur operationellen Beschreibung reaktiver Systeme kennen lernen

Petrinetze

- **Petri-Netz** (auch Stellen-/Transitions-Netz):

  Formaler Kalkül zur **Modellierung von Abläufen mit nebenläufigen Prozessen** und **kausalen Beziehungen**

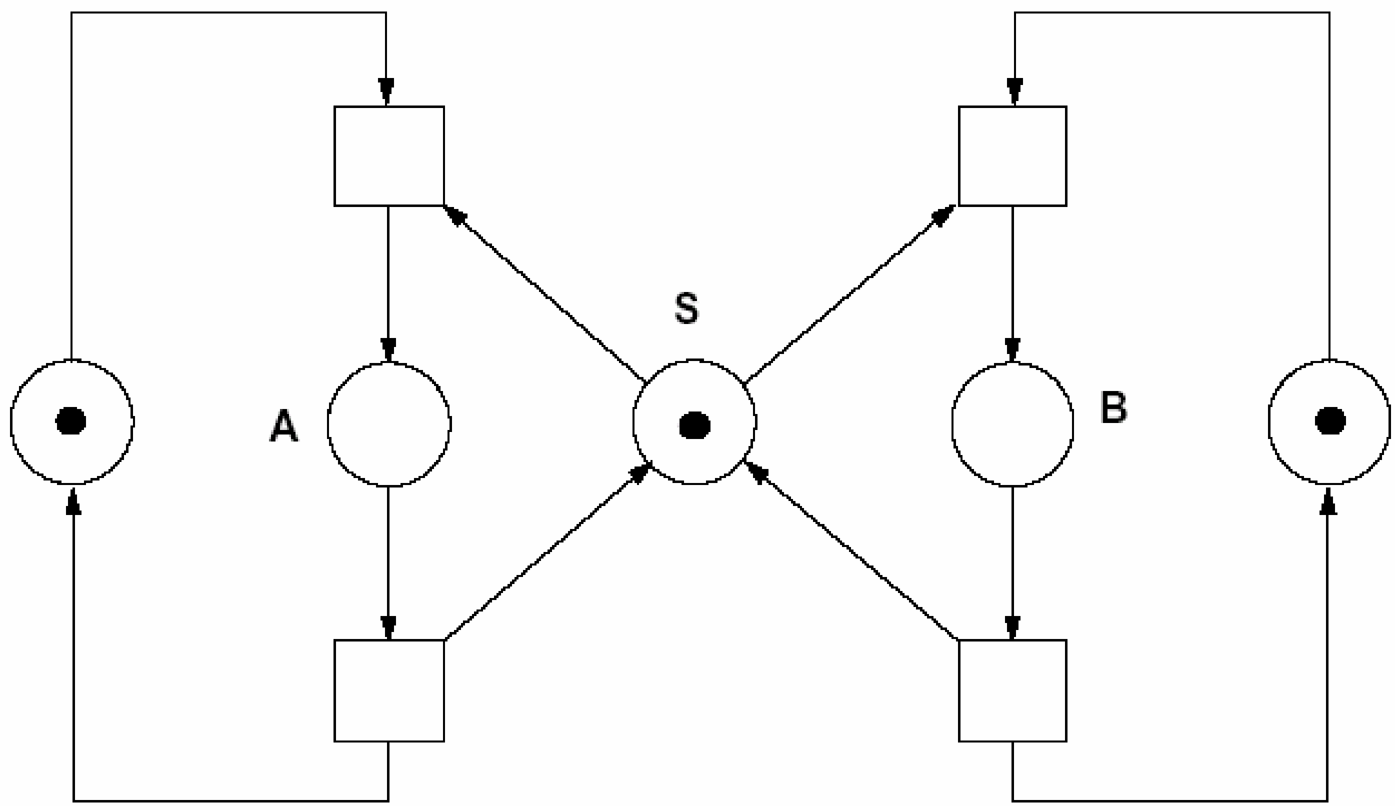
- Basiert auf **bipartiten gerichteten Graphen**:
  - **Knoten** repräsentieren **Bedingungen**, Zustände bzw**. Aktivitäten.**
  - **Kanten** verbinden **Aktivitäten** mit ihren **Vor- und Nachbedingungen.**
  - **Knotenmarkierung** repräsentiert den veränderlichen **Zustand des Systems.**
  - **graphische Notation.**

- **C. A. Petri** hat sie 1962 eingeführt.
- Es gibt zahlreiche Varianten und Verfeinerungen von Petri-Netzen. Hier nur die Grundform.

- **Anwendungen** von Petri-Netzen zur Modellierung von
  - realen oder abstrakten Automaten und Maschinen
  - kommunizierenden Prozessen in der Realität oder in Rechnern
  - Verhalten von Hardware-Komponenten/Geschäftsabläufen
  - **Semantik von UML 2.0 Aktivitätsdiagrammen**

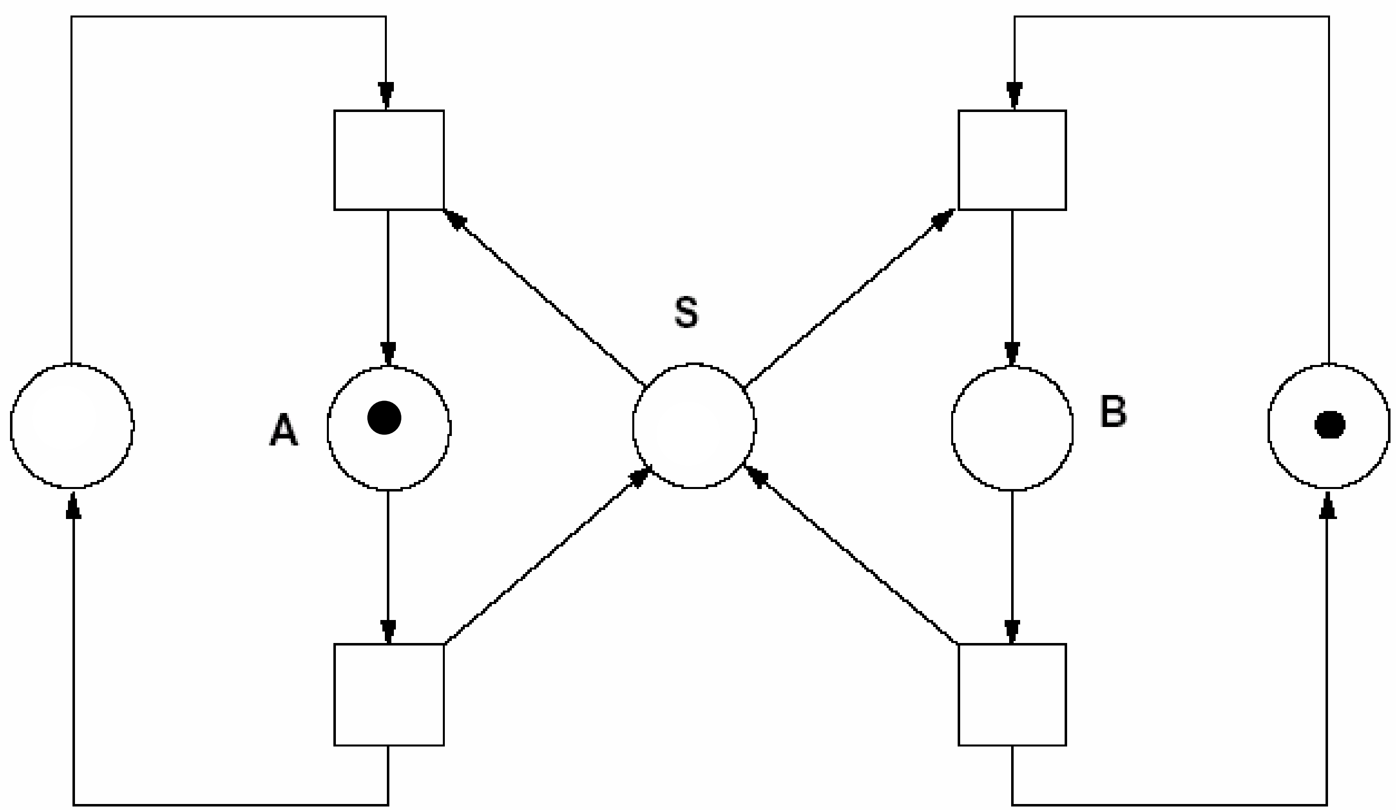
Petrinetz: Beispiel gegenseitiger Ausschluss

- Das Petri-Netz modelliert zwei **zyklisch ablaufende** Prozesse.
- Die mittlere Stelle synchronisiert die beiden Prozesse, so dass sie sich **nicht zugleich** in den Zuständen A und B befinden können.
- Prinzip: **gegenseitiger Ausschluss** durch **Semaphor**



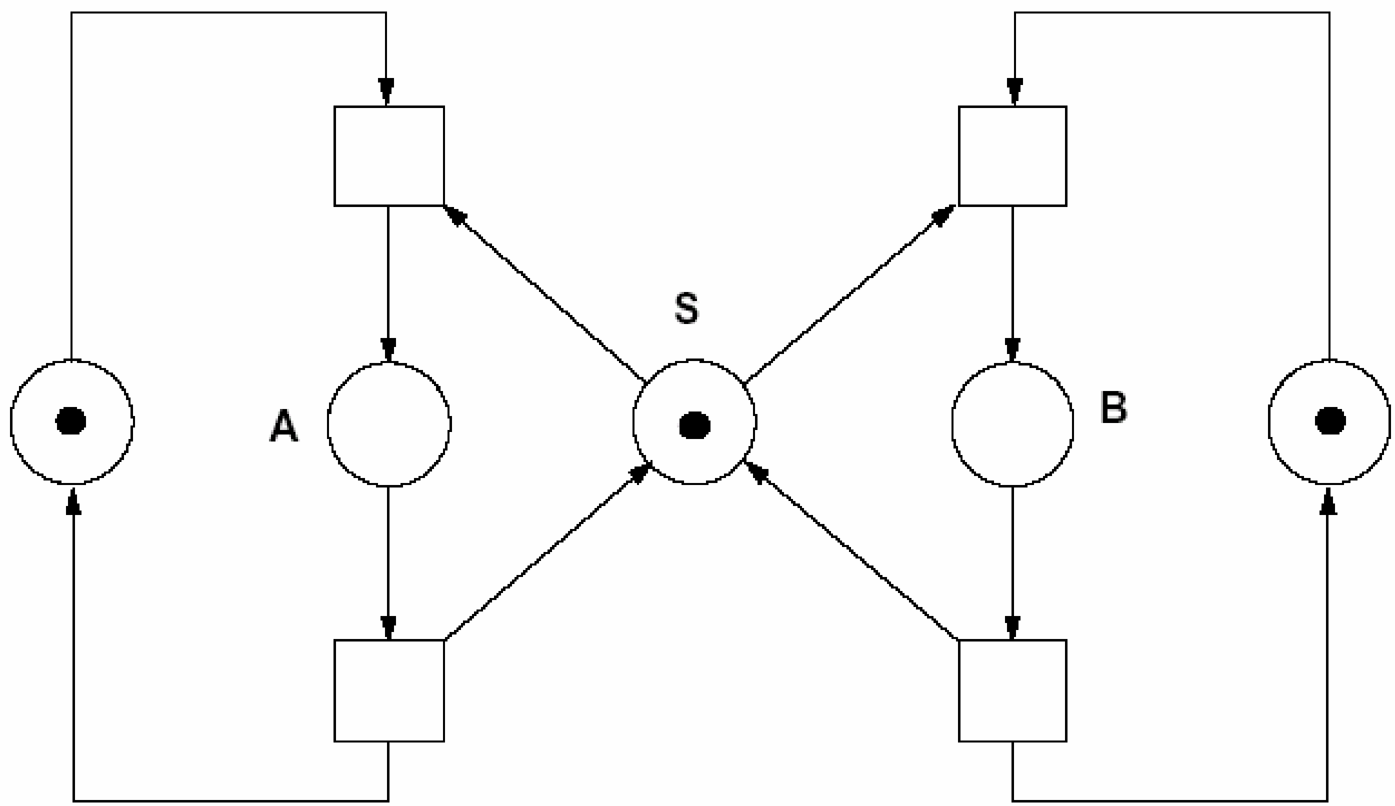
Petrinetz: Beispiel gegenseitiger Ausschluss

- Das Petri-Netz modelliert zwei **zyklisch ablaufende** Prozesse.
- Die mittlere Stelle synchronisiert die beiden Prozesse, so dass sie sich **nicht zugleich** in den Zuständen A und B befinden können.
- Prinzip: **gegenseitiger Ausschluss** durch **Semaphor**



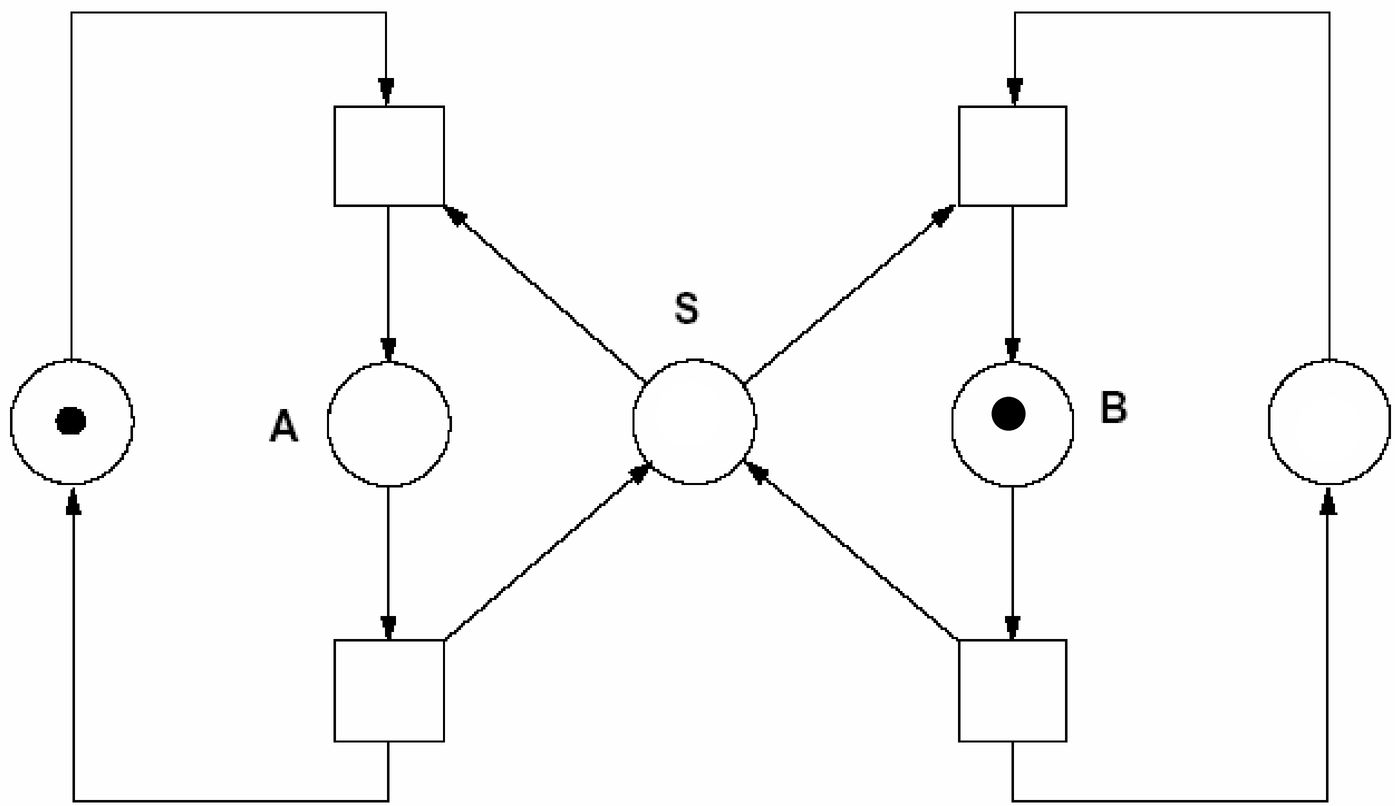
Petrinetz: Beispiel gegenseitiger Ausschluss

- Das Petri-Netz modelliert zwei **zyklisch ablaufende** Prozesse.
- Die mittlere Stelle synchronisiert die beiden Prozesse, so dass sie sich **nicht zugleich** in den Zuständen A und B befinden können.
- Prinzip: **gegenseitiger Ausschluss** durch **Semaphor**



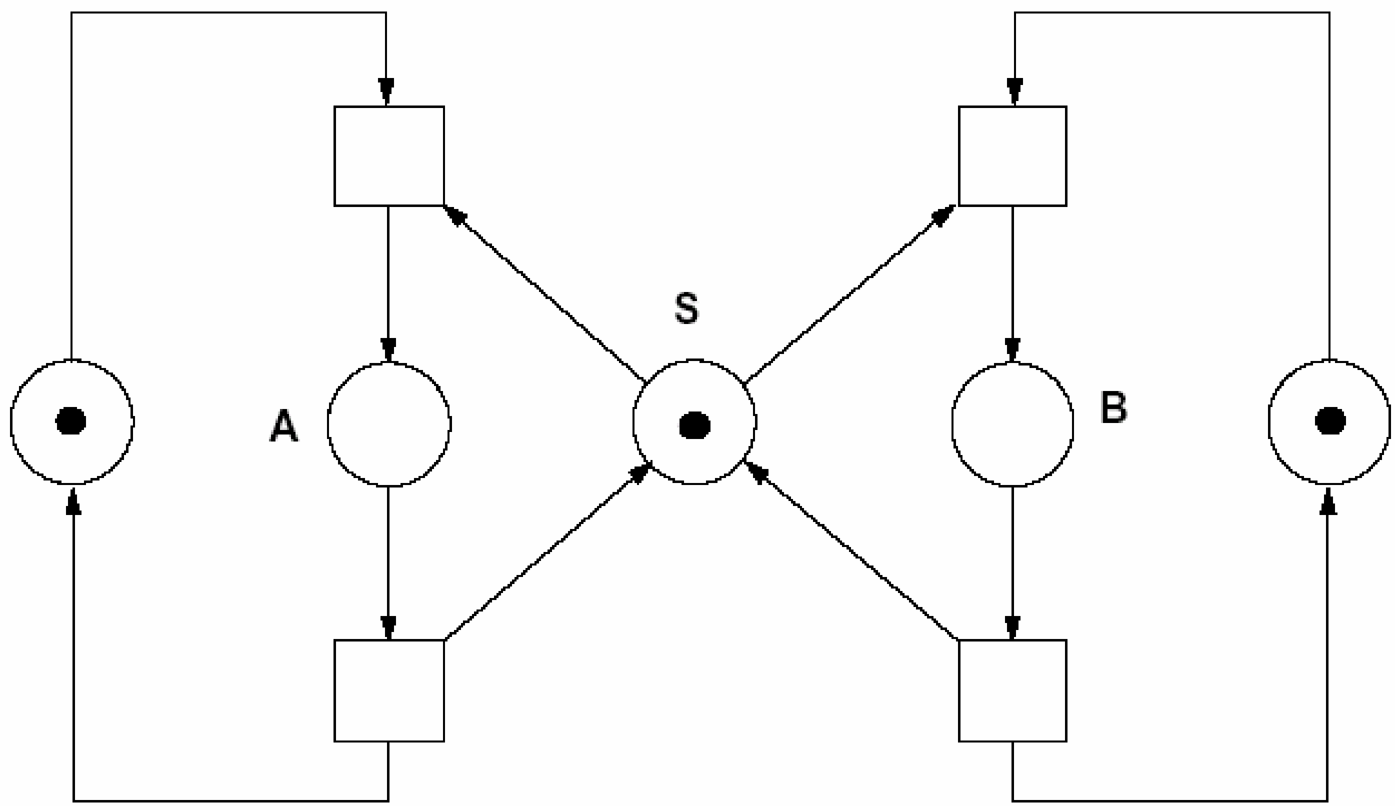
Petrinetz: Beispiel gegenseitiger Ausschluss

- Das Petri-Netz modelliert zwei **zyklisch ablaufende** Prozesse.
- Die mittlere Stelle synchronisiert die beiden Prozesse, so dass sie sich **nicht zugleich** in den Zuständen A und B befinden können.
- Prinzip: **gegenseitiger Ausschluss** durch **Semaphor**



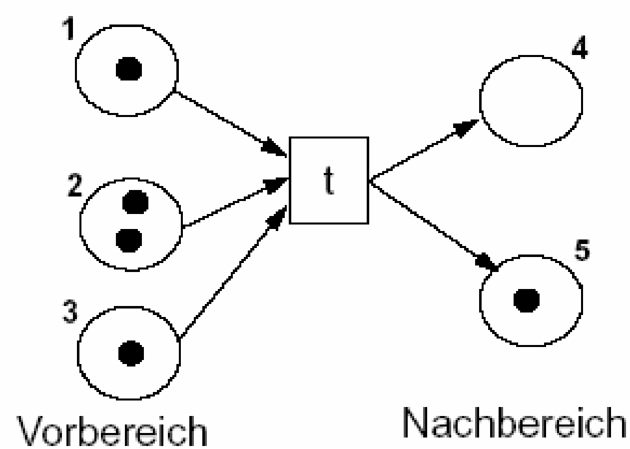
Petrinetz: Beispiel gegenseitiger Ausschluss

- Das Petri-Netz modelliert zwei **zyklisch ablaufende** Prozesse.
- Die mittlere Stelle synchronisiert die beiden Prozesse, so dass sie sich **nicht zugleich** in den Zuständen A und B befinden können.
- Prinzip: **gegenseitiger Ausschluss** durch **Semaphor**



Petrinetze

- Ein Petri-Netz ist ein Tripel P = (S, T, F) mit
  - S — Menge von Stellen, **repräsentieren Bedingungen, Zustände; graphisch Kreise**
  - T — Menge von Transitionen **oder Übergänge, repräsentieren Aktivitäten; graphisch Rechtecke**
  - F — Relation **mit $F \subseteq S \times T \cup T \times S$ repräsentieren kausale oder zeitliche Vor-, Nachbedingungen von Aktivitäten aus T**
- P bildet einen **bipartiten, gerichteten Graphen** mit den Knoten $S \cup T$ und den Kanten F.
- Zu einer **Transition** t in einem Petri-Netz P sind folgende Stellenmengen definiert
  Vorbereich (t) $= \{s \mid (s, t) \in F\}$
  Nachbereich (t) $= \{s \mid (t, s) \in F\}$



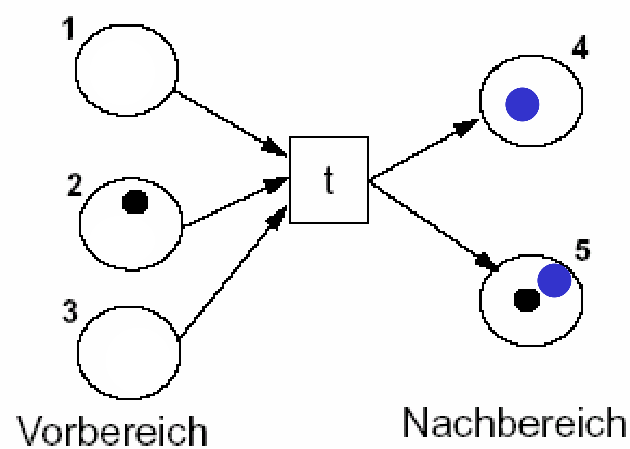
- Der **Zustand des Petri-Netzes** wird durch eine Markierungsfunktion angegeben, die jeder Stelle eine Anzahl von Marken zuordnet:

$$M_P: S \to \mathbb{N}_0$$

- Sind die Stellen von 1 bis n nummeriert, so kann man M_P als Folge angeben, z.B. (1,2,1,0,1)

Petrinetze

- Ein Petri-Netz ist ein Tripel $P = (S, T, F)$ mit
  - S — Menge von Stellen, **repräsentieren Bedingungen, Zustände; graphisch Kreise**
  - T — Menge von Transitionen **oder Übergänge, repräsentieren Aktivitäten; graphisch Rechtecke**
  - F — Relation **mit $F \subseteq S \times T \cup T \times S$ repräsentieren kausale oder zeitliche Vor-, Nachbedingungen von Aktivitäten aus T**
- P bildet einen **bipartiten, gerichteten Graphen** mit den Knoten $S \cup T$ und den Kanten F.
- Zu einer **Transition** t in einem Petri-Netz P sind folgende Stellenmengen definiert
  Vorbereich (t) $= \{s \mid (s, t) \in F\}$
  Nachbereich (t) $= \{s \mid (t, s) \in F\}$



- Der **Zustand des Petri-Netzes** wird durch eine Markierungsfunktion angegeben, die jeder Stelle eine Anzahl von Marken zuordnet:
  $$M_p: S \rightarrow \mathbf{N}_0$$
- Sind die Stellen von 1 bis n nummeriert, so kann man M_p als Folge angeben, z.B. (1,2,1,0,1)

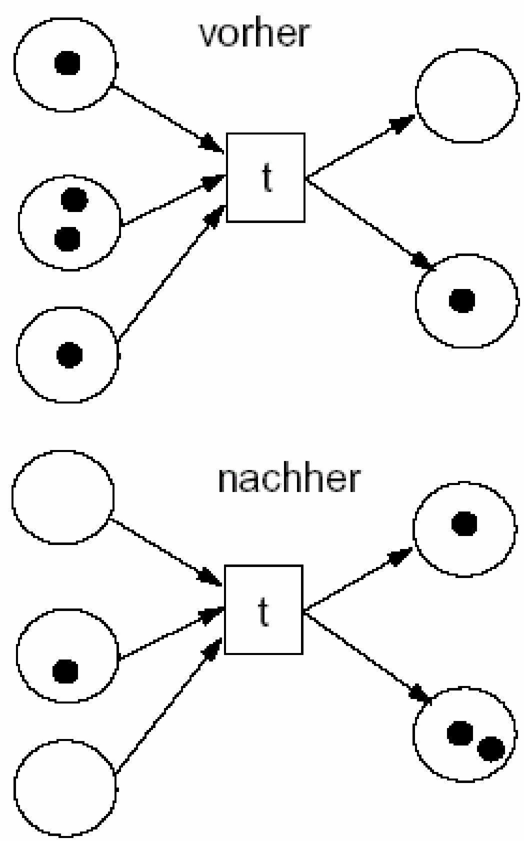
Schaltregel für Petrinetze

Das **Schalten einer Transition** t überführt eine Markierung M in eine Markierung M´.

Eine **Transition t kann schalten**, wenn für alle Stellen $s \in$ Vorbereich (t) gilt $M(s) \geq 1$.

Wenn eine Transition t **schaltet**, gilt für die **Nachfolgemarkierung M'**:

$M'(v) = M(v) - 1$	für alle $v \in$ Vorbereich(t) \ Nachbereich(t)
$M'(n) = M(n) + 1$	für alle $n \in$ Nachbereich(t) \ Vorbereich(t)
$M'(s) = M(s)$	sonst



Wenn in einem Schritt **mehrere Transitionen schalten können**, wird eine davon **nicht-deterministisch** ausgewählt.

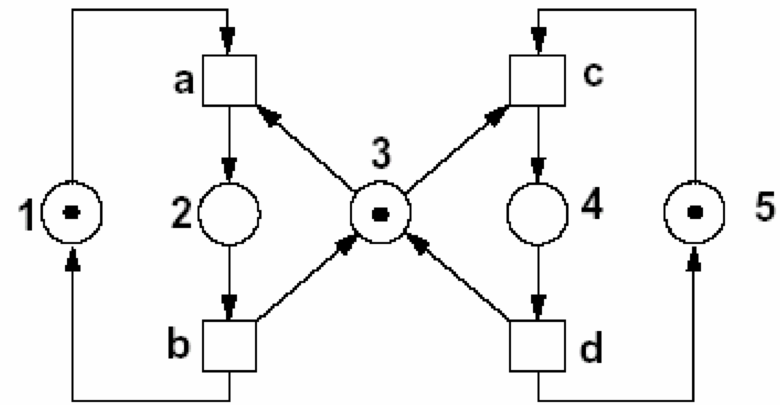
In jedem Schritt schaltet genau eine Transition –auch wenn das Petri-Netz parallele Abläufe modelliert!

Zwei Transitionen mit gemeinsamen Stellen im Vorbereich können (bei passender Markierung) im **Konflikt** stehen: Jede kann schalten, aber nicht beide nacheinander.

Markierungen

Zu jedem Petri-Netz wird eine Anfangsmarkierung M_0 angeben.

z.B. $M_0 = (1, 0, 1, 0, 1)$

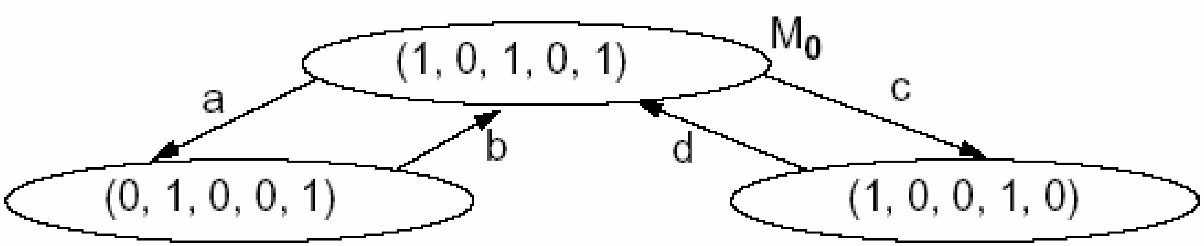


Wir sagen, eine Markierung M_2 ist von einer Markierung M_1 aus erreichbar, wenn es ausgehend von M_1 eine Folge von Transitionen gibt, die nacheinander schalten und M_1 in M_2 überführen können

Die Markierungen eines Petri-Netzes kann man als gerichteten Markierungsgraphen darstellen:

- **Knoten**: erreichbare Markierung
- **Kante M1->M2**: Die Markierung M1 kann durch Schalten einer Transition in M2 übergehen.

Markierungsgraph



Schaltfolgen

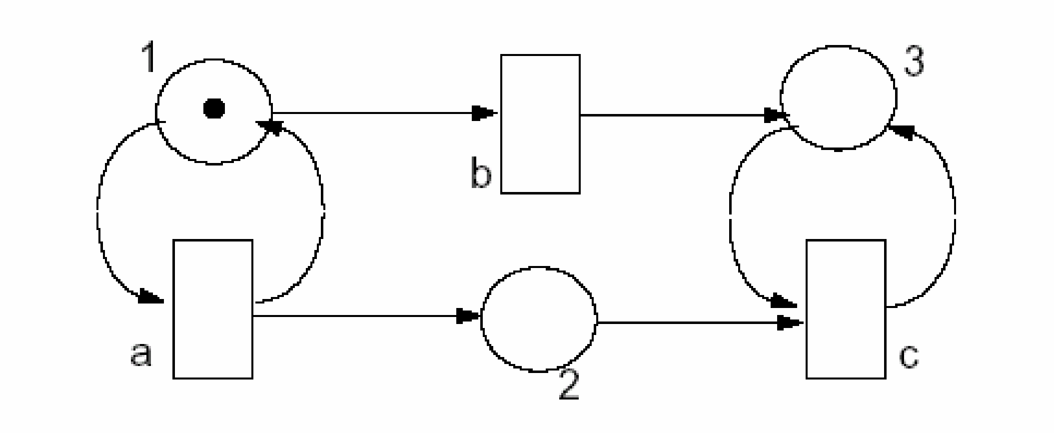
Schaltfolgen kann man angeben als

- Folge von Markierungen
- Folge der geschalteten Transitionen

Beispiel für eine **Schaltfolgen** zum Petri-Netz auf Folie 17:

(1, 0, 1, 0, 1)	a
(0, 1, 0, 0, 1)	b
(1, 0, 1, 0, 1)	c
(1, 0, 0, 1, 0)	d
(1, 0, 1, 0, 1)	

Schaltfolgen können als Wörter einer Sprache aufgefasst werden.



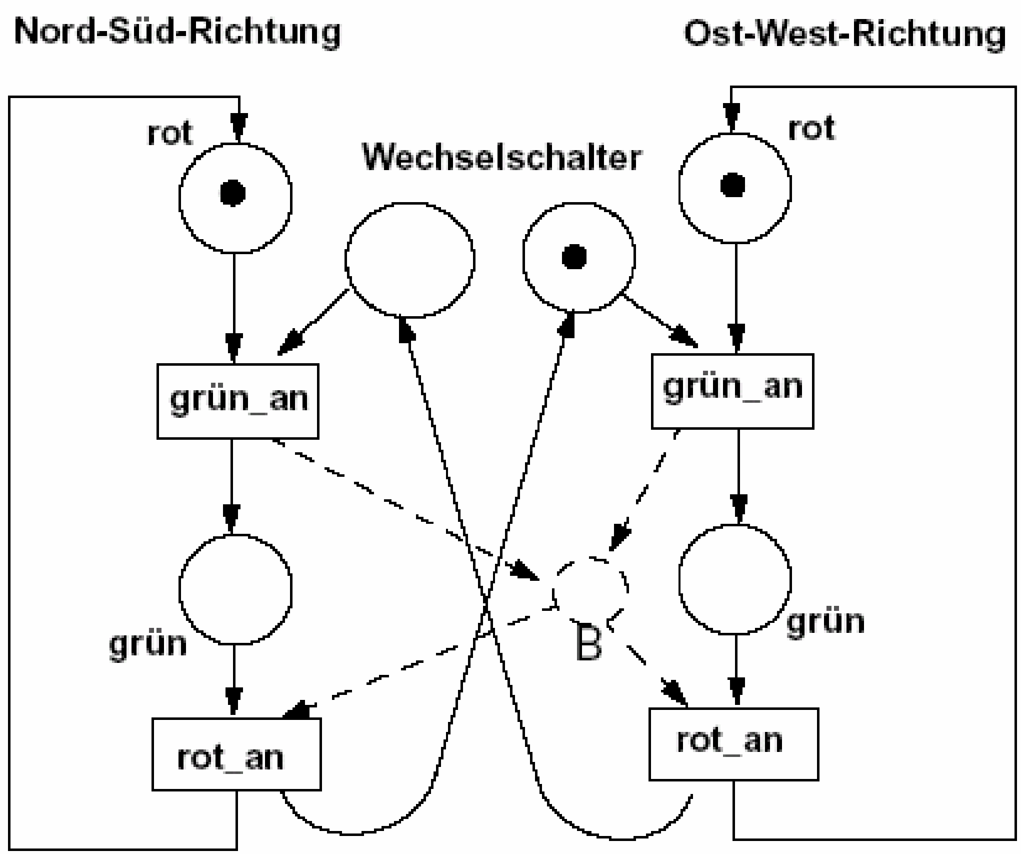
alle Schaltfolgen ohne Nachfolgemarkierung haben die Form

$a^n\ b\ c^n$

Petri-Netze können unbegrenzt zählen: Anzahl der Marken auf einer Stelle

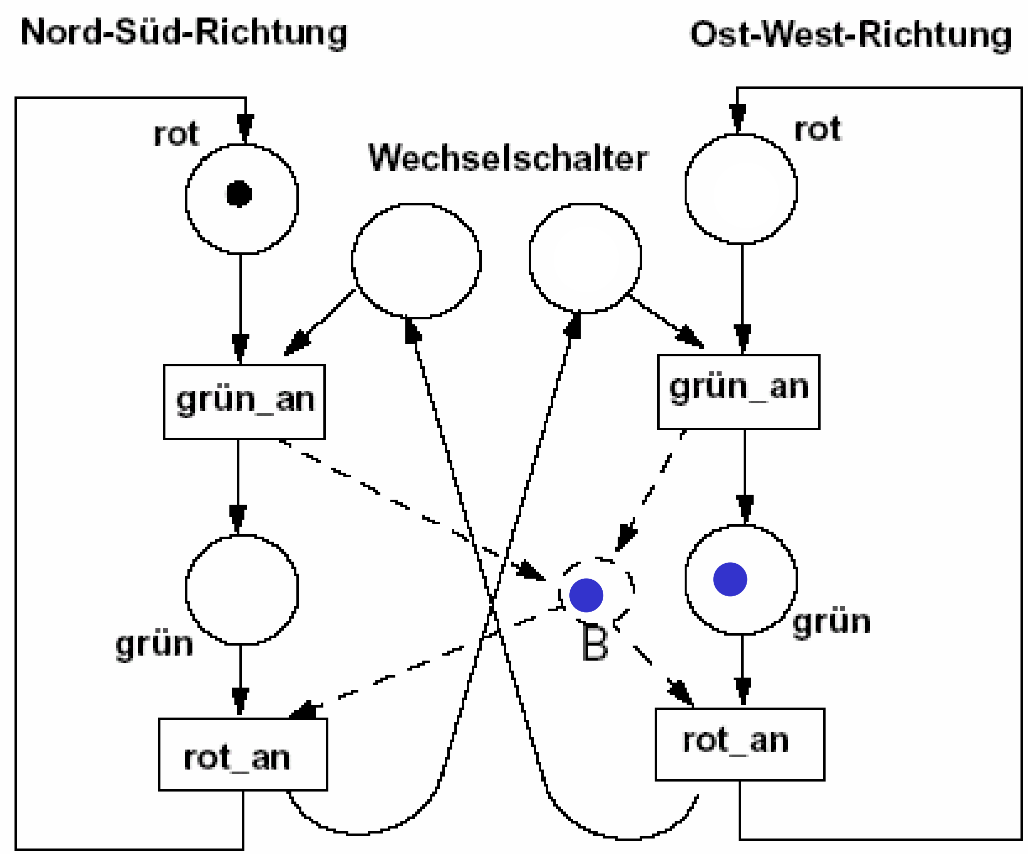
Schaltfolgen: Beispiel alternierende Ampel

- 2 sich zyklisch wiederholende Prozesse
- Die beiden Stellen „Wechselschalter“ koppeln die Prozesse, sodass sie alternierend fortschreiten
- Alle Stellen repräsentieren Bedingungen: 1 oder 0 Marken
- „Beobachtungsstelle“ B modelliert, wieviele Richtungen „grün“ haben



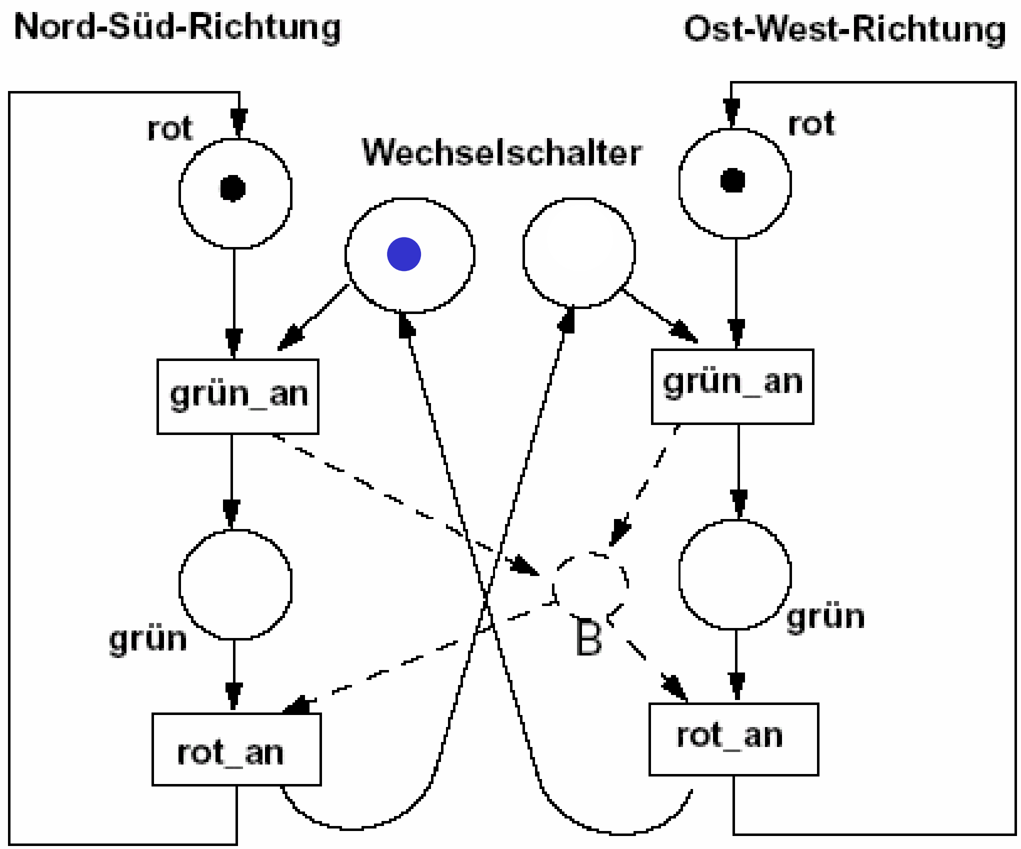
Schaltfolgen: Beispiel alternierende Ampel

- 2 sich zyklisch wiederholende Prozesse
- Die beiden Stellen „Wechselschalter“ koppeln die Prozesse, sodass sie alternierend fortschreiten
- Alle Stellen repräsentieren Bedingungen: 1 oder 0 Marke
- „Beobachtungsstelle“ B modelliert, wieviele Richtungen „grün“ haben



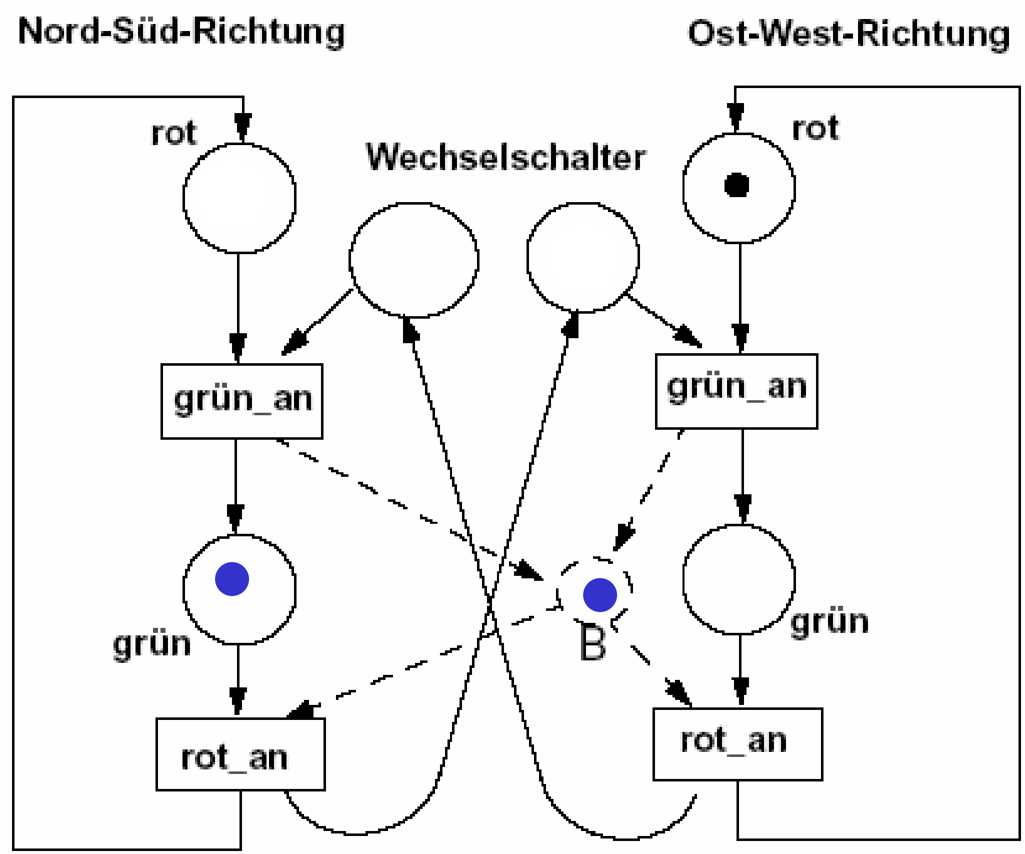
Schaltfolgen: Beispiel alternierende Ampel

- 2 sich zyklisch wiederholende Prozesse
- Die beiden Stellen „Wechselschalter“ koppeln die Prozesse, sodass sie alternierend fortschreiten
- Alle Stellen repräsentieren Bedingungen: 1 oder 0 Marken
- „Beobachtungsstelle“ B modelliert, wieviele Richtungen „grün“ haben



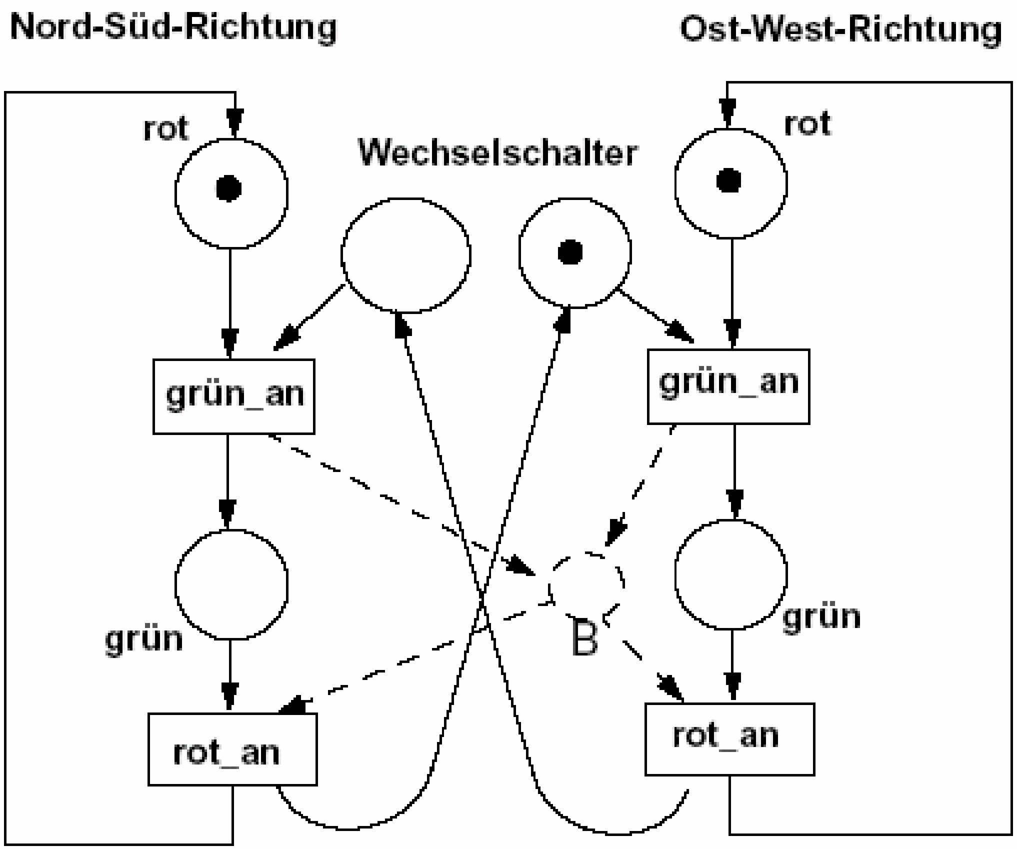
Schaltfolgen: Beispiel alternierende Ampel

- 2 sich zyklisch wiederholende Prozesse
- Die beiden Stellen „Wechselschalter“ koppeln die Prozesse, sodass sie alternierend fortschreiten
- Alle Stellen repräsentieren Bedingungen: 1 oder 0 Marken
- „Beobachtungsstelle“ B modelliert, wieviele Richtungen „grün“ haben



Schaltfolgen: Beispiel alternierende Ampel

- 2 sich zyklisch wiederholende Prozesse
- Die beiden Stellen „Wechselschalter“ koppeln die Prozesse, sodass sie alternierend fortschreiten
- Alle Stellen repräsentieren Bedingungen: 1 oder 0 Marken
- „Beobachtungsstelle“ B modelliert, wieviele Richtungen „grün“ haben

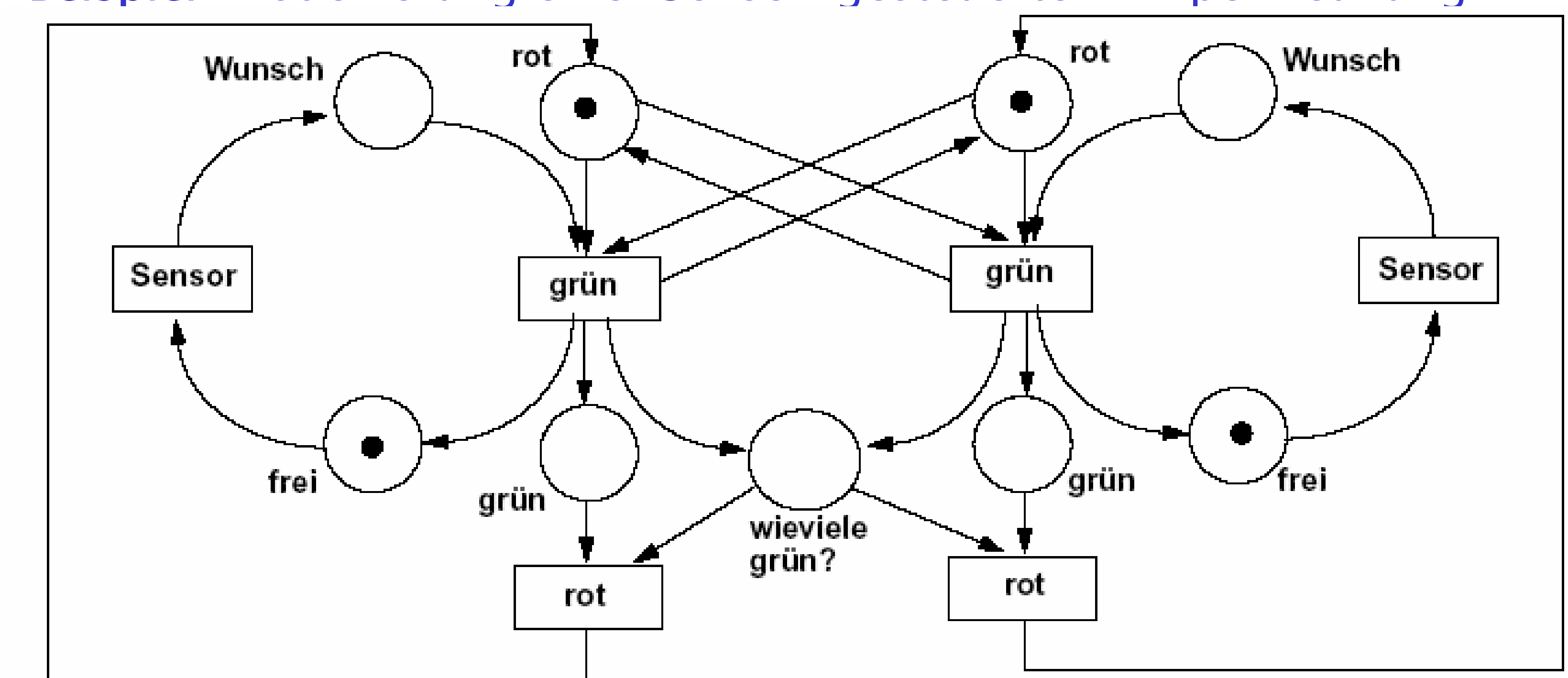


Sichere Netze

Ein Petri-Netz heißt **binär (sicher)**, wenn für alle aus M_0 erreichbaren Markierungen M und für alle Stellen s gilt $M(s) \leq 1$.

Petri-Netze, deren **Stellen Bedingungen repräsentieren,** müssen binär sein.

Beispiel: Modellierung einer Sensor-gesteuerten Ampelkreuzung:



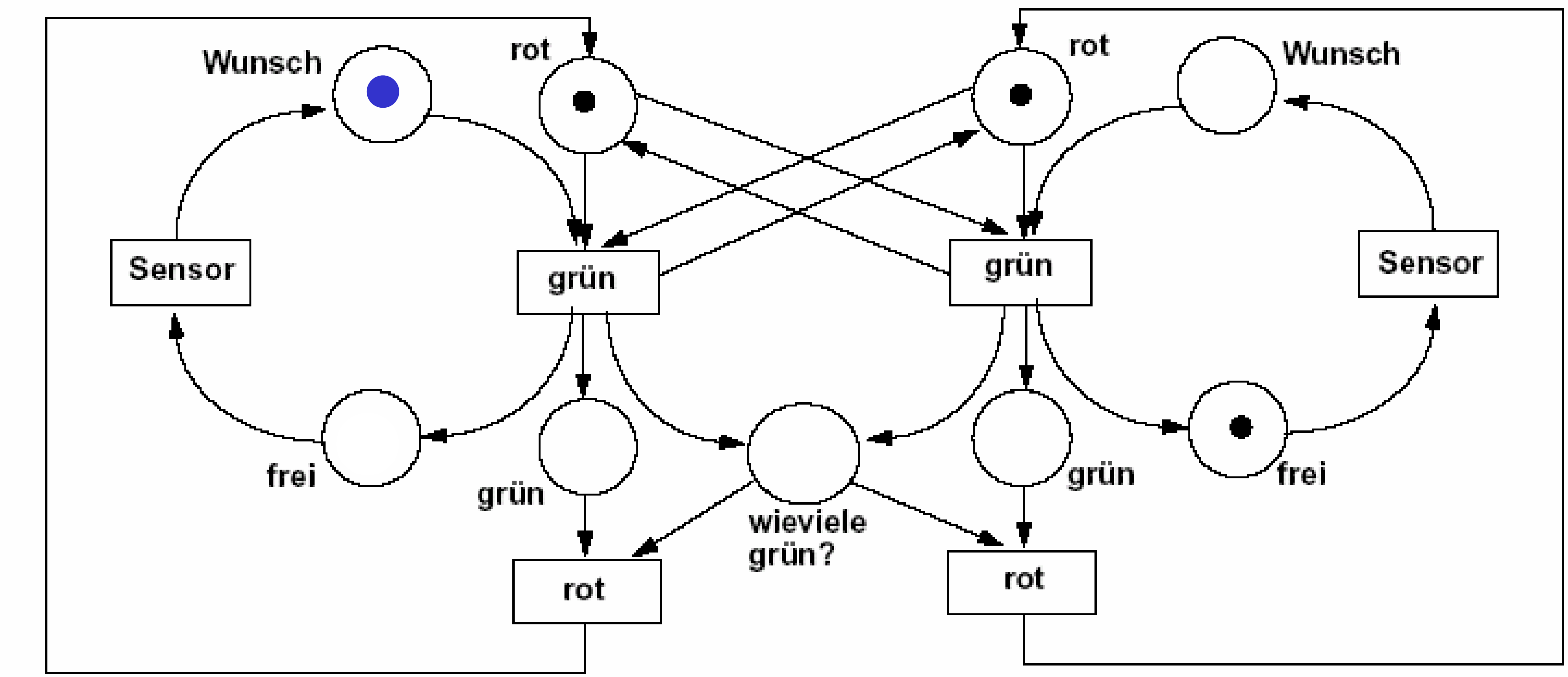
Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Sichere Netze

Ein Petri-Netz heißt **binär (sicher)**, wenn für alle aus M_0 erreichbaren Markierungen M und für alle Stellen s gilt $M(s) \leq 1$.

Petri-Netze, deren **Stellen Bedingungen repräsentieren,** müssen binär sein.

Beispiel: Modellierung einer Sensor-gesteuerten Ampelkreuzung:



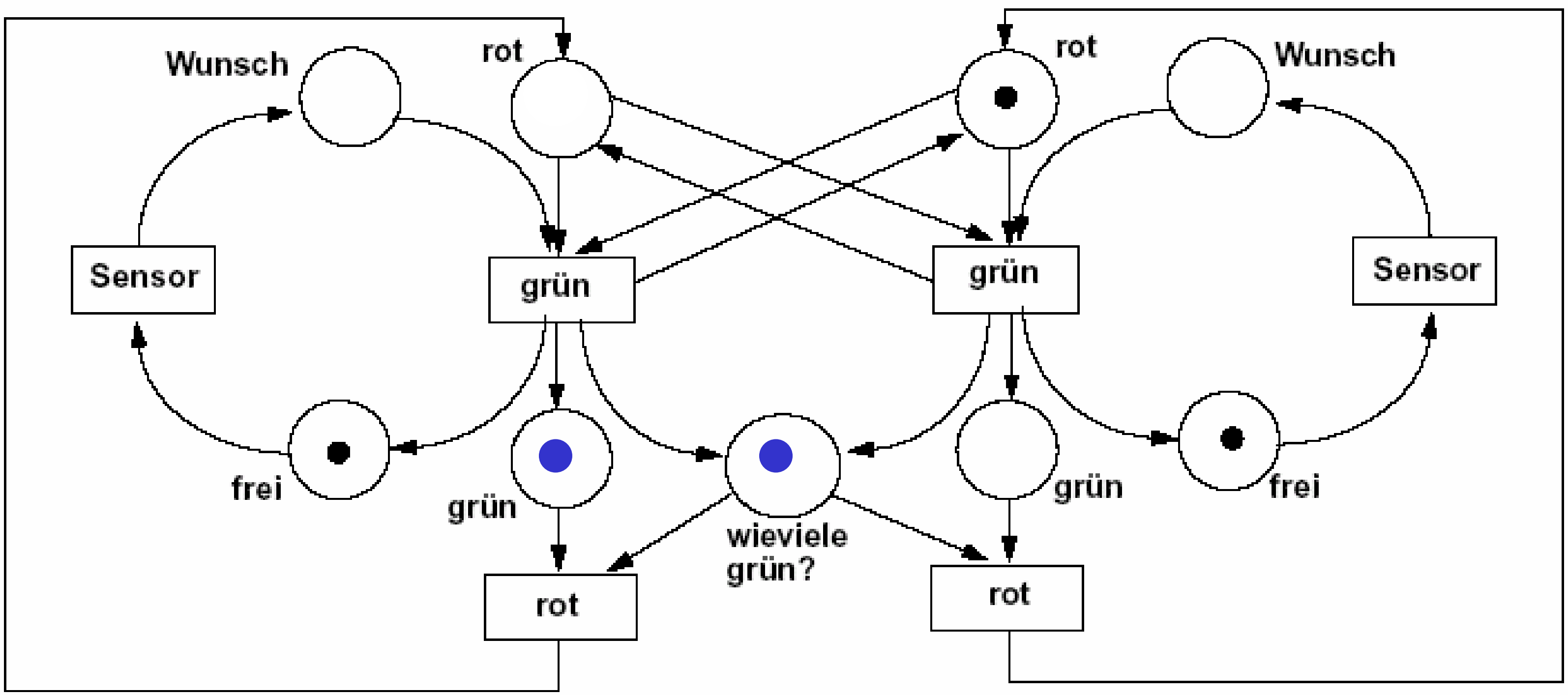
Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Sichere Netze

Ein Petri-Netz heißt **binär (sicher)**, wenn für alle aus M_0 erreichbaren Markierungen M und für alle Stellen s gilt $M(s) \leq 1$.

Petri-Netze, deren **Stellen Bedingungen repräsentieren,** müssen binär sein.

Beispiel: Modellierung einer Sensor-gesteuerten Ampelkreuzung:



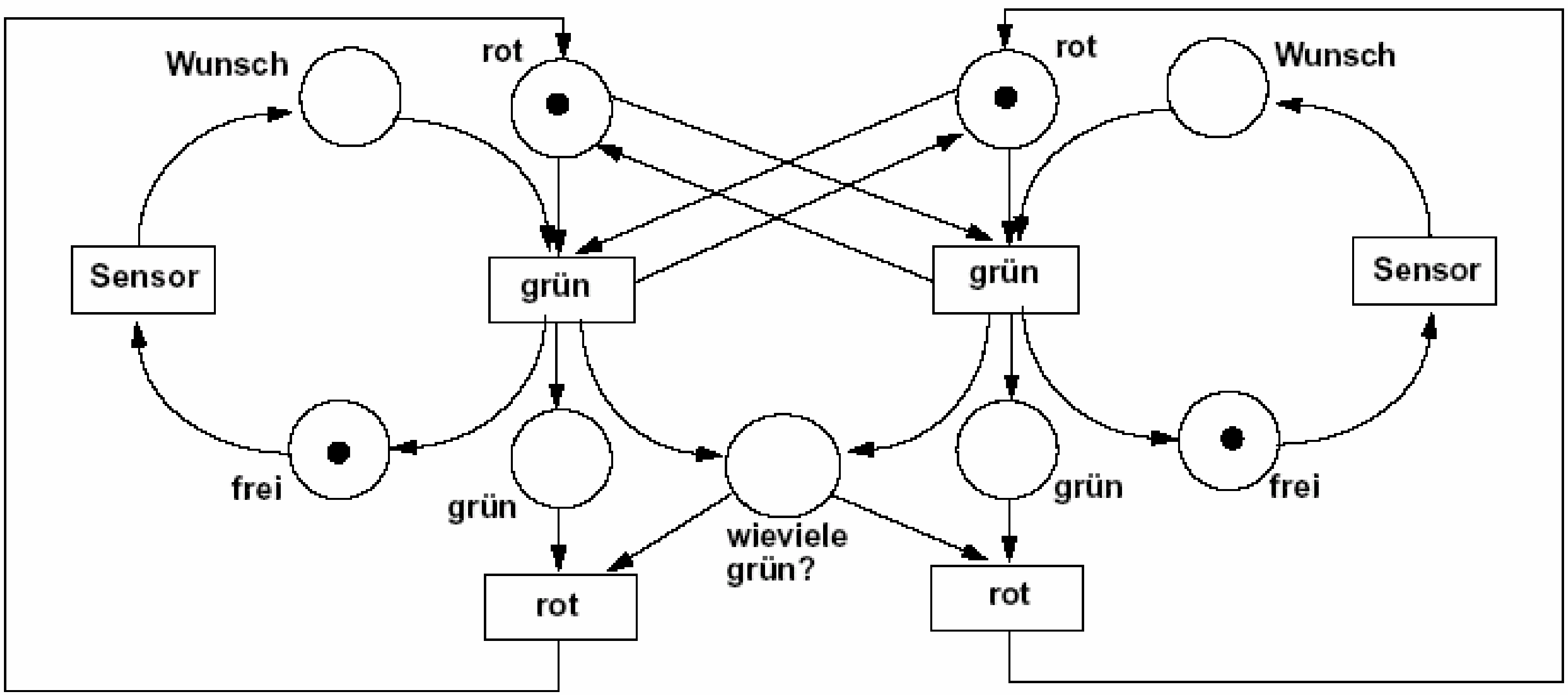
Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Sichere Netze

Ein Petri-Netz heißt **binär (sicher)**, wenn für alle aus M_0 erreichbaren Markierungen M und für alle Stellen s gilt $M(s) \leq 1$.

Petri-Netze, deren **Stellen Bedingungen repräsentieren,** müssen binär sein.

Beispiel: Modellierung einer Sensor-gesteuerten Ampelkreuzung:



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Lebendige Netze

- Petri-Netze modellieren häufig **Systeme, die nicht anhalten** sollen.
- Ein Petri-Netz heißt **schwach lebendig**, wenn es zu jeder von M_0 erreichbaren Markierung eine Nachfolgemarkierung gibt.
- Eine **Transition t heißt lebendig**, wenn es zu jeder von M_0 erreichbaren Markierung M´eine Markierung M gibt, die von M`erreichbar ist, und in der t schalten kann.

  Ein Petri-Netz heißt **lebendig**, wenn alle seine Transitionen lebendig sind.
- **Verklemmung**: Ein System kann unerwünscht anhalten,

  weil **das Schalten einiger Transitionen zyklisch voneinander abhängt.**

  Sei $\sigma \subseteq S$ eine Teilmenge der Stellen eines Petri-Netzes und

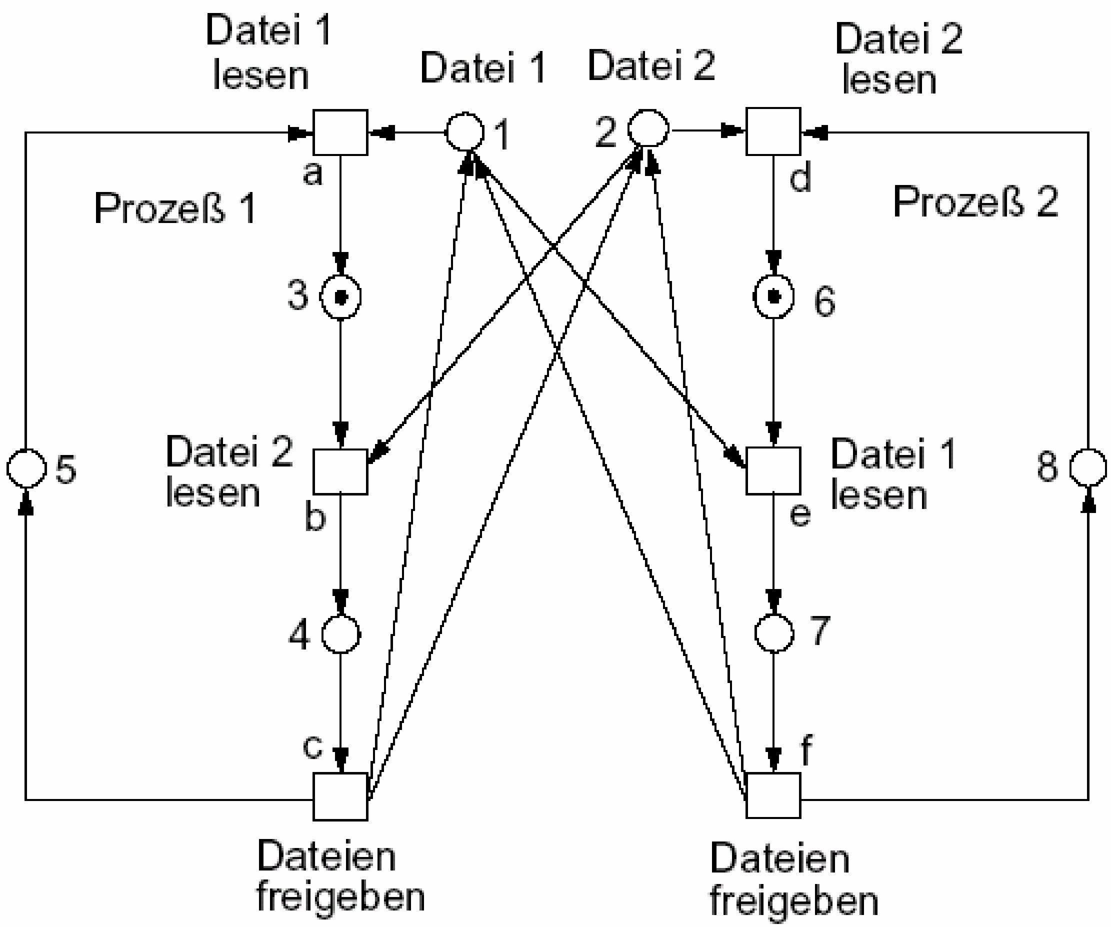
  Vorbereich (σ) $=\{t \mid \exists\, s \in \sigma : (t, s) \in F\}$,
  d.h. die Transitionen, die auf Stellen in σ wirken

  Nachbereich (σ) $=\{t \mid \exists\, s \in \sigma : (s, t) \in F\}$
  d.h. die Transitionen, die auf Stellen in σ als Vorbedingung haben

  Dann ist σ eine **Verklemmung, wenn Vorbereich (σ) $\subseteq$ Nachbereich (σ)**.

  Wenn **für alle $s \in \sigma$ gilt $M(s) = 0$**, dann kann es **keine Marken auf Stellen in σ** in einer Nachfolgemarkierung von M geben

Verklemmung



$\sigma = \{1, 2, 4, 5, 7, 8\}$

Vorbereich (σ)

$= \{b, c, e, f\}$

Nachbereich (σ)

$= \{a, b, c, d, e, f\}$

$M(\sigma) = 0$

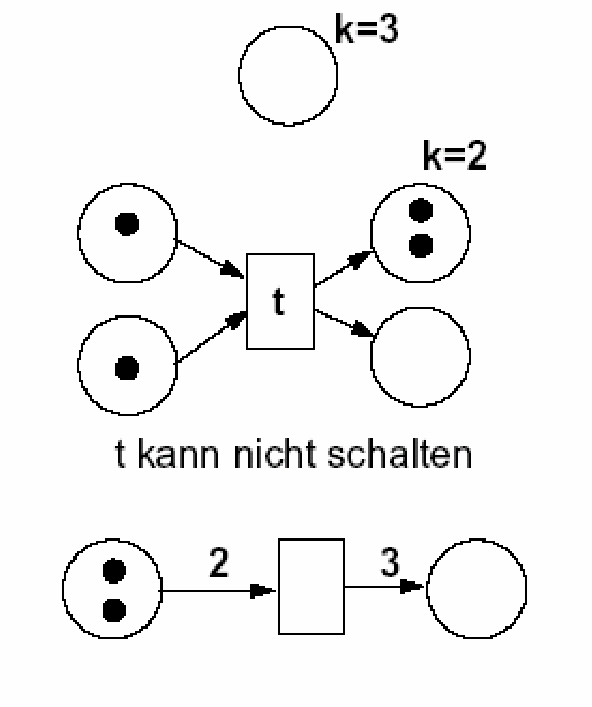
Kapazitäten und Gewichte

Man kann Stellen eine begrenzte Kapazität von $k \in$ IN Marken zuordnen.

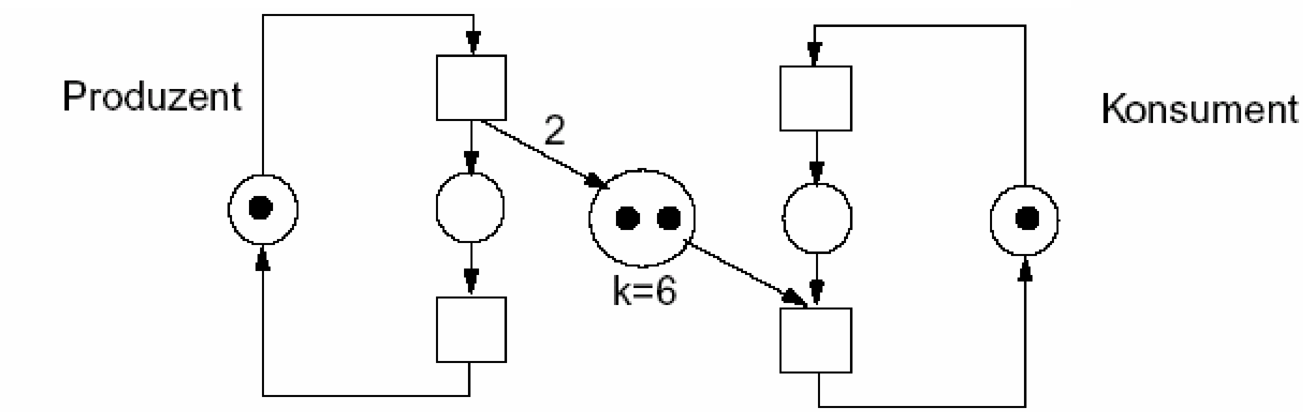
Die Bedingung, dass **eine Transition t schalten kann,** wird erweitert um:

> Die **Kapazität keiner der Stellen im Nachbereich von t darf überschritten** werden.

Kanten kann ein Gewicht $n \in$ IN zugeordnet werden: sie bewegen **beim Schalten n Marken.**



Beispiel: **Beschränkter Puffer**



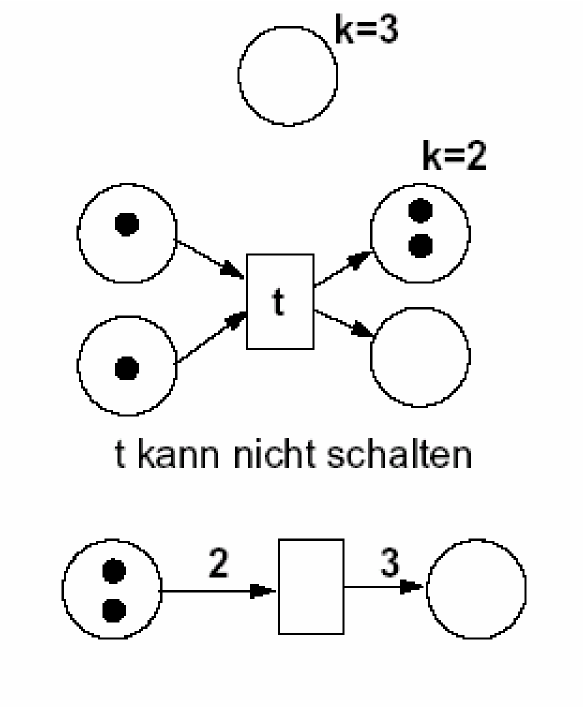
Kapazitäten und Gewichte

Man kann Stellen eine begrenzte Kapazität von k $\in$ IN Marken zuordnen.

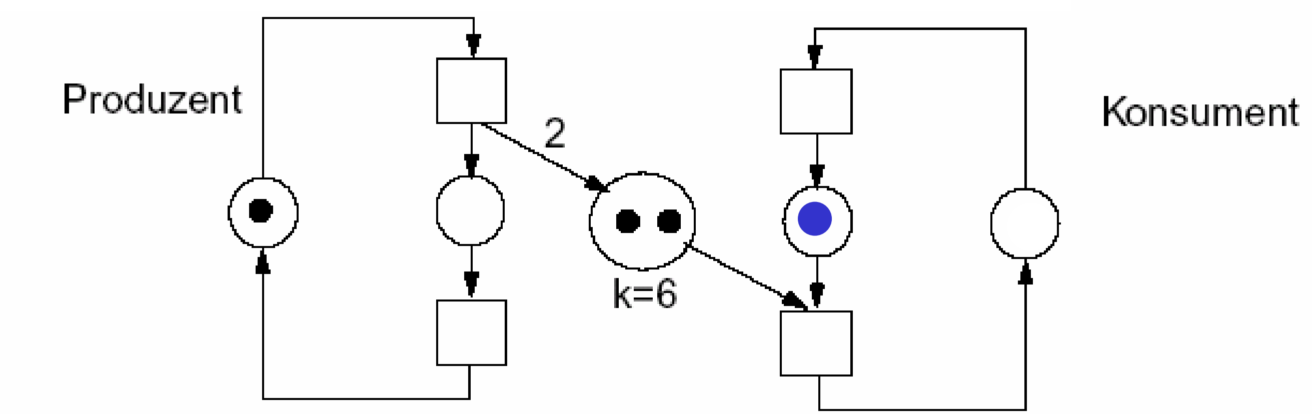
Die Bedingung, dass **eine Transition t schalten kann**, wird erweitert um:

> Die **Kapazität keiner der Stellen im Nachbereich von t darf überschritten** werden.

Kanten kann ein Gewicht n $\in$ IN zugeordnet werden: sie bewegen **beim Schalten n Marken.**



Beispiel: **Beschränkter Puffer**



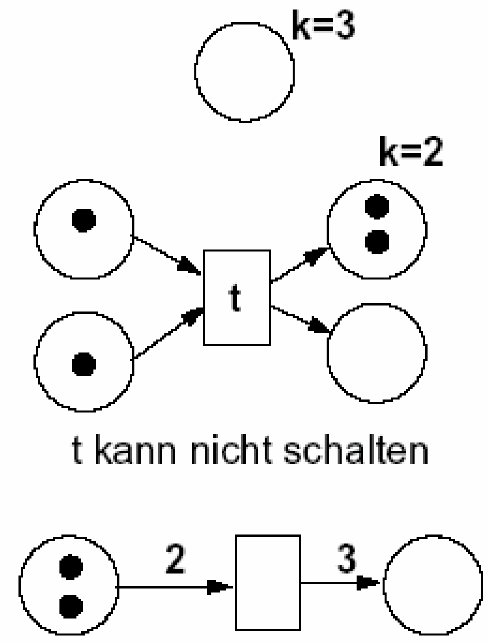
Kapazitäten und Gewichte

Man kann Stellen eine begrenzte Kapazität von k ∈ IN Marken zuordnen.

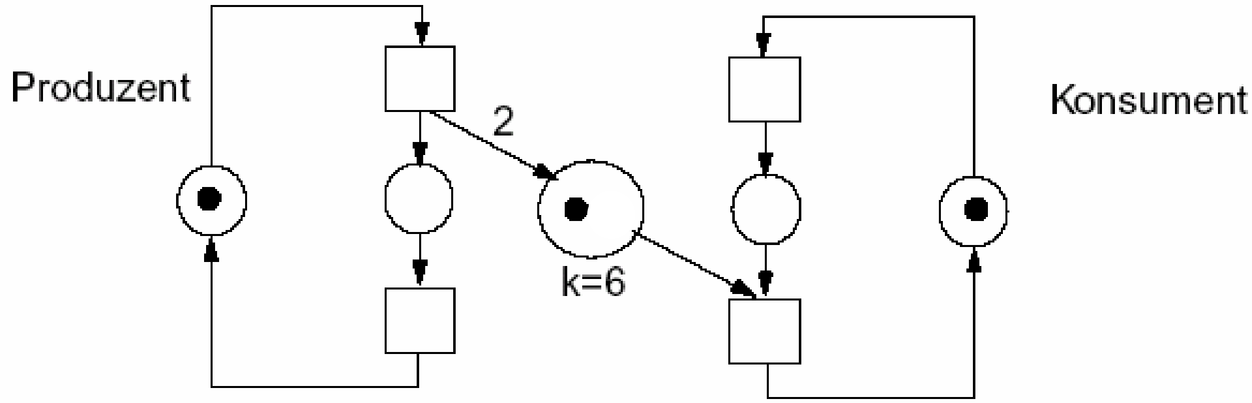
Die Bedingung, dass **eine Transition t schalten kann,** wird erweitert um:

> Die **Kapazität keiner der Stellen im Nachbereich von t darf überschritten** werden.

Kanten kann ein Gewicht n ∈ IN zugeordnet werden: sie bewegen **beim Schalten n Marken.**



Beispiel: **Beschränkter Puffer**

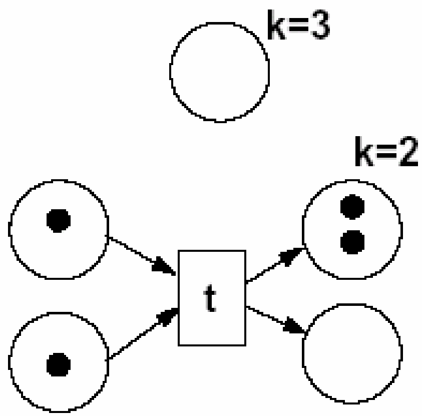


Kapazitäten und Gewichte

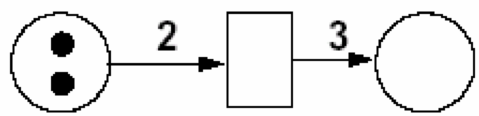
Man kann Stellen eine begrenzte Kapazität von $k \in$ IN Marken zuordnen.

Die Bedingung, dass **eine Transition t schalten kann,** wird erweitert um:

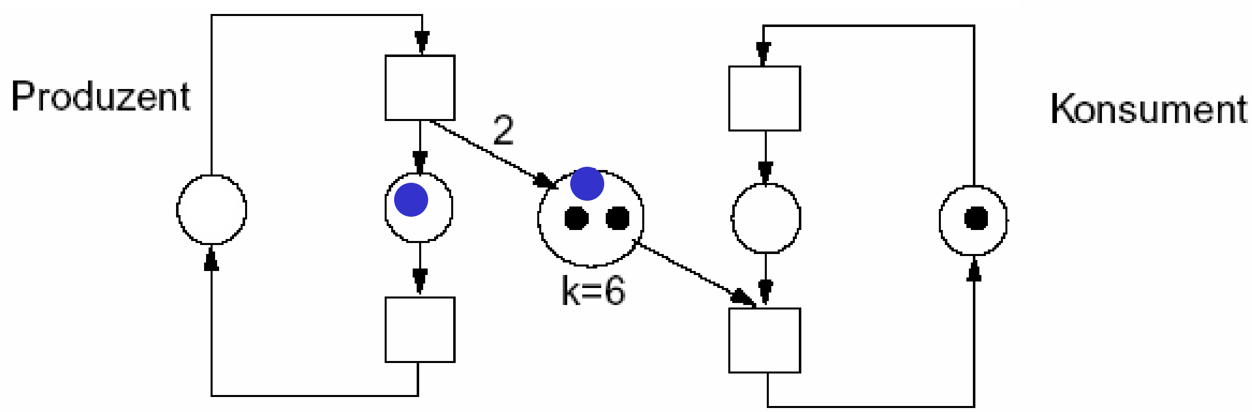
> Die **Kapazität keiner der Stellen im Nachbereich von t darf überschritten** werden.



Kanten kann ein Gewicht $n \in$ IN zugeordnet werden: sie bewegen **beim Schalten n Marken.**

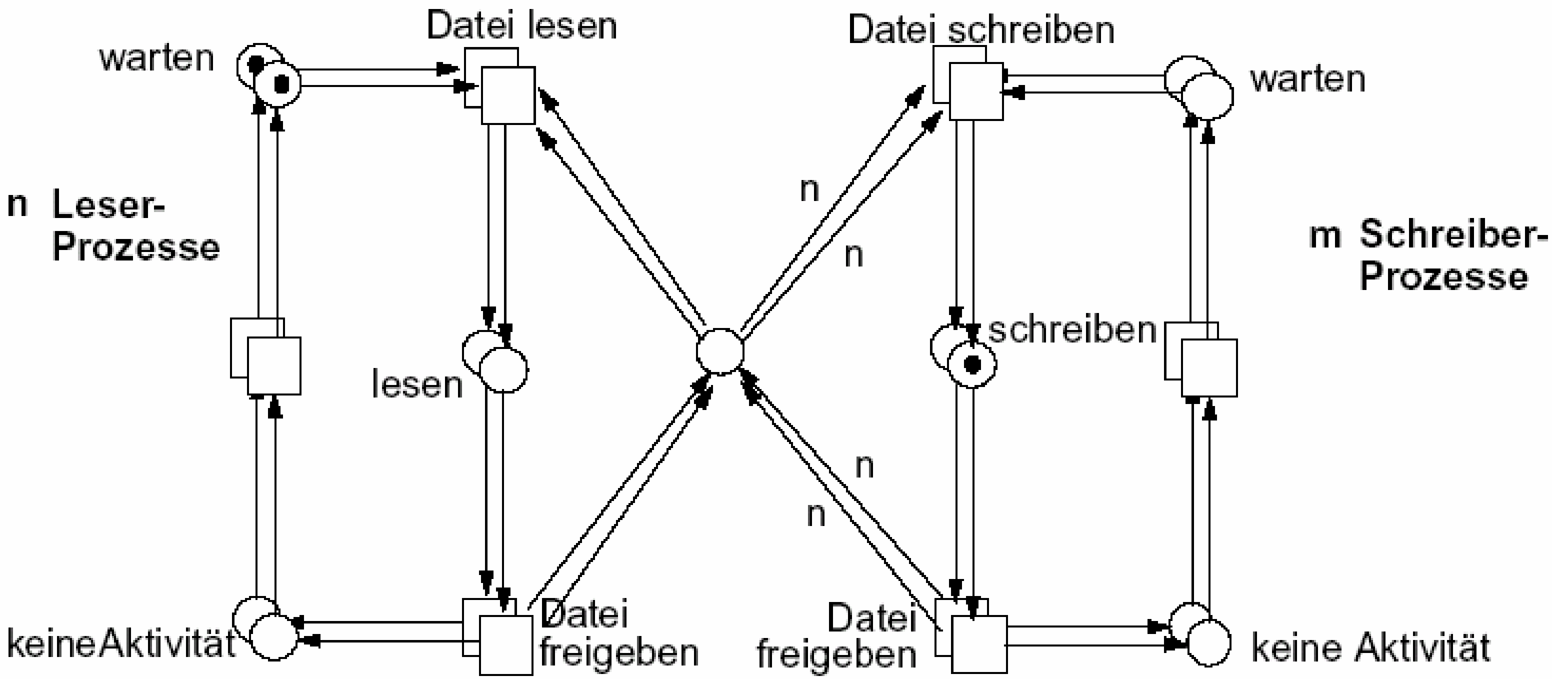


Beispiel: **Beschränkter Puffer**



Leser- /Schreiber-System

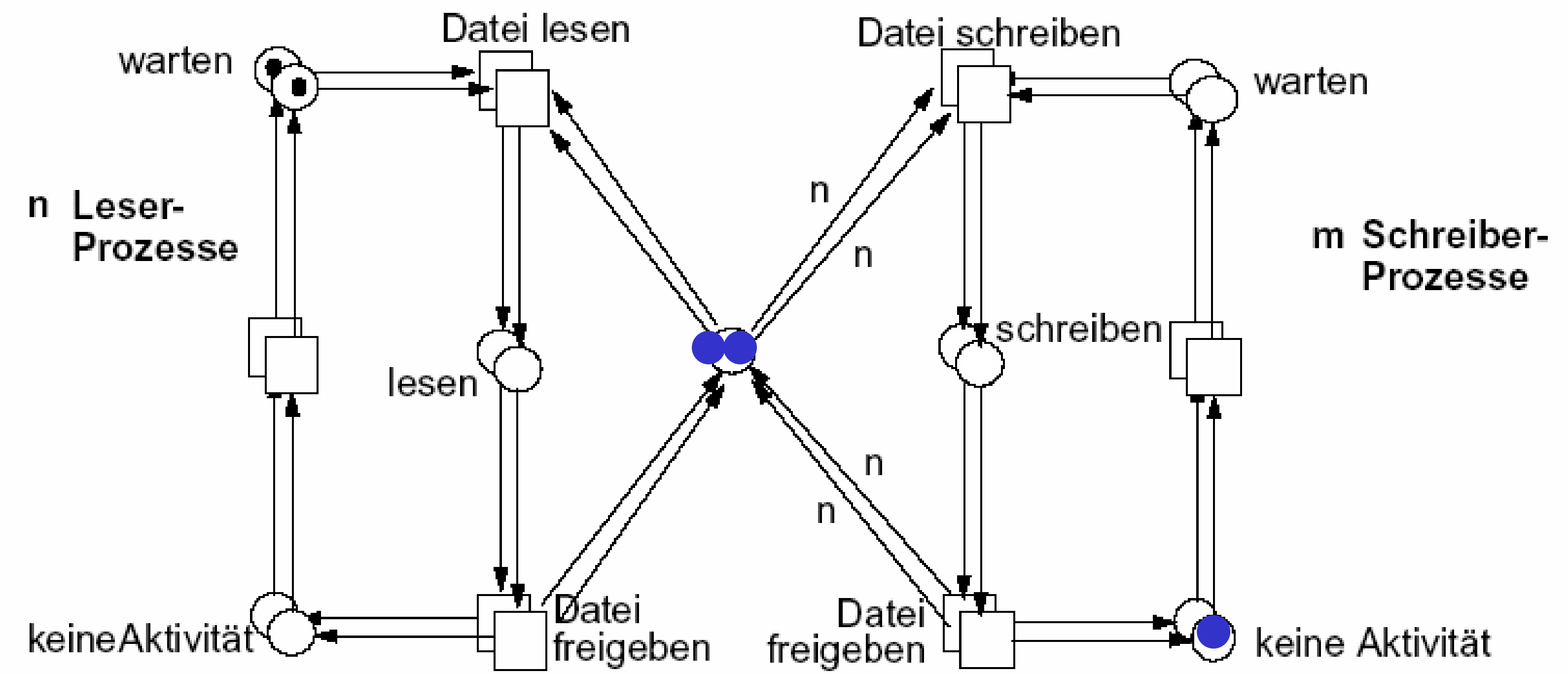
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

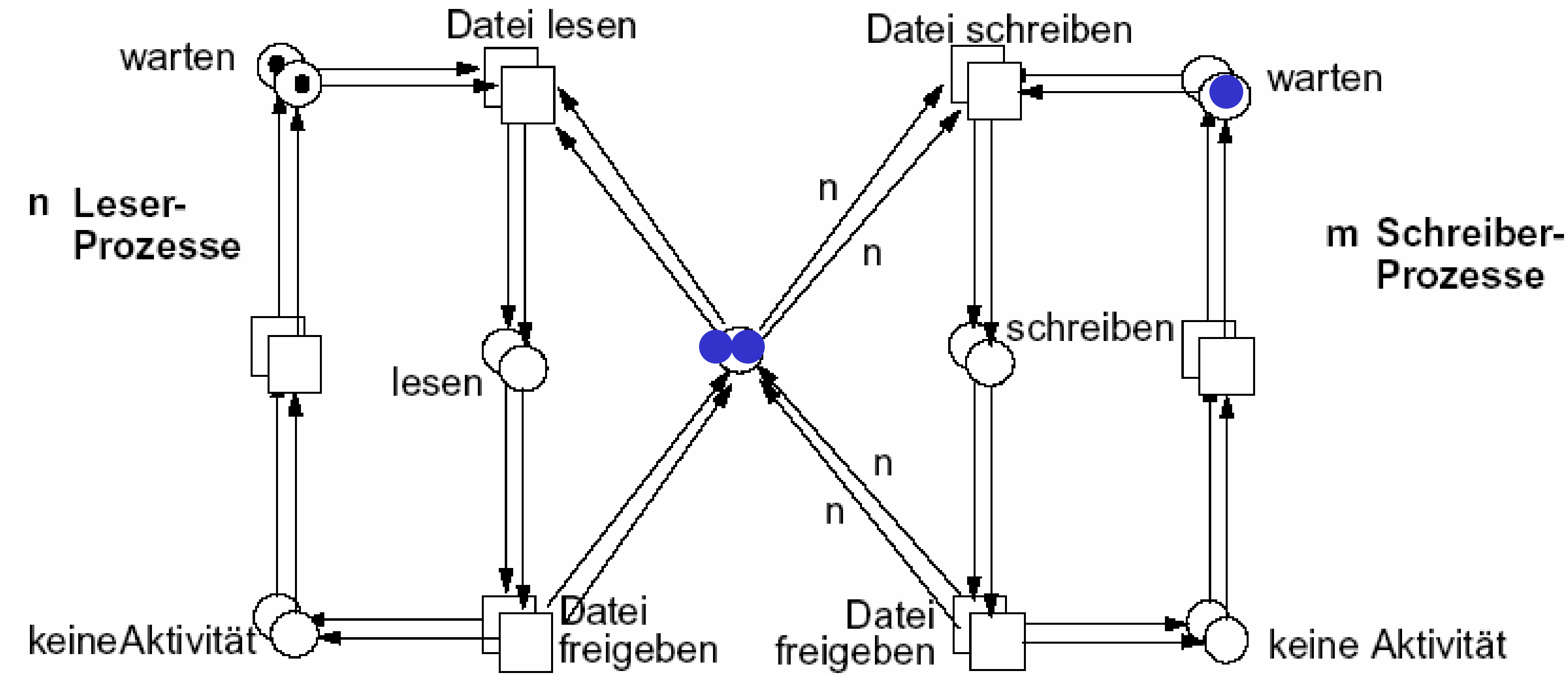
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

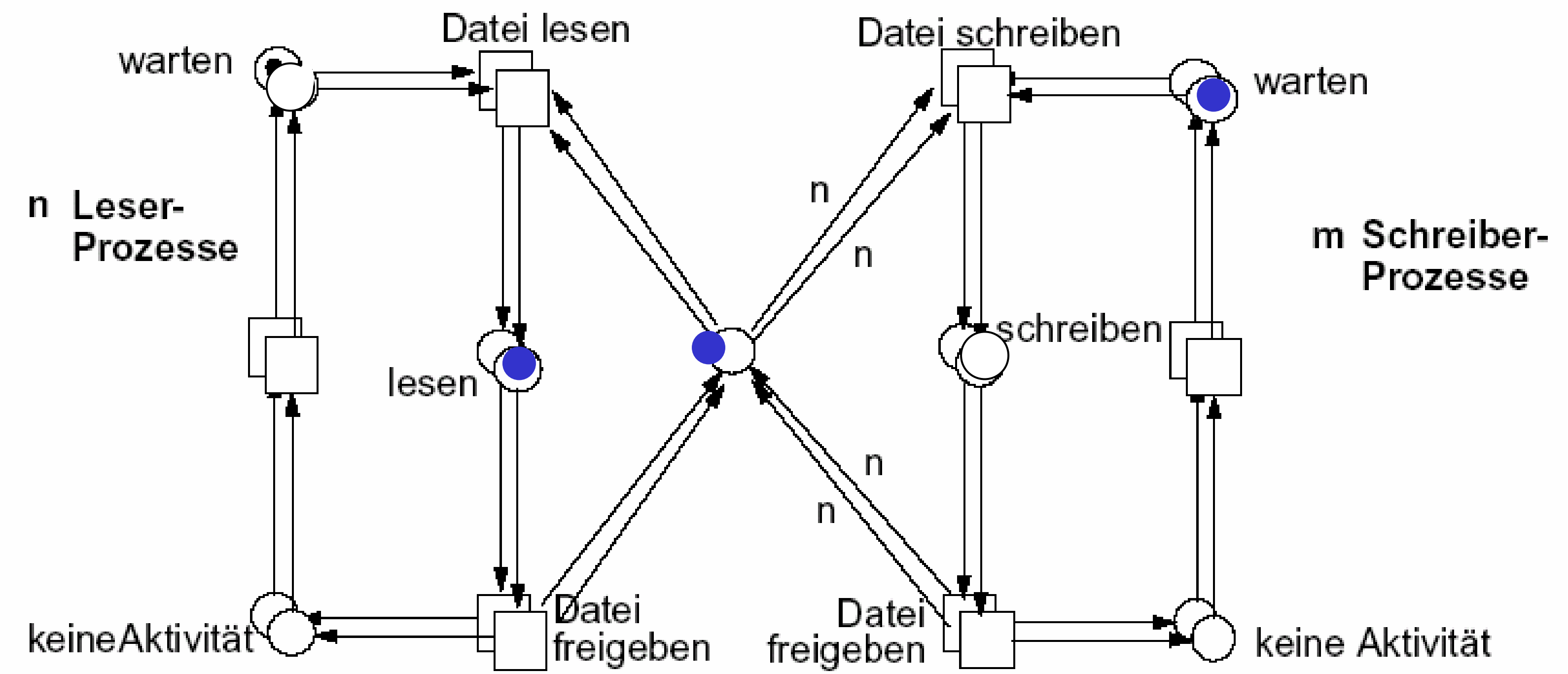
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung:** ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

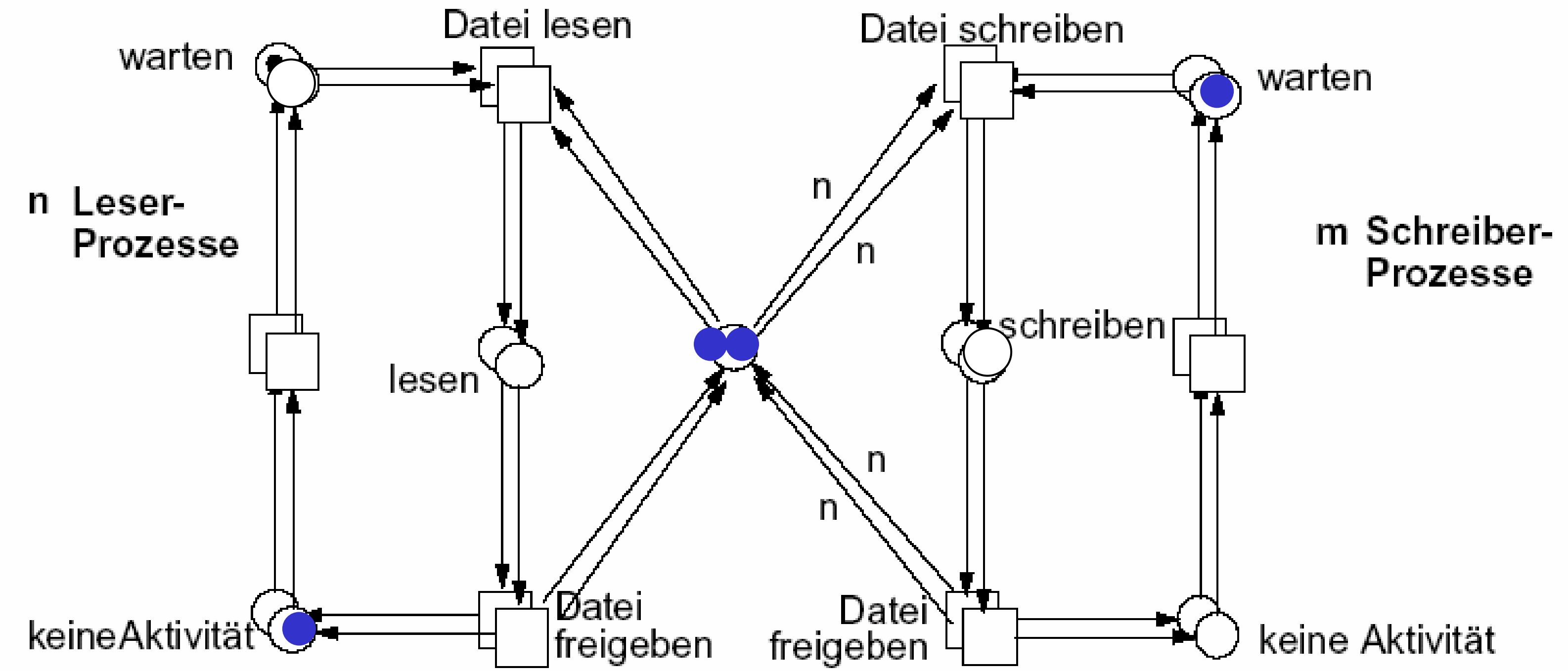
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

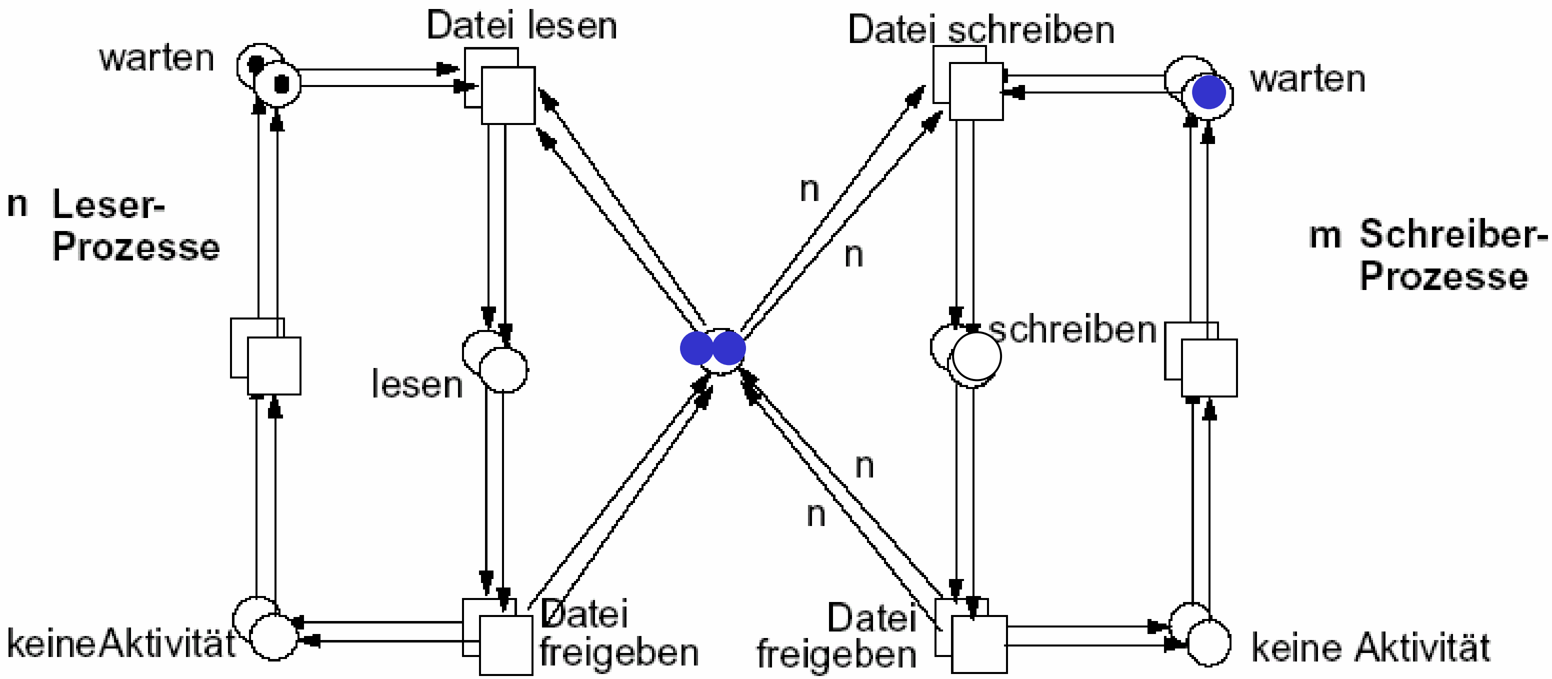
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

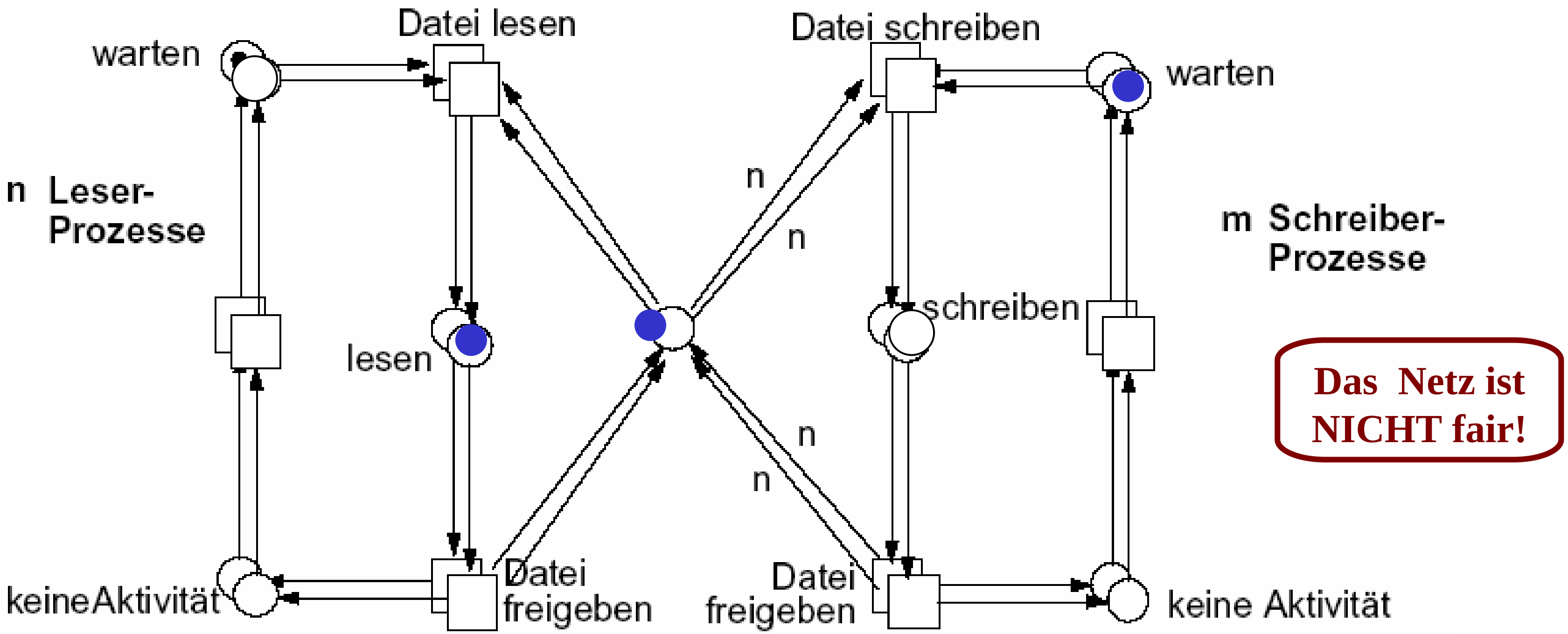
- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Leser- /Schreiber-System

- **n Leser-Prozesse und m Schreiber-Prozesse** operieren auf derselben Datei.
- **Mehrere Leser** können **zugleich** lesen.
- Ein Schreiber darf **nur dann schreiben, wenn kein anderer** Leser oder Schreiber **aktiv** ist.
- **Modellierung**: ein Schreiber entzieht der Synchronisationsstelle alle n Marken.



Aus B. Baumgarten: Petri-Netze, Bibliographisches Institut & F.A.Brockhaus AG, 1990

Zusammenfassung

- Petri-Netze wurden 1962 von C. A. Petri eingeführt. Sie dienen zur Modellierung von Abläufen mit nebenläufigen Prozessen und kausalen Beziehungen.
- Petri-Netze basieren auf bipartiten gerichteten Graphen:
  - Knoten repräsentieren Bedingungen, Zustände bzw. Aktivitäten.
  - Kanten verbinden Aktivitäten mit ihren Vor- und Nachbedingungen.
  - Die Knotenmarkierung repräsentiert den veränderlichen Zustand des Systems.

Literatur

- Kastens: Petrinetze, Vorlesungsmitschrift
- G. Goos: Vorlesungen über Informatik, Band 1, Springer-Verlag, 1995