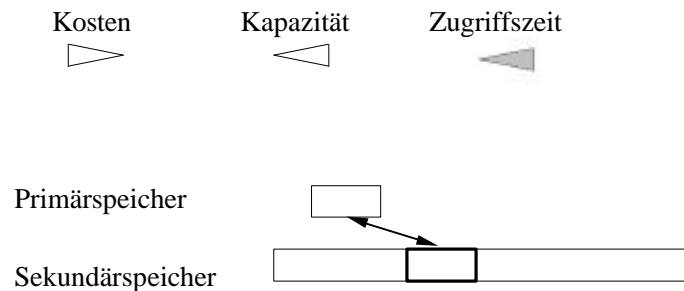


### 3. Leistungsbewertung von Betriebssystem-Komponenten

#### 3.1. Speicherverwaltung

- **Grundproblem:** Speicherhierarchie

Proz.-Reg. – Cache – Hauptspe. – int. Massensp. – ext. Massensp. – ...



- **Lösungsmöglichkeiten und deren Probleme**

– *Segmentierter Speicher:*

Freispeicherlisten → externe Fragmentierung  
 Auffüllen und Kompaktifizieren → Aufwand

– *Seitenorientierte Systeme:*

Arbeitsmengen → Fenstergröße; Seitenflattern  
 statische Ersetzungsstrategien → Seitenfehlerrate

#### 3.1.1. Freispeicherlisten

- **Berechnung der mittleren Anzahl  $EF$  freier Segmente bei abhängigen Lebensdauern im stationären Fall**

– *Bezeichnungen:*

Riegel: zusammenhängender Bereich belegter Blöcke

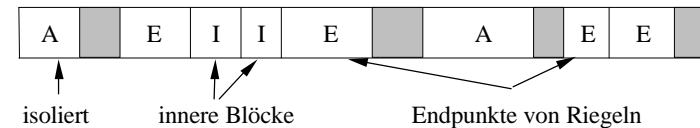
$F$ : Anzahl der freien Blöcke

$B$ : Anzahl der belegten Blöcke Zufallsgrößen!

– *Annahme:*

Freigabe und unmittelbar anschließende Belegung sind unabhängig und gleichverteilt. Damit ist  $B$  konstant.

– *Betrachtung eines typischen Speicherzustands  $z$ :*



$p = p(z)$ : Wahrscheinlichkeit dafür, daß einzulagerndes Segment nicht genau in eine Lücke paßt.

Dann folgt bei starker Streuung der Segmentgrößen:

$$EF \approx \frac{B}{2}$$

KNUTHs 50%-Regel

• **Freispeicherformel**

Es sei im stationären Zustand

- $f$  durchschnittliche Freiraumgröße
- $s$  durchschnittliche Segmentgröße
- $k$  Verhältnis von  $f$  zu  $s$
- $m$  Hauptspeichergröße
- $u$  Anteil des freien (unbelegten) Speichers.

Dann ist

$$u = \frac{k}{k+2}$$

oder

$$k = \frac{2u}{1-u}$$

Interpretation:

$$k = \frac{1}{2} \Rightarrow u = \frac{1}{5} = 20\%$$

$$k = \frac{1}{4} \Rightarrow u = \frac{1}{9} \approx 11\%$$

Simulationsuntersuchungen:

$$s = \frac{m}{10} \rightarrow u = 0,1 \Rightarrow k = 0,22$$

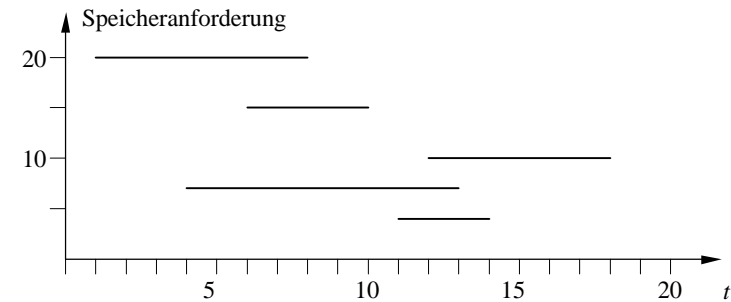
$$s = \frac{m}{3} \rightarrow u = 0,5 \Rightarrow k = 2$$

**3.1.2. Auffüllen und Kompaktifizieren**

• **Beschreibung**

Zu zufälligen Zeiten  $t_i$  wird eine Anforderung in zufälliger Größe  $x_i$  gestellt, die eine zufällige Dauer  $d_i$  im Primärspeicher verbleibt.

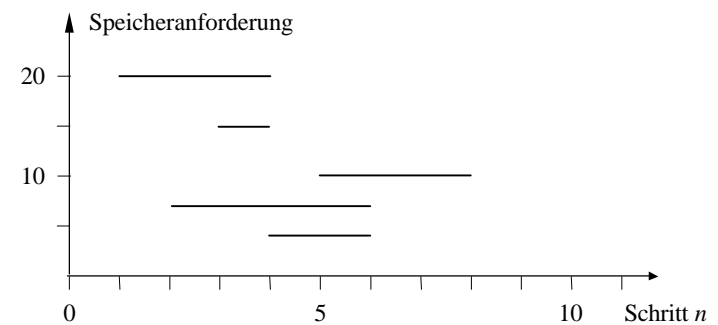
Beispiel. Folge (1,20,7) (4,7,9) (6,15,4) (11,4,3) (12,10,6)



Transformation: Zeit  $\rightarrow$  Schritt: Zeitpunkt der nächsten Anforderung

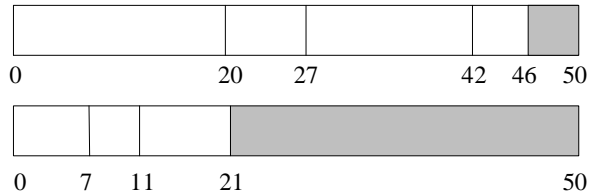
Beispiel. Folge (20,3), (7,4), (15,1), (4,2), (10,3).

„Reservierungsschaubild“:



- **Vorgehen**

Neue Segmente anhängen – freiwerdende kennzeichnen – kompaktifizieren.



- **Formale Beschreibung**

Stochastischer Prozeß mit diskreter Zeit („Schritt“: Zeitpunkt der  $i$ -ten Anforderung)

$$(X_1, D_1), (X_2, D_2), \dots \quad (\text{Größe/Lebenszeit})$$

Folge unabhängiger ganzzahliger, identisch verteilter zufälliger Vektoren,  $X_i/D_i$  unabhängig.

- **Leistungsanalyse**

– Kompaktifizierungs-Wahrscheinlichkeit: Wahrscheinlichkeit für die Anzahl  $N$  der Segmente, die in einem Speicher der Größe  $m$  Platz finden, ohne daß Kompaktifizierung nötig ist;

– Kompaktifizierungsaufwand  $A$ : Gesamtlänge der zu verschiebenden Segmente.

- **Kompaktifizierungs-Wahrscheinlichkeit  $\mathbf{P}(N = n)$**

$m$ : Speichergröße

$F$ : Verteilungsfunktion von  $X_i$

$$\begin{aligned} \mathbf{P}(N \geq n) &= \mathbf{P}\left(\sum_{i=1}^n X_i \leq m\right) = \\ &= F_{\sum X_i}(m) = \\ &= F^{(n)}(m). \end{aligned}$$

Damit folgt:

$$\begin{aligned} \mathbf{P}(N = n) &= \mathbf{P}(N \geq n) - \mathbf{P}(N \geq n+1) = \\ &= F^{(n)}(m) - F^{(n+1)}(m). \end{aligned}$$

*Spezialfall:* Segmentgröße POISSON-verteilt, Parameter  $\lambda$ :

$$\mathbf{P}(N = n) = \sum_{k=0}^m \frac{e^{-n\lambda}}{k!} \cdot \lambda^k (n^k - e^{-\lambda} (n+1)^k)$$

• **Kompaktifizierungsaufwand A**

– Allgemein:

$$A = \sum_{i=1}^N X_i \cdot \Delta_{D_i > N-i}$$

mit

$$\Delta_{\text{bedingung}} = \begin{cases} 1 & \text{bedingung ist wahr} \\ 0 & \text{bedingung ist falsch} \end{cases} \quad (\text{Indikatorfunktion}).$$

– Spezialfall:

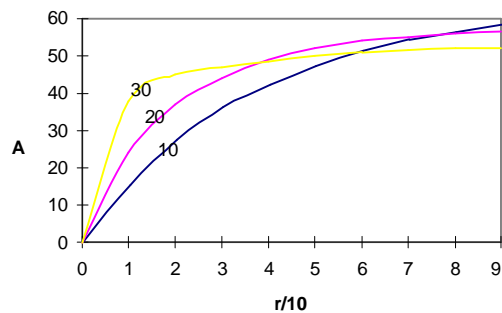
Konstante Segmentlänge  $s$ ,  
Reservierungsdauer geometrisch verteilt, im Mittel

$$r = \frac{1}{p}, \quad \text{d.h. } \mathbf{P}(D_i = k) = p(1-p)^{k-1}, \quad k \in \mathbb{N}^+.$$

Dann ist  $N$  konstant mit  $N = \left\lfloor \frac{m}{s} \right\rfloor$  ( $m$ : HS-Kapazität). Damit folgt:

$$\mathbf{E} A = rs \left( 1 - \left( 1 - \frac{1}{r} \right)^{\left\lfloor \frac{m}{s} \right\rfloor} \right)$$

– Bewertung:



$m = 60$   
Parameter:  $s$

**3.1.3. Arbeitsmengenmodell**

• **Begriff**

Arbeitsmenge eines Prozesses mit der Seitenreferenzfolge  $R_1 R_2 \dots$  zum Zeitpunkt  $t$  mit dem Arbeitsmengenparameter  $k$ :

$$W(k, t) = \{R_t, R_{t-1}, \dots, R_{t-k+1}\}.$$

Annahme: Seitenreferenz ist stationärer Prozeß.

Dann ist  $W(k, t)$  „unabhängig von  $t$ “, und es sei

$$w(k) := \mathbf{E}|W(k, t)|.$$

Weitere Annahme: Seitenreferenz ist stationäre ergodische  $???\text{?}\text{?}\text{?}\text{?}\text{?}$ sche Kette.

Dann läßt sich  $w(k)$  berechnen:

$$w(k) = \frac{\sum_{i=1}^n \sum_{j=1}^k \min(k, j) \cdot f_{ii}(j)}{\sum_{j=1}^{\infty} j \cdot f_{ii}(j)}$$

mit

$n$  Adreßraumgröße

$T^{(i)}$  Interferenz - Intervall

$$f_{ii}(j) = \mathbf{P}(T^{(i)} = j | R_0 = i).$$

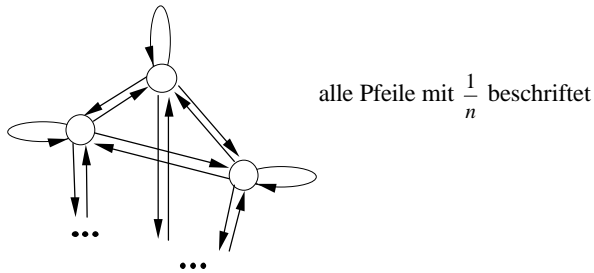
- **Spezielle Modelle**

Es bezeichne  $n$  die Größe des Programm-Adreßraums in Seiten.

- **Unabhängige Seitenreferenzen:**

Es sei  $R_1, R_2, \dots$  Folge unabhängiger, identisch gleichverteilter Zufallsgrößen.

Intensitätsgraph der zugehörigen  $?????$ -Kette:



Dann ist

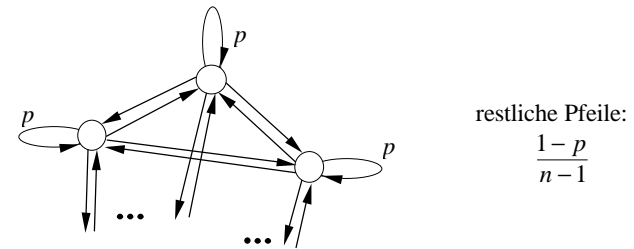
$$w(k) = n \cdot \left(1 - \left(1 - \frac{1}{n}\right)^k\right).$$

- **Gleichverteilte Seitenwechselwahrscheinlichkeit:**

Es sei  $p$  die Wahrscheinlichkeit für das Beibehalten einer Seite, die restlichen Seiten werden gleichverteilt referenziert, d.h.

$$P_{ij} = \begin{cases} p & i = j \\ \frac{1-p}{n-1} & i \neq j \end{cases} \quad i, j = 1, \dots, n.$$

Intensitätsgraph:



Dann ist

$$w(k) = n - (n-1) \left(1 - \frac{1-p}{n-1}\right)^{k-1}.$$

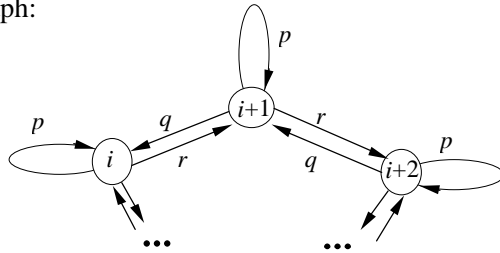
– Irrfahrtmodell:

Seitenreferenzen betreffen nur benachbarte Seiten:

$$p_{i,i} = p \quad p_{i,i-1} = q \quad p_{i,i+1} = r \quad (p + q + r = 1)$$

$$p_{i,j} = 0 \quad \text{sonst}$$

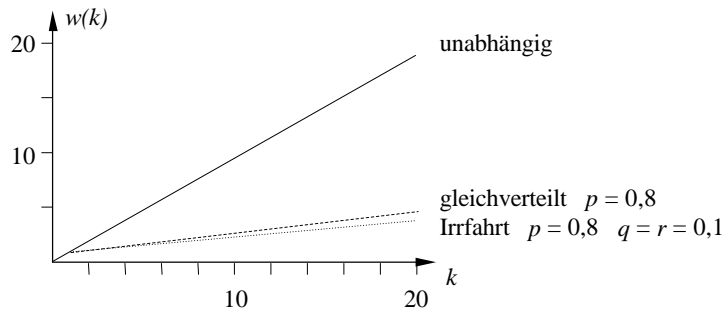
Intensitätsgraph:



Dann ist

$$w(k) = k - (k-1)p - 2qr \cdot \sum_{t=2}^{k-1} \left[ (k-t) \cdot \sum_{j=0}^{\lfloor \frac{t-2}{2} \rfloor} \binom{t-2}{2j} (qr)^j \binom{2j}{j} \cdot \frac{1}{j+1} p^{t-2-2j} \right]$$

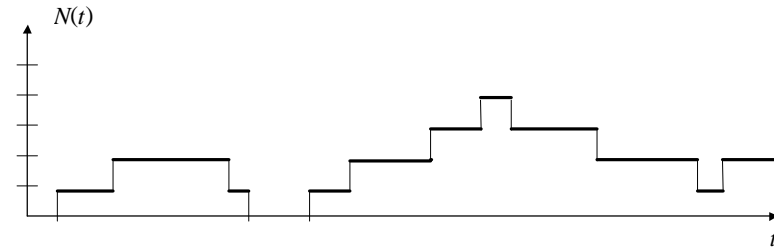
– Bewertung: Beispiel:  $n = 100$



### 3.2. Externspeicher-Zugriff

• Allgemein: *M/G/1/∞-Modell*.

$N(t)$ : Anzahl der Forderungen im System zur Zeit  $t$ .



Betrachtung des Prozesses zu den Abfertigungszeitpunkten einer Forderung

eingebettete MARKOVsche Kette

Formeln von POLLACZEK/? ? ? ? ? für mittlere Verweilzeit:

$$\mathbf{E} X_n = r + r^2 \cdot \frac{1 + v^2}{2(1 - r)}$$

– Speziell: Magnetplattenspeicher.

Es sei

$T$  Umdrehungszeit

$N$  Anzahl der Sektoren pro Spur

$K$  Anzahl der Spuren pro Plattenseite

Konstanten!

• **Positionierzeit**

Betrachtung des mittleren Abstands  $D$  zweier nacheinander zu positionierender Spuren.

– *Annahmen:*

\* Forderungsstrom an die Platte sei POISSONsch mit der Intensität  $I$ .

\* Die angeforderten Spurnummern  $S$  seien gleichverteilt, d.h.

$$P(S = k) = \frac{1}{K}, \quad k = 1, \dots, K.$$

– *Bedienungsstrategie FIFO*

Offenbar gilt:

$$D = |S_1 - S_2|, \quad S_1, S_2 \text{ identisch wie } S \text{ verteilt.}$$

Damit ist

$$ED_{\text{FIFO}} = \sum_{k_1=1}^K \sum_{k_2=1}^K \frac{1}{K^2} \cdot |k_1 - k_2|$$

	1	2	3	...	K
1					
2					
3					
...					
K					

$$ED_{\text{FIFO}} = \frac{K^2 - 1}{3K}$$

Asymptotisches Verhalten ( $K$  groß):

$$ED_{\text{FIFO}} \sim \frac{K}{3}$$

• **Weitere Verbesserungen**

– *SSTF Shortest Seek Time First:*

$$ED_{\text{SSTF}, n} \sim \frac{K}{n+1} \quad n: \text{Anzahl der Aufträge in Warteschlange}$$

Vergleich mit FIFO:

– *SCAN (Fahrstuhl-Algorithmus):*

Überfahren der gesamten Platte dauere die (konstante) Zeit  $t$ .

Dann gilt:

$$ED_{\text{SCAN}} = \frac{K}{It} (1 - e^{-It}).$$

Offenbar gilt für großes  $K$ :

$$ED_{\text{SCAN}} < ED_{\text{FIFO}} \quad \text{für} \quad I > \frac{3}{t}.$$

### 3.3. Leistungsanalyse eines replizierten Dateisystems

#### 3.3.1. Modifikationen von PETRI-Netzen

- **Bedingungs-Ereignis-Netze**

$$m: P \rightarrow \{0, 1\} \quad \text{d.h. } m(p) \leq 1 \quad \forall p \in P.$$

- **Platz-Transitions-Netze**

$$K: P \rightarrow \mathbb{N}^+ \quad \text{Kapazitätsfunktion}$$

Plätze können festgelegte Anzahl von Marken aufnehmen;

$$V: F \rightarrow \mathbb{N}^+ \quad \text{Kanten-Vielfachheit}$$

Kanten transportieren feste Anzahl von Marken.

- **Gefärbte PETRI-Netze**

– *Motivation:* Große Netze mit gleichmäßigem Aufbau  
z.B. Philosophenproblem, Ampel.

– *Multimenge  $m$  über einer endlichen Menge  $M$ :*

Abbildung  $m: M \rightarrow \mathbb{N}$ ,

geschrieben als formale Summe  $m = \sum_{x \in M} m(x) * x$

Beispiel:  $M = \{\text{Apfel, Birne}\}$ ;  $m = 2 * \text{Apfel} + 1 * \text{Birne}$ .

Es sei  $\hat{M} = \mathbb{N}^M$  Menge aller Multimengen über  $M$ .

– *Netzdefinition.* Das Tupel  $(P, T, F, C, V, m_0)$  heißt *gefärbtes PETRI-Netz* (mit der endlichen Farbmenge  $\mathbf{C}$ ), wenn gilt:

(1)  $(P, T, F)$  ist ein PETRI-Netz.

(2)  $C: P \cup T \rightarrow \mathbf{P}(\mathbf{C}) \setminus \{\emptyset\}$ .

(3)  $V: F \rightarrow \hat{C}(p)^{C(t)}$ ,  $F$ : Kantenmenge des PN.

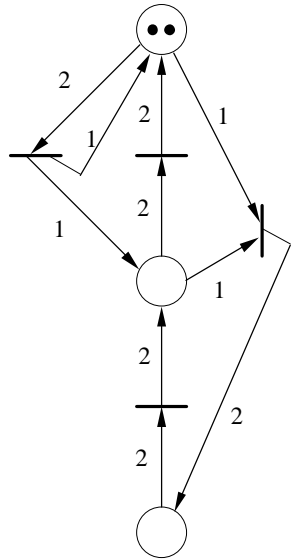
d.h.  $V(k): C(t) \rightarrow \hat{C}(p)$  bei  $k = (t,p) \vee k = (p,t)$

(4)  $m_0: p \mapsto m \in \hat{C}(p) \quad \forall p \in P$ .



– Beispiel. Ampel.

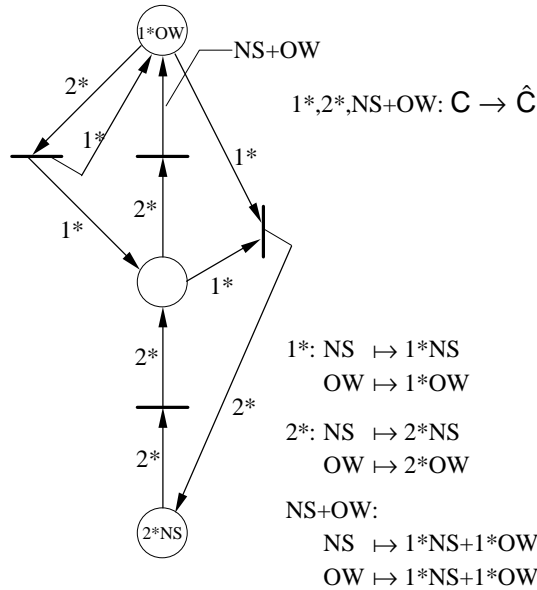
Gewöhnliches Netz  
für 1 Richtung



Gefärbtes Gesamt-Netz;

$C = \{NS, OW\}$ ;

$C(t) = C(p) = C \quad \forall p \in P, t \in T.$



• **Prädikat-Transitionen-Netze**

PN mit Anfangsmarkierung sowie

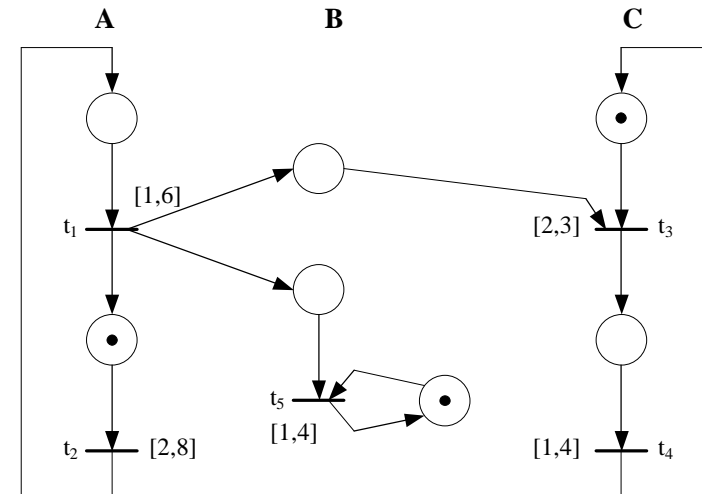
- \* Transitionen mit Formeln des Prädikatenkalküls (über einem Individuenbereich) beschriftet
- \* Plätze mit einer Angabe, wieviel Komponenten die Marken besitzen
- \* Kanten mit Tupeln von Variablen (über dem Individuenbereich).

• **Zeitbewertete PETRI-Netze**

Durch Zeiten bewertbar sind:

Marken – Plätze – Kanten – Transitionen: Schaltdauer (timed PN)  
Schaltintervall (time PN).

– Beispiel. Prozeß A sendet eine Nachricht an Prozeß B und C, die mit unterschiedlichen Geschwindigkeiten empfangen werden. C verarbeitet Nachricht, A bereitet nächste Nachricht vor. Schaltintervalle!



• **Stochastische PETRI-Netze**

Jede Transition besitzt eine zufällige Schaltzeit.

\* Alle Schaltzeiten exponentiell verteilt:

Erreichbarkeitsgraph entspricht homogener M? ??? -Kette → Vorgehen wie in Bedienungstheorie.

\* Sonst: nur Simulation möglich.

### 3.3.2. Votierungsverfahren für replizierte Dateisysteme

Literatur: DUGAN, J.B.; G. CIARDO: Stochastic Petri Net Analysis of a Replicated File System. IEEE Transact. of Software Eng., Vol. 15/4, 4/1989.

- **Motivation**

Leistung – Konsistenz – Verfügbarkeit – Fehlertoleranz – Transparenz  
Partitionierung!

- **Voraussetzungen**

Struktur eines Datenblocks:

Datenblocknummer
Versionsnummer
Sperrzustand
Datenblockinhalt

Beispiel.

Rechner 1	Rechner 2	Rechner 3
k	k	k
5	6	6
XX	YY	YY

„Mehrheit“ von Rechnern muß aktuelle Version besitzen.

Beispiel.

R <sub>1</sub>	R <sub>2</sub>	R <sub>3</sub>
k	k	k
7	7	6
ZZ	ZZ	YY

Kontrolle: Transparent – optimal / pessimal.

- **Votierungsverfahren**

– **Votum:** Stimme (evtl. mit Gewicht) derjenigen Rechner, die Zugriff auf Datenblock gestatten.

– **Quorum QU:** untere Schranke für eine Anzahl von Stimmen, um Zugriff zu gestatten.

– **Kriterien für die Wahl von QU:**

Bei erfolgreichem Votum muß sich unter den Rechnern, die Zugriff gestatten, wenigstens einer mit einer aktuellen Version befinden.

Bei Partitionierung muß verhindert werden, daß sich parallele Teilkonsistenzen entwickeln.

Bei Abstürzen muß wenigstens eine eingeschränkte Weiterarbeit möglich sein.

– **Mehrheitsvotum:**

$$QU := \left\lfloor \frac{r}{2} \right\rfloor + 1 \quad r: \text{Anzahl der Replikate der Datei}$$

– **Votierung mittels „Zeugen“ (witnesses):**

Zeuge besitzt lediglich die Zustandsinformationen, nicht jedoch den Datenblock.

Beispiel.



R <sub>1</sub>	R <sub>2</sub>	R <sub>3</sub>
k	k	k
5	6	6
XX	YY	

### 3.3.3. Modellierung durch ein PETRI-Netz

- **Voraussetzungen und Annahmen**

- Jeder Rechner hat 1 Votum.
- *Ablauf:*  
Bei Zugriffsanforderung auf Dateiblock wird Nachricht an jeden Rechner gesandt, Quorum zu bilden. Jeder verfügbare Rechner antwortet.  
Reihenfolge der Auswahl:  
1 aktuelle Kopie – aktuelle Zeugen – weitere aktuelle Kopien – veraltete Zeugen – veraltete Kopien.
- Rechner kann ausgefallen sein und wird instandgesetzt.
- Rechner fallen nicht während der Bildung des Quorums aus.
- Kann Quorum nicht gebildet werden, so wird System manuell rekonstruiert.
- Zeiten sind exponentiell verteilt.

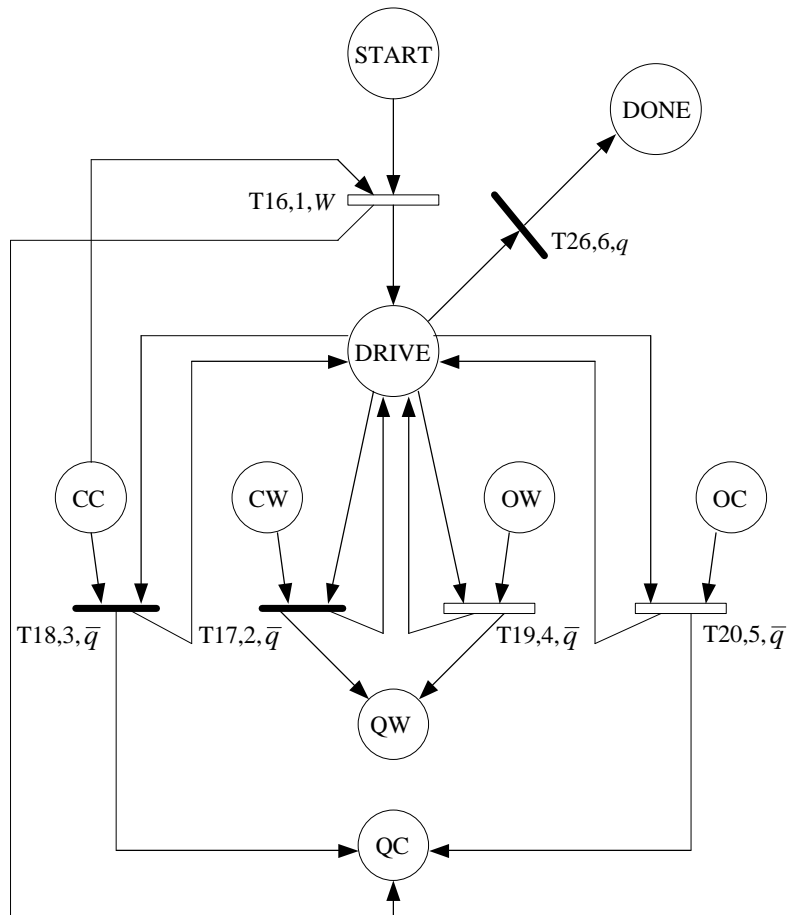
- **Beschreibung des PETRI-Netzes**

- Verallgemeinertes stochastisches PN; besitzt
  - gewöhnliche Transitionen 
  - zeitbewertete Transitionen 
- „Behinderungsplätze“ (inhibitors):  
Ist Vorplatz einer Transition markiert, so wird das Schalten verhindert.
- Priorität von Transitionen (kleinste Zahl = höchste Priorität)
- „Erlaubnis-Funktion“ (enabling function): logische Funktion, def. auf Menge der Markierungen; Transition darf nicht schalten bei Wert F.
- Transition hat die Form (name, priorität, funktion).

- **Modellierung der Bildung eines Quorums**

- Plätze: START, DRIVE, DONE
- CC,..., OW: current copy,..., outdated witness
- QC, QW: quorum of copies, quorum of witnesses
- log. Funktion  $q$ : Mehrheit erreicht?

- **PETRI-Netz: Bildung des Quorums**



- **Modellierung des Gesamtnetzes**

einschl. Behebung von Fehlern und manueller Rekonstruktion.

weitere Plätze: DC, DW: down copy, down witness.

- **Ausgewählte Ergebnisse**

– Analyse mittels eines Programmpakets CSPL.

Gestattet Lösung des aus den MARKOV-Ketten resultierenden Gleichungssystems. Damit Bestimmung der Zustandswahrscheinlichkeiten und weiter der Verfügbarkeit des Systems möglich.

– Untersucht wurden die statische und die dynamische Bildung eines Quorums im Blick auf

Anzahl der Kopien – Anzahl der Zeugen – mittlere Reparaturzeit.

– *Ergebnisse:*

\* Verfügbarkeit sinkt, wenn Kopien durch Zeugen ersetzt werden, bis zu einem gewissen Punkt, danach steigt sie wieder.

Beispiel.  $r = 7$

Kopien	Zeugen	Verfügbarkeit
7	0	1,000
6	1	1,000
5	2	0,999
4	3	0,957
3	4	0,966
2	5	0,972
1	6	0,975

\* Hinzufügen von Zeugen verringert Verfügbarkeit.

\* In den meisten Fällen führt dynamische Votierung zu höherer Verfügbarkeit.