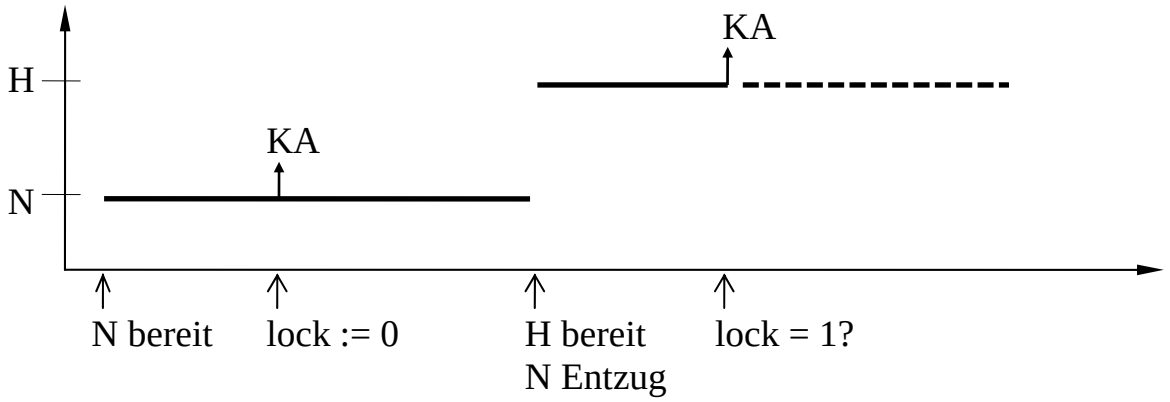
9. Konkurrierender Betriebsmittelzugriff

9.1. Probleme

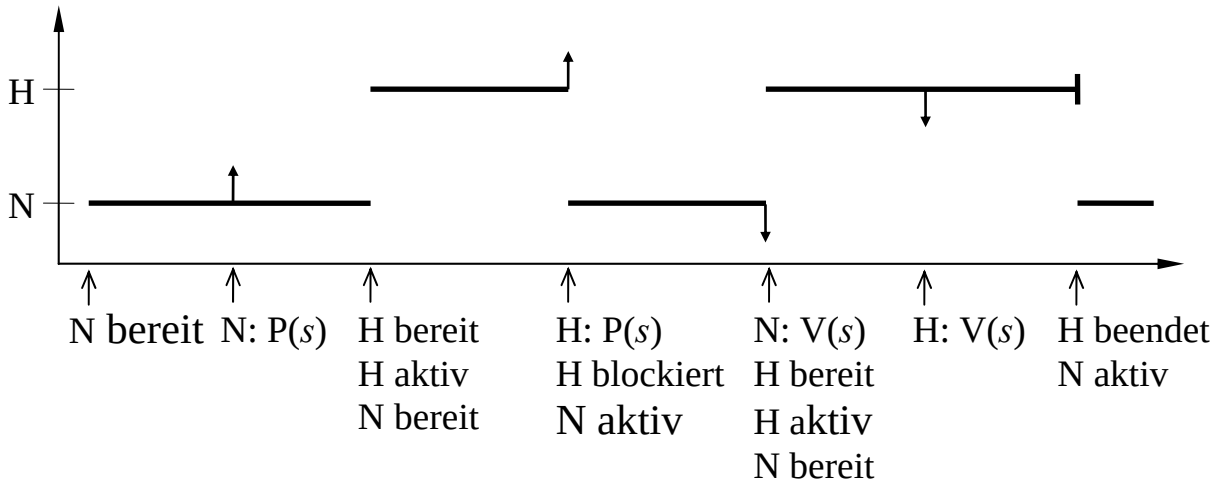
- **Prioritätsumkehr (priority inversion)**

  Annahme: Tasks besitzen Deadlines (sind aber nicht notwendig periodisch), und es bestehen Abhängigkeiten (kritische Abschnitte, KA). Prozessor ist entziehbar durch Tasks höherer Priorität.

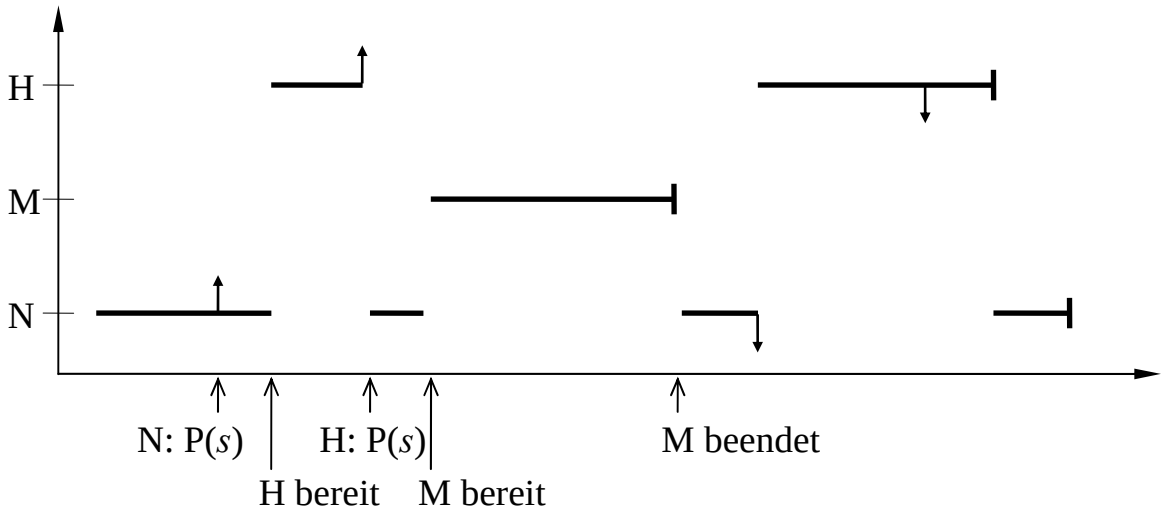
– ***Prioritätsumkehr bei aktivem Warten***



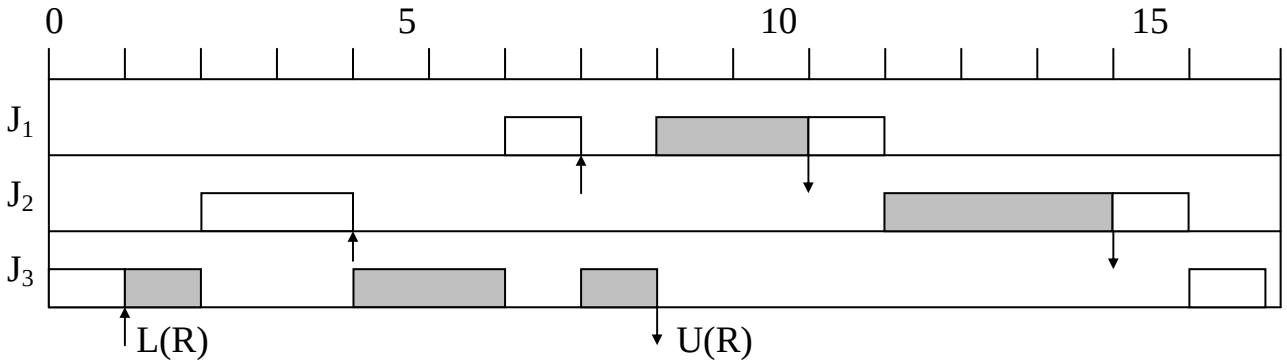
– ***Prioritätsinversion bei Verwendung von Semaphoren***



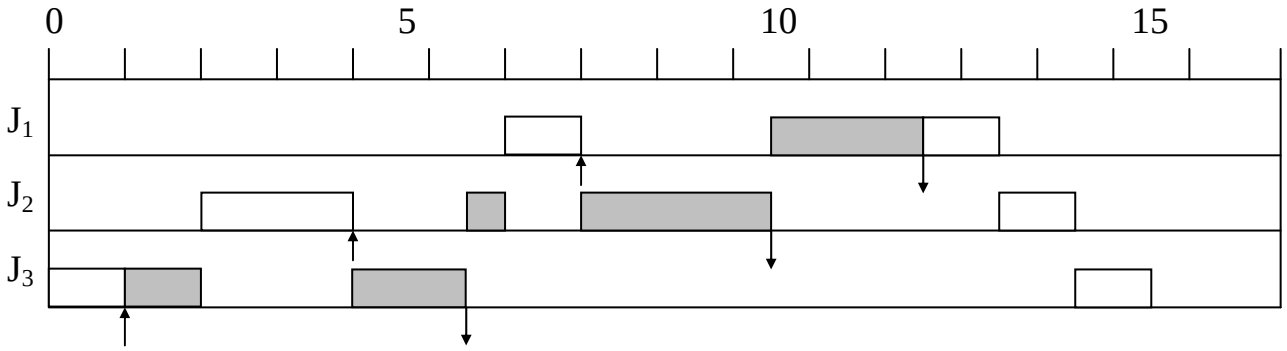
M: Task mittlerer Priorität (ohne Verwendung von *s*)



- **Zeitanomalie**



Verkürzung der BM-Nutzungszeit von J_3 auf 2,5:



- **Verklemmungen**

9.2. Annahmen und Bezeichnungen

– 1 Prozessor, entziehbar, keine Selbstunterbrechung
  Scheduling prioritätsbasiert
  Tasks nicht notwendig periodisch.

– $\boldsymbol{R_1,...,R_r}$ Betriebsmittel (BM); nicht entziehbar, exklusiv

– $\mathbf{L}(\boldsymbol{R_k})$, $\mathbf{U}(\boldsymbol{R_k})$ Anfordern/Freigeben von R_k; Freigabe nach LIFO
  $\uparrow\boldsymbol{R_k}$ $\downarrow\boldsymbol{R_k}$

– $\boldsymbol{J_1,...,J_n}$ Jobs;
  $\boldsymbol{J_h}$, $\boldsymbol{J_l}$ Job hoher/niedriger Priorität

  $\boldsymbol{p_1,\ldots,p_n}$ Prioritäten (höchste Priorität: 1), „fest“ zugeordnet
  o.B.d.A. J_i geordnet gemäß Prioritäten

  $\boldsymbol{p_i(t)}$ aktuelle Priorität von J_i

– ***Jobs in Konflikt***
  benötigen dasselbe BM

  Jobs konkurrieren um ein BM
  einer der Jobs hat das BM, das ein anderer Job momentan braucht

  Job blockiert
  kann angefordertes BM nicht erhalten

– ***Prioritätsumkehr***
  J_l wird ausgeführt, während J_h blockiert ist

9.3. Prioritätsvererbung

für präemptives prioritätsbasiertes Scheduling SHA et al., 1990

- **Basic priority-inheritance protocol**

(1) ***Scheduling-Regel***

J_i erhält Prozessor gemäß aktueller Priorität $p_i(t)$;

Freigabezeit: $p_i(t) := p_i$.

(2) ***Zuteilungsregel***

J_i fordere R_k zur Zeit t.

(a) R_k belegt: Anforderung verweigert, J_i blockiert.

(b) R_k frei: R_k wird J_i zugeteilt.

(3) ***Prioritätsvererbungs-Regel***

Wird J_i blockiert durch J_l, so erbt J_l die aktuelle Priorität von J_i, d.h.

$p_l(t) := p_i(t)$.

J_l wird mit dieser Priorität bearbeitet.

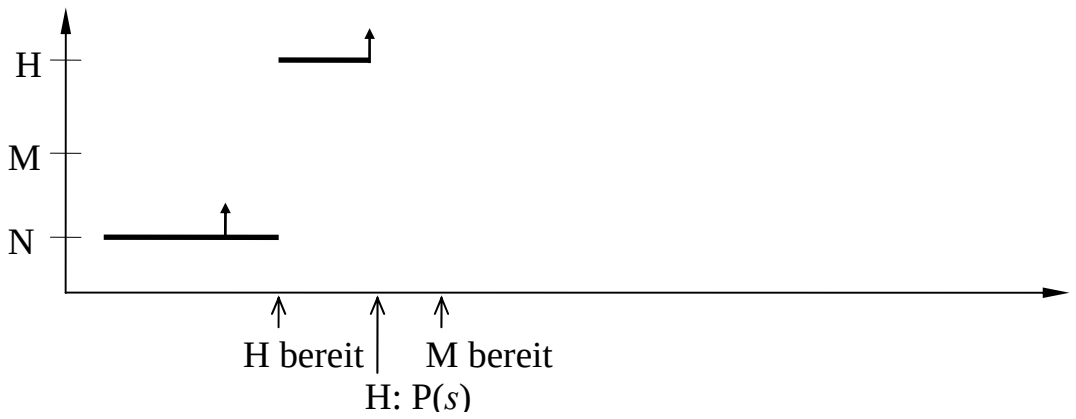
Zur Freigabezeit $\tilde{t}$ von R_k wird Priorität zurückgesetzt:

$p_l(\tilde{t}) := p_l(t')$ t': Zuteilungszeitpunkt von R_k an J_l.

- **Beispiel 9.1.**

2 Tasks: keine Auswirkung!

3 Tasks:

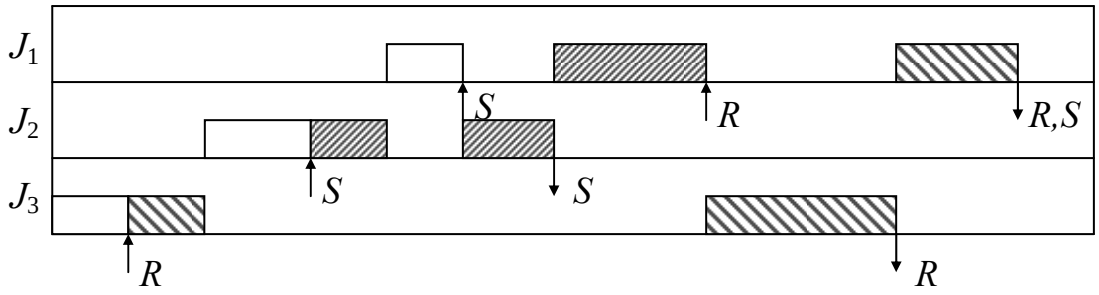


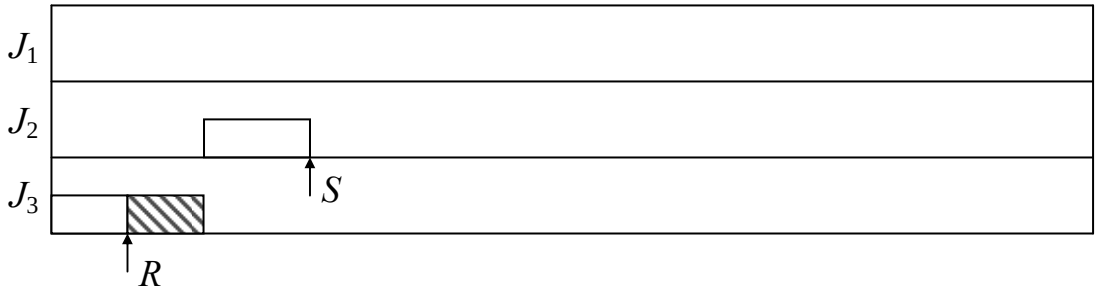
- **Eigenschaften**

– Prioritätsvererbung ist transitiv.

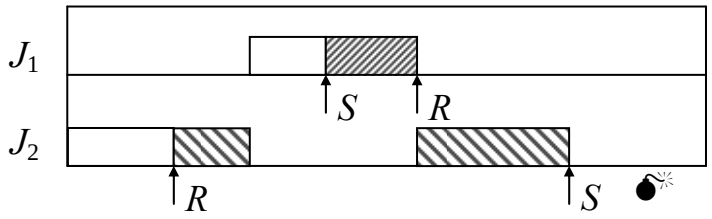
– Keine unbegrenzte unkontrollierte Prioritätsumkehr.

– Prioritätsvererbung führt nicht notwendig zu minimaler Blockierungszeit.





– Prioritätsvererbung verhindert keine Deadlocks.



9.4. Prioritätsschranken
Basic Priority-Ceiling Protocol

SHA/RAJKUMAR/LEHOCZKY, 1990

- **Voraussetzungen und Begriffe**

– 1 Prozessor, präemptives prioritätsbasiertes Scheduling, keine Selbstunterbrechung.

– Zugeordnete Prioritäten p_i sind fest.

Prioritäten: natürliche Zahlen, 1 höchste Prior.; Ω niedrigste Priorität.

– BM-Anforderungen aller Jobs a priori bekannt.

– ***P(R_k) Prioritätsschranke (priority ceiling) von R_k***

höchste Priorität aller Jobs, die R_k anfordern.

– $\hat{P}(t)$ ***Prioritätsschranke des Systems zur Zeit t***

höchste Prioritätsschranke aller derjenigen BM, die zur Zeit t belegt sind.

- **Prioritätsschranken-Protokoll**

  (1) ***Scheduling-Regel***

  Zur Freigabezeit t^{rel} von J_i ist $p_i(t^{rel}) = p_i$.

  (2) ***Zuteilungsregel***

  J_i fordere R_k zur Zeit t.

  (a) R_k belegt: Anforderung verweigert, J_i blockiert („an R_k“).

  (b) R_k frei:

  (α) $p_i(t) \succ \hat{P}(t)$: R_k wird J_i zugeteilt.

  (β) sonst: R_k wird J_i genau dann zugeteilt, wenn J_i derjenige Job ist, der die BM R besitzt mit $P(R) = \hat{P}(t)$.

  Andernfalls Verweigerung und Blockierung.

  (3) ***Prioritätsvererbungsregel***

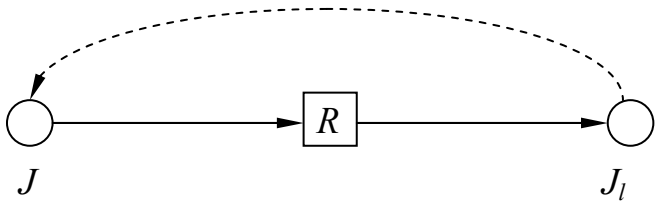
  Wird J_i durch J_l blockiert, so erbt J_l die aktuelle Priorität $p_i(t)$ von J_i. J_l wird mit dieser Priorität (preemptiv) bearbeitet bis zur Freigabe aller BM, deren Prioritätsschranke mindestens $p_i(t)$ ist. Danach fällt J_l auf die Priorität $p_l(t')$ zurück (t': Zuteilungszeitpunkt von R_k an J_l).

- **Beispiel 9.2.**

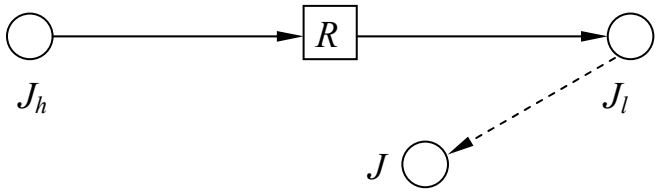
- **Eigenschaften**

– Unterschied zu Prioritätsvererbung: *drei* Blockierungsformen

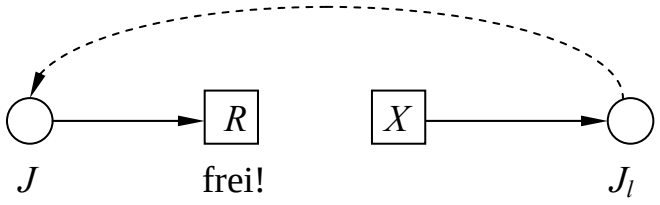
direkt:



durch Vererbung:



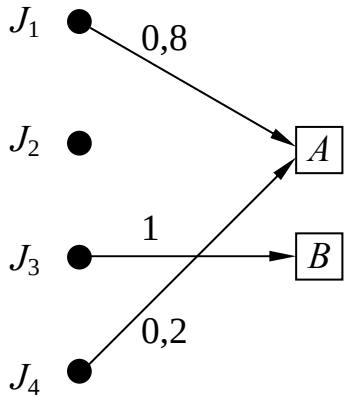
durch Prioritätsschranken:



– Deadlocks sind ausgeschlossen.

– Ein Job kann höchstens für die Dauer *eines* BM-Zugriffs blockiert sein.

Berechnung der Blockierungsdauer – Beispiel:



	J_1	J_2	J_3	J_4

– Blockieren ist „anti-transitiv“ (keine mehrfache Prioritätsinversion).

9.5. Kellerbasiertes Prioritätsschranken-Protokoll
Stack-Based Priority-Ceiling Protocol

- **Weitere Voraussetzungen**

– Gemeinsamer Laufzeit-Stack für alle Jobs

– Stack eines aktiven Jobs stets „ganz oben“ (Verdrängung!)

– Stack wird bei Ende eines Jobs freigegeben

- **Protokoll**

(1) $\hat{P}(t) = \Omega$, falls alle BM R_k frei.

Aktualisierung von $\hat{P}(t)$ bei jeder Belegung oder Freigabe von R_k.

2) ***Scheduling-Regel***

Nach seiner Freigabe ist ein Job J_i blockiert, bis $p_i(t) \succ \hat{P}(t)$ gilt.

Bereite Jobs werden gemäß ihrer zugeordneten Priorität ausgeführt.

(3) ***Zuteilungsregel***

Wenn ein Job ein BM anfordert, wird es ihm zugeteilt.

- **Beispiel.**

- **Eigenschaften**

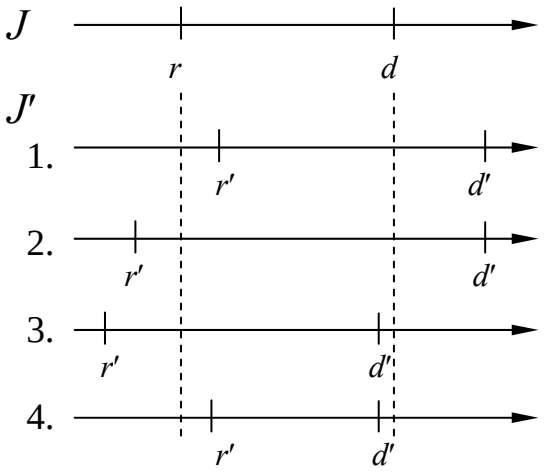
– Bei Beginn der Ausführung eines Jobs sind alle BM, die der Job anfordert, frei.

– Beide Protokolle führen zu derselben maximalen Blockierungszeit eines Jobs.

– Deadlocks und mehrfache Prioritätsinversion sind ausgeschlossen.

9.6. Verdrängungsstufen (preemption levels)

- **Eigenschaften von EDF**



– Werden periodische Tasks mittels EDF ausgewählt, so sind die Prioritäten auf *Jobebene* statisch.

– Ein Job J mit kürzerer relativer Deadline kann nie durch einen Job J' mit längerer relativer Deadline verdrängt werden.

$\rightarrow J_l$ kann J_h nur dann blockieren, wenn J_h Job J_l verdrängen kann.

- **Verdrängungstufe π_J eines Jobs J**

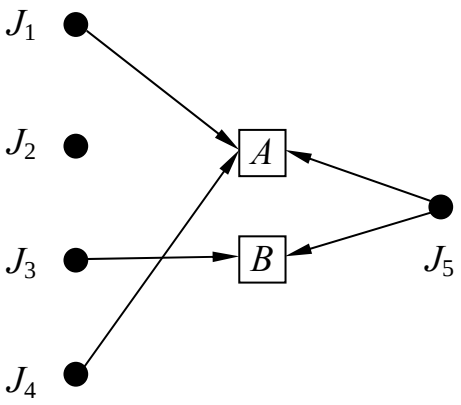
– ***Eigenschaft***

Ein Job höherer Verdrängungsstufe kann nicht durch einen Job niedrigerer Verdrängungsstufe unterbrochen werden.

– ***Bedingung für Zuordnung (validation condition)***

Hat J eine höhere Priorität als J', aber eine spätere Freigabezeit, so muß J eine höhere Verdrängungsstufe als J' erhalten.

– ***Beispiel 9.3.***



Jobs	p_i	r_i	π_i	π_i'
J_1	1	5		
J_2	2	7		
J_3	3	4		
J_4	4	0		
J_5	5	3		

- **EDF-basiertes Scheduling für periodische Tasks**

  „Deadline-monoton“ ist gültige Zuordnung von Verdrängungsstufen zu Jobs: je kürzer die relative Deadline, desto höher die Verdrängungstufe.

  → Alle Jobs einer periodischen Task haben dieselbe Verdrängungsstufe („fixed preemption-level system“).

- **Verdrängungsschranken (Preemption Ceilings)**

  $\Pi(R)$ Verdrängungsschranke von BM R

  $\hat{\Pi}(t)$ Verdrängungsschranke des Systems zum Zeitpunkt t

  analog Prioritätsschranken

- **Kellerbasiertes Verdrängungsstufen-Protokoll (Stack-Based, Preemption-Ceiling Protocol)**

  (0) $\hat{\Pi}(t) = \Omega$, wenn alle BM R frei.

  $\hat{\Pi}(t)$ wird aktualisiert bei jeder BM-Zuteilung oder -Freigabe.

  (1) ***Scheduling-Regel***

  Nach seiner Freigabe ist ein Job J blockiert, bis gilt:

  $\pi_J \succ \hat{\Pi}(t)$ und $\pi_J \succ \pi_{J'}$ (J': aktueller Job).

  Bereite Jobs werden gemäß ihrer zugeordneten Priorität ausgeführt.

  (2) ***Zuteilungsregel***

  Wenn ein Job ein BM anfordert, wird es ihm zugeteilt.

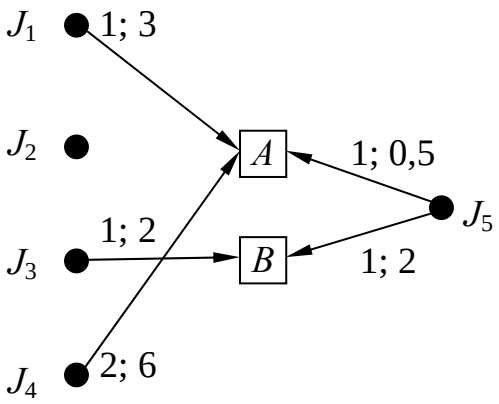
  (3) ***Prioritätsvererbungsregel***

  Blockiert Job J einen anderen Job bei dessen Freigabe, so erbt J die höchste Prioriät aller blockierten Jobs.

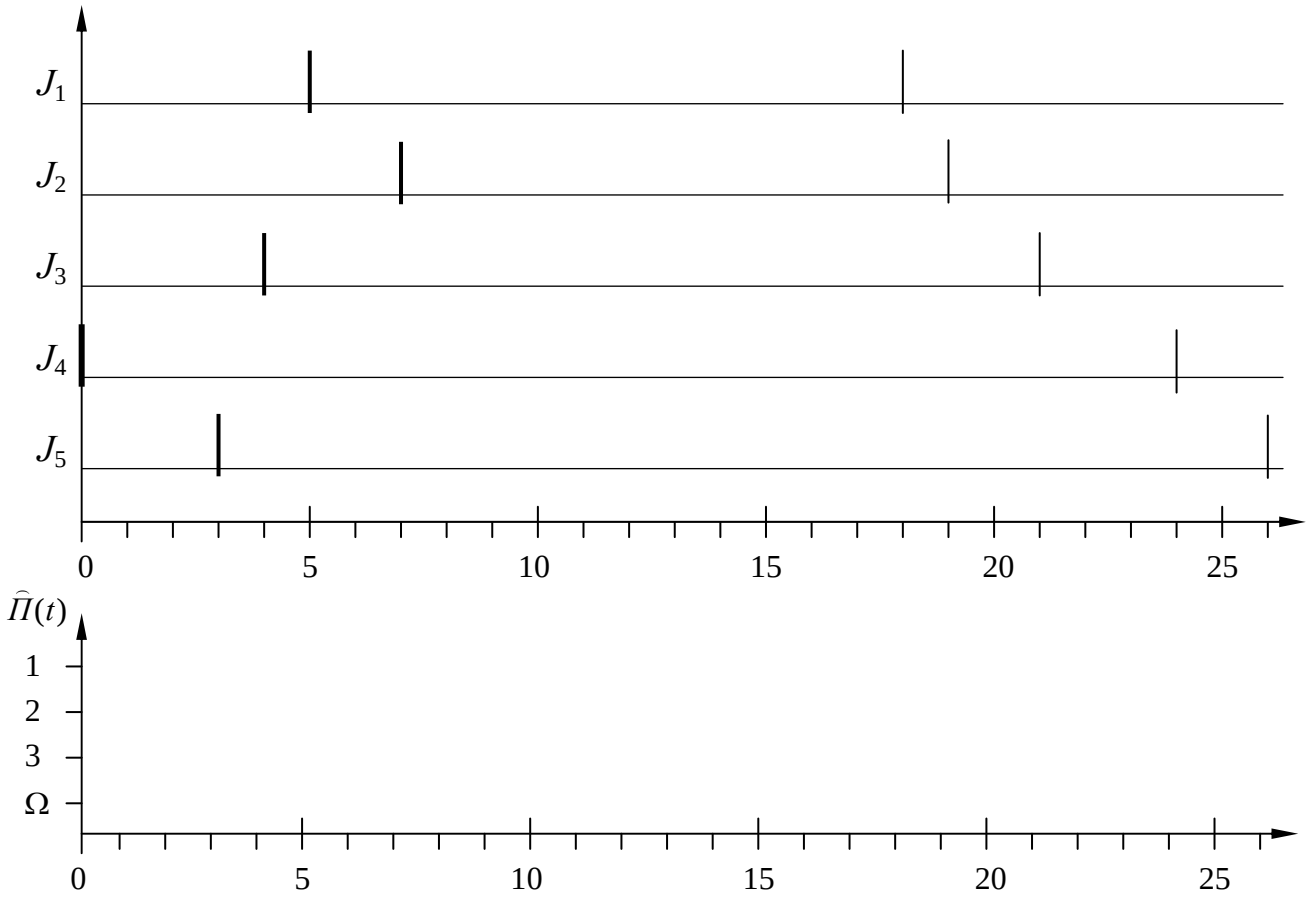
- **Beispiel 9.4.**

  Kellerbasiertes Verdrängungsstufen-Protokoll für Bsp. 9.3 bei EDF.

  Kantenbeschriftung: Beginn und Dauer der BM-Nutzung



Jobs	p_i	r_i	d_i	e_i	π_i
J_1		5	13	5	
J_2		7	12	3	
J_3		4	17	4	
J_4		0	24	9	
J_5		3	23	4	



- **Eigenschaften**

  wie Prioritätsschranken-Protokoll

9.7. Mehrexemplar-Betriebsmittel

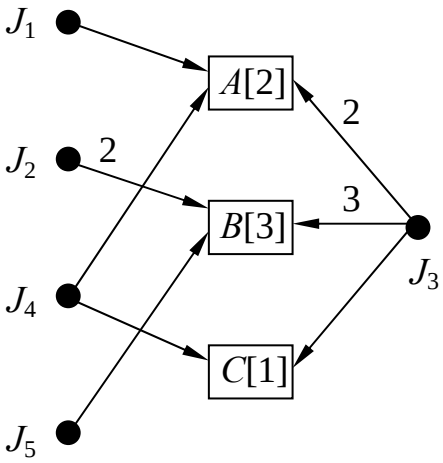
r_k: Anzahl der Exemplare von BM R_k

f: Anzahl der freien Exemplare eines BM

- **Prioritätsschranke $P_f(R_k)$**

  höchste Priorität aller Jobs, die (momentan) mehr als f Exemplare anfordern; sonst Ω.

- **Beispiel 9.5.**



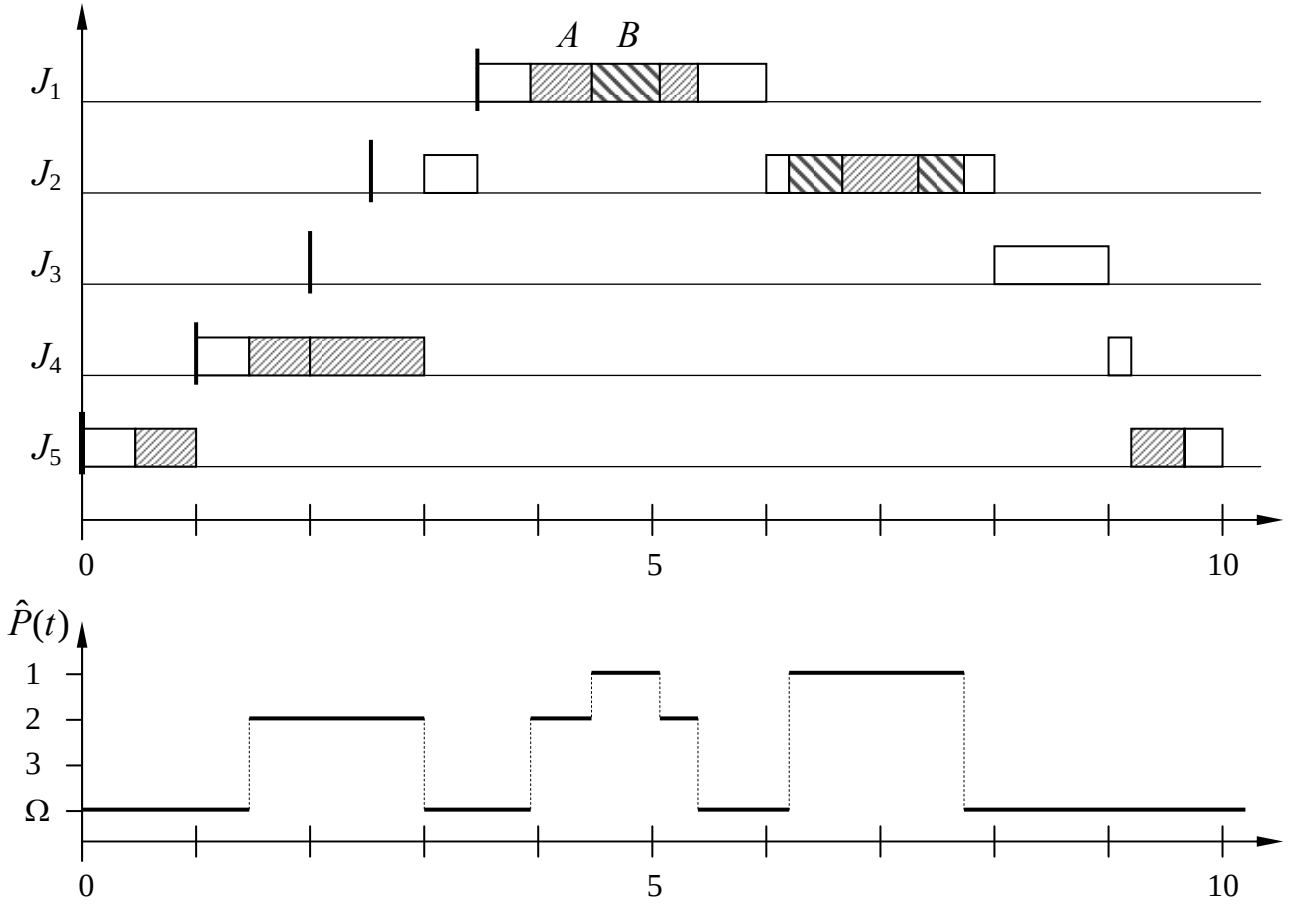
	BM-Anford. von J_i					$P_f(R)$			
$i \mid f$	1	2	3	4	5	0	1	2	3
A									
B									
C									

- **Protokoll und Eigenschaften**

  analog (bis auf Vererbung bei Prioritätsschranken-Protokoll).

- **Beispiel 9.6.**

	BM-Anford. von J_i					$P_f(R)$					
$i \mid f$	1	2	3	4	5	0	1	2	3	4	5
$A[5]$	2	4	0	1	1						
$B[1]$	1	1	0	0	1						



Aufgabe 9.1.

5 Jobs J_i; $p_i = i$, r_i Freigabezeitpunkt, e_i Bearbeitungszeit

2 BM A, B; a_i Beginn der BM-Nutzung (relativ zu r_i)

b_i Nutzungsdauer

i	r_i	e_i	A: a_i	b_i	B: a_i	b_i
1	7	3	1	1	-	-
2	5	3	-	-	1	1
3	4	2	-	-	-	-
4	2	6	1	4	3	1,5
5	0	6	-	-	1	4

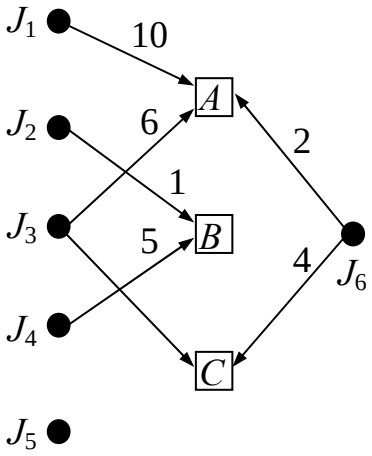
Gesucht: Ablaufpläne und Blockierungszeiten bei Einplanung

a) ohne Prioritätsvererbung
b) mit Prioritätsvererbung
c) gemäß Prioritätsschranken-Protokoll
d) gemäß kellerbasiertem Prioritätsschranken-Protokoll

sowie Verlauf von $\hat{P}(t)$ bei c) und d).

- **Aufgabe 9.2.**

  Bestimmen Sie die Blockierungszeiten (unterschieden nach der Ursache der Blockierung) für das nachstehend angegebene Jobnetz!



	direkt					Pr.-Vererbung					Pr.-Schranken				
	J_2	J_3	J_4	J_5	J_6	J_2	J_3	J_4	J_5	J_6	J_2	J_3	J_4	J_5	J_6
J_1															
J_2															
J_3															
J_4															
J_5															