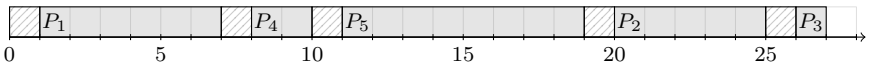
Verwendung von `blockgraph`

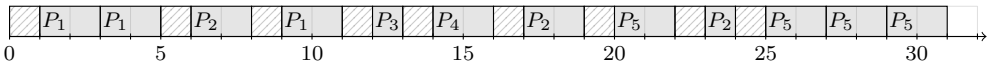
Aufgabe 1a

Scheduling-Strategie: Beste Bediengüte



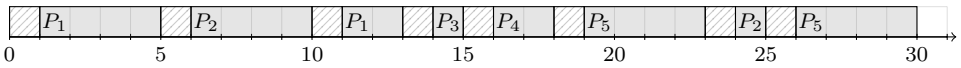
Aufgabe 1b

Scheduling-Strategie: Round Robin $\Delta t = 2$



Aufgabe 1c

Scheduling-Strategie: Round Robin $\Delta t = 4$



Aufgabe 1d

Antwortzeiten der Prozesse

Aufgabe	P_1	P_2	P_3	P_4	P_5	Ø
1a	6	20	21	3	10	$\frac{60}{5} = 12$
1b	10	19	7	9	22	$\frac{67}{5} = 13,4$
1c	12	20	9	11	21	$\frac{73}{5} = 14,6$

Aufgabe 2a

Annahme: Wenn ein idealer Scheduler alle Deadlines in unbeschränkter Zeit einhalten soll, muss ein Zeitintervall (I) existieren, für das gilt:
a) Es ist (ganzzahlig) durch die Periodendauern ($Pd(A_i)$) teilbar.
b) Die Summe der Bedienzeiten ($B(A_i)$) im Zeitintervall ist kleiner, als die Länge des Intervalles.

Da $B(A_i)$ durch die Anzahl der Perioden im Intervall ($PI(A_i)$) in Abhängigkeit von $|I|$ steht, ist das kleinste gemeinsame Vielfache der Periodenlängen praktikabel. Daher: $|I| = kgV(4, 7, 3) = 84$

Für $PI(A_i)$ gilt:
$PI(A_i) = |I| : Pd(A_i)$

$Task$	$PI(A_i)$	$B(A_i)$
A_1	21	21
A_2	12	36
A_3	24	24

$$\Rightarrow \quad \sum_{i=1}^{3} B(A_i) = 85 > 84 = |I|$$

Somit ist gezeigt, dass es selbst für einen perfekten Scheduler unmöglich ist, alle Zei

Wenn die Deadlines aller Aufträge auf unbeschränkte Zeit eingehalten werden könnten, so muss es ein Zeit-Intervall geben, das (1) genau so lang ist, dass es ganzzahlig durch die Periodendauer aller Aufträge teilbar ist und (2) genug Platz für die Summe der benötigten Bedienzeiten der Aufträge im Intervall bietet.

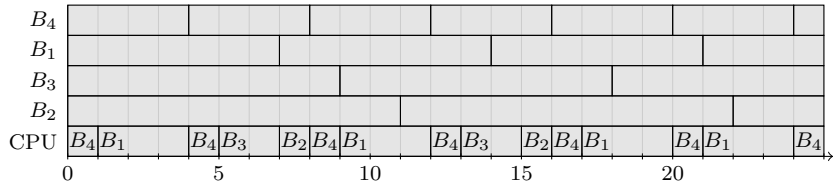
Länge des Intervalls: $kgV(4; 7; 3) = 4 * 7 * 3 = 84$

Aufgabe	Anz. Perioden		Benötigte Bedienzeit
A_1 :	$84 : 4 = 21$	$\rightarrow$	$28 * 1 = 28$
A_2 :	$84 : 7 = 12$	$\rightarrow$	$21 * 1 = 21$
A_3 :	$84 : 3 = 28$	$\rightarrow$	$12 * 3 = 36$

Gesamte benötigte Bedienzeit: $28 + 21 + 36 = 85$

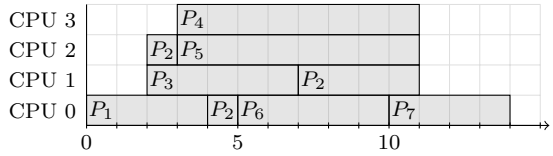
Da insgesamt mehr Bedienzeit benötigt wird als im Intervall verfügbar ist, können die Deadlines selbst von einem idealen Scheduler nicht eingehalten werden.

Aufgabe 2b i



Die ersten vier Zeilen visualisieren die Periodendauer der einzelnen Aufträge.

Aufgabe 2c



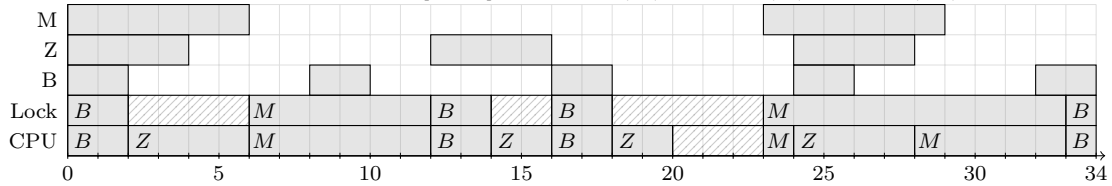
Da die Aufgabenstellung etwas schwammig formuliert war, haben wir angenommen, dass eine kleinere Zahl in der Priorität-Zeile eine höhere Priorität bedeutet, so hat zum Beispiel P_3 eine höhere Priorität als P_2.

Aufgabe 3a

Aus Gründen der Darstellung sind sämtliche Werte durch 5 dividiert. Dies führt zu folgender Tabelle:

Auftrag	M	B	Z
Periodendauer p_i	23	8	12
Bedienzeit b_i	6	2	4

und der Darstellung im Intervall $[0, 34]$ mit $Prio(B) > Prio(Z) > Prio(M)$.



Zeitpunkt 12: B wird verzögert ausgeführt, da in $[6, 12]$ M die Semaphore verwendet.
Zeitpunkt 16: Z wird von B verdrängt.
Zeitpunkt 24: M wird von Z verdrängt, Semaphore weiterhin gesperrt.
Die letzte Verdrängung von M durch Z führt dazu, dass in $[24, 32]$ Auftrag B seine Deadline nicht einhalten kann.

Aufgabe 3b

Als Prioritätsinversion wird die unbeabsichtigte Umkehrung der Prioritäten bezeichnet: Auftrag B konnte Z nicht verdrängen, da die von B benötigte Semaphore noch von M verwendet wurde, sodass Z für eine kurze Zeit eine höhere Priorität als B hatte.