Realzeit Nachweis

Josef Mueller, Isabella Schoen
Gruppe1

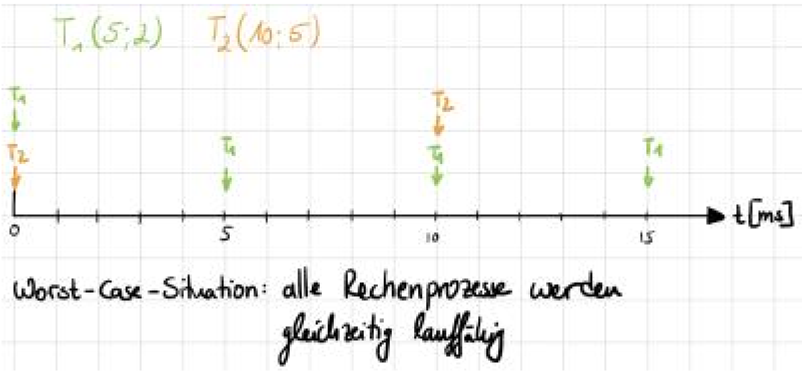
Deadline Scheduling

Beschreibung

Gegeben sind die beiden unterbrechbaren Tasks T1 (5;2) und T2 (10;5) in einem Einprozessorsystem, welches nach dem EDF Verfahren arbeitet. Aufgaben

1. Definieren Sie die Worst Case Situation der Anforderungen als Anforderungsfunktion.

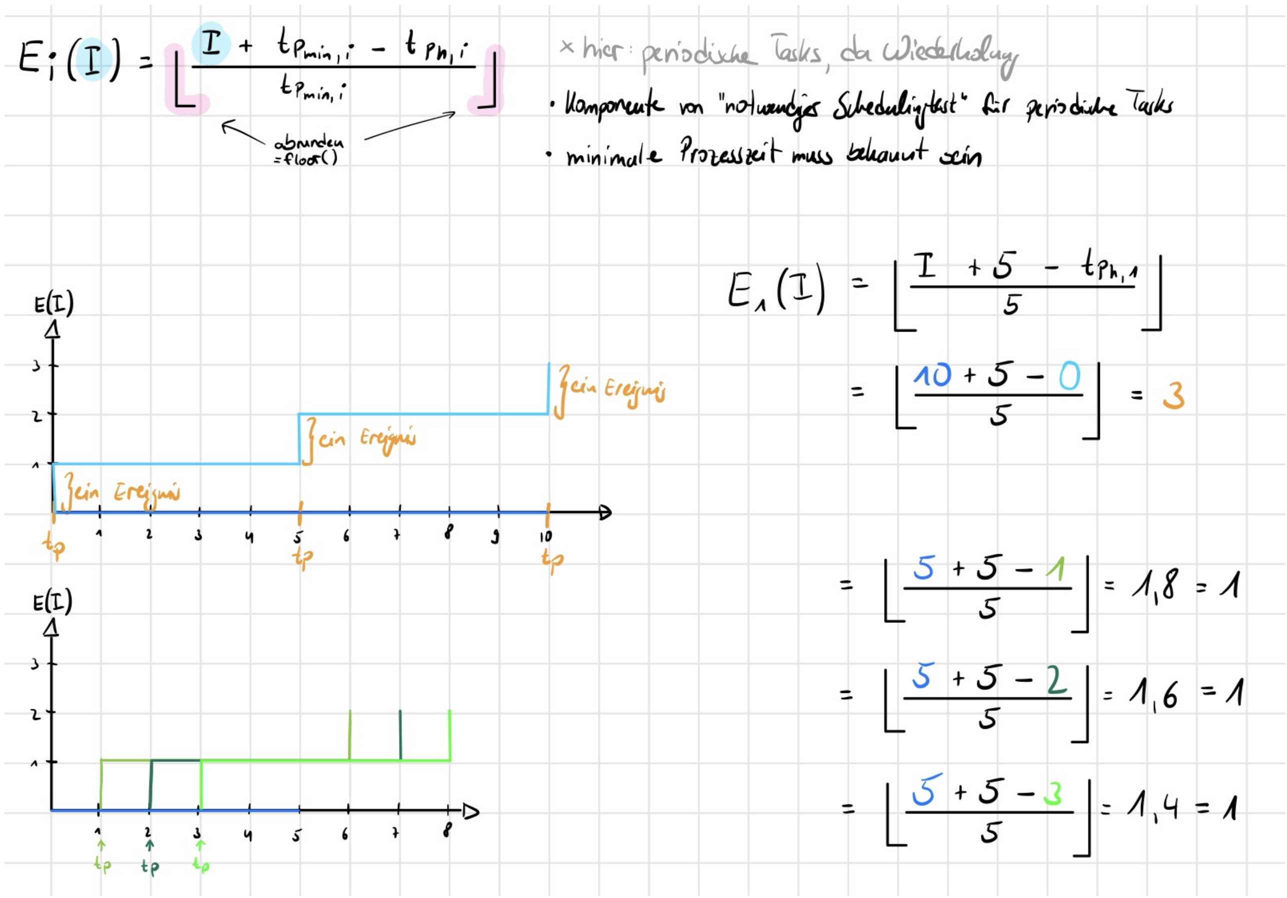
> A: Beim EDF-Scheduling ist das Worst Case Szenario, dass alle Tasks gleichzeitig lauffähig werden.



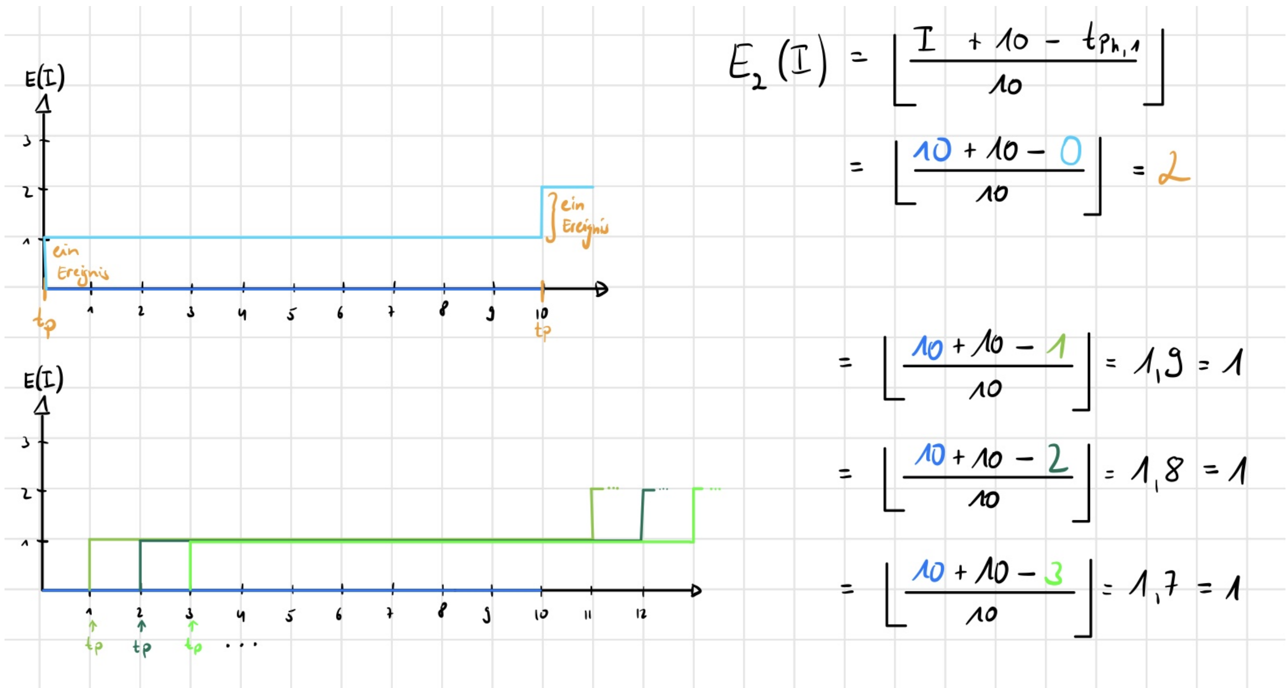
2. Stellen Sie jeweils für T1 und T2 die Ereignisdichtefunktion E(I) grafisch dar.

> A: Bei dieser Aufgabe wurden beide Tasks sowohl grafisch dargestellt und auch als Rechnung aufgestellt. Um den Einfluss der Phase darzustellen, wurde ebenso eine kleine Darstellung in der ersten Grafik beigefügt.

Task 1



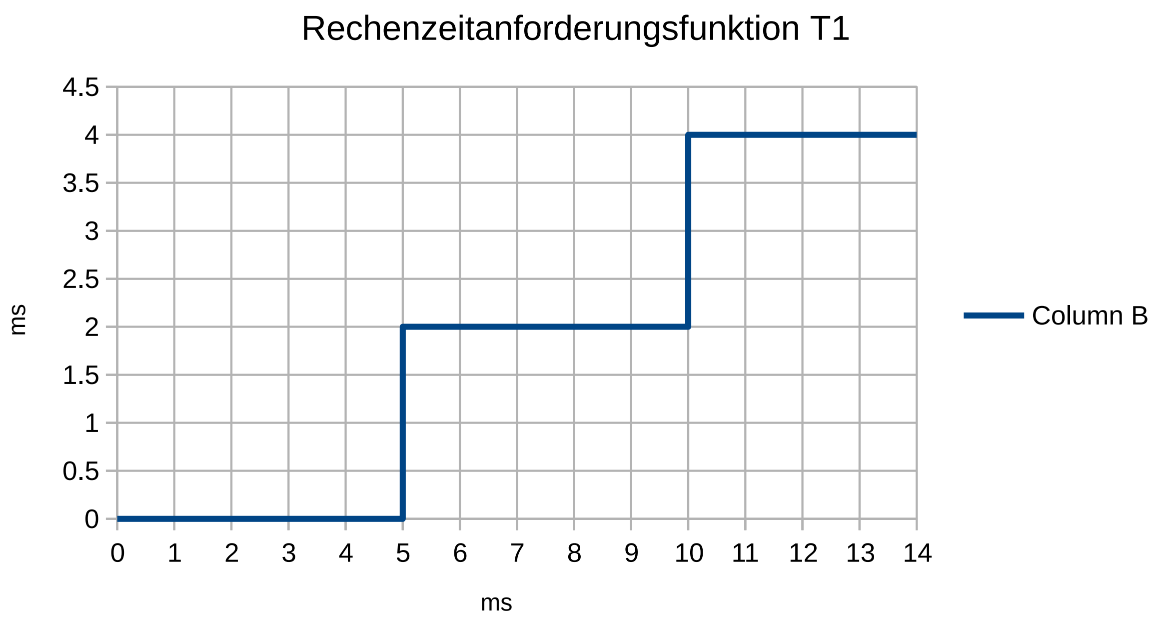
Task 2



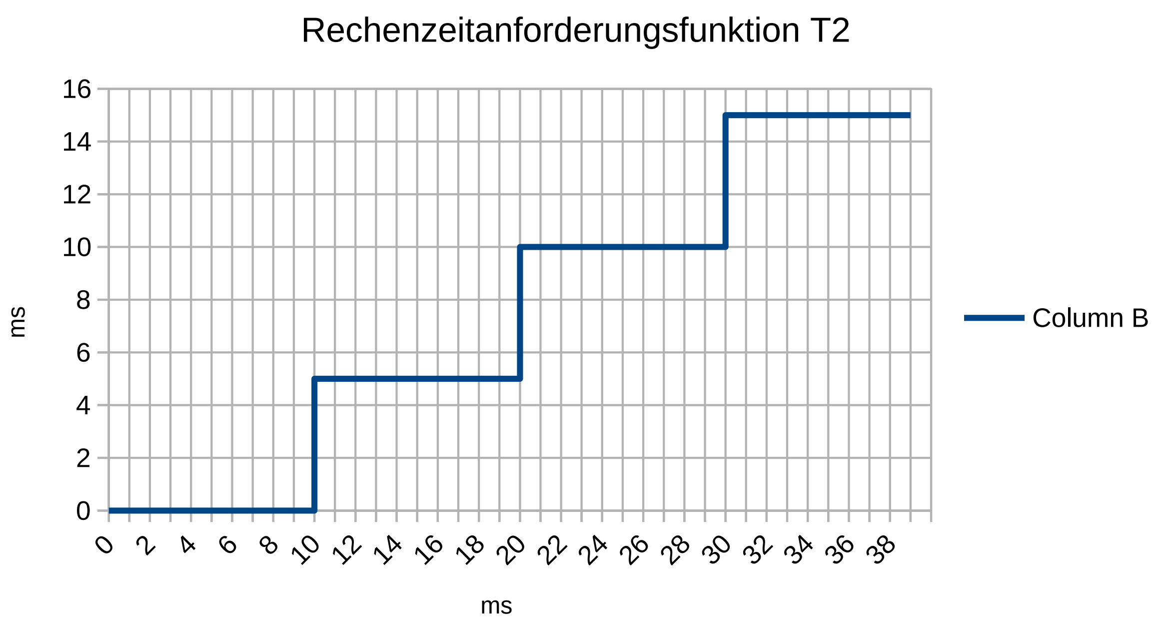
3. Skizzieren Sie die Rechenzeitanforderungsfunktion tC,1(I) und tC,2(I).

> A: Für die bessere Darstellung der Stufenfunktionen der einzelnen Tasks, wurden größere Intervalle gewählt.

Task 1

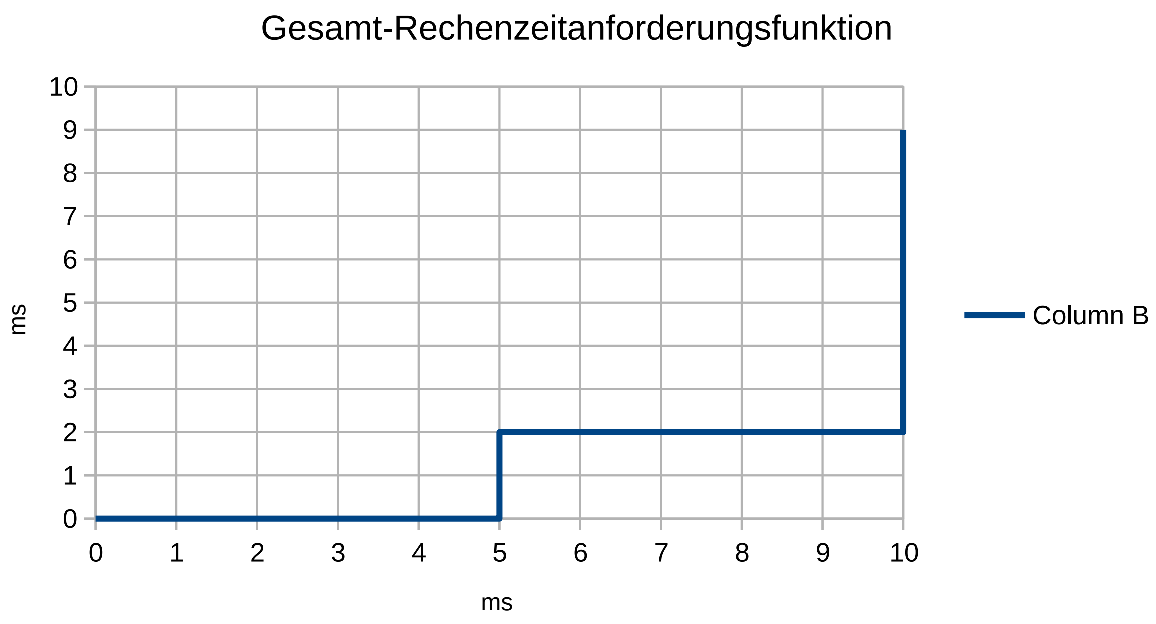


Task 2



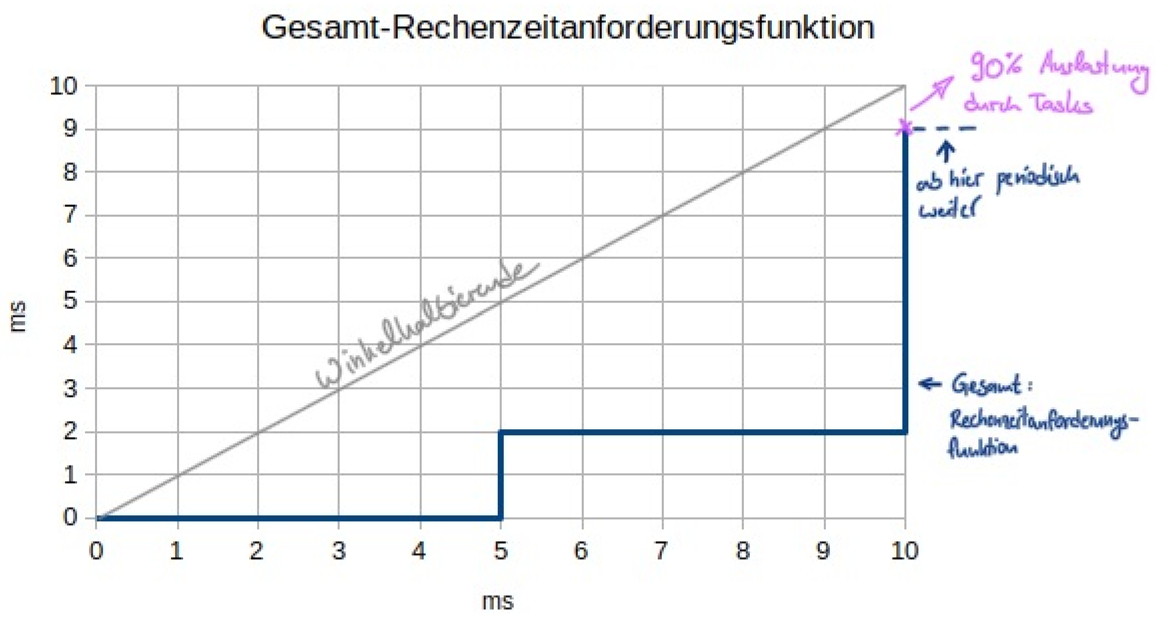
4. Skizzieren Sie die Gesamt-Rechenzeitanforderungsfunktion tC,ges(I).

> A: Die Hyperperiode beträgt hier 10 [ms], wodurch sich nach dieser Zeit alle Tasks aus dem Taskset wiederholen.



5. Lesen Sie aus der Skizze die Auslastung des Rechners durch die Abarbeitung der beiden Tasks ab.

> A: Anhand der Skizze, im Intervall [0;10] (10 = Hyperperiode), wurde für die Auslastung durch die zwei Tasks folgende Auslastung ausgelesen: 90%. Das lässt sich anhand des Abstands zwischen der Winkelhalbierenden, welche die maximale Auslastung angibt, und der Task-Auslastung ablesen.
> Genauer: Der Abstand gibt an, wie viel *Rest-Ressourcen* noch vorhanden sind. (so unser Verständnis)



6. Geben Sie die Ereignisdichtefunktion E1(I) und E2(I) formelmäßig an.

> A: Die Funktion wurde bereits in der Aufgabe 2 aufgestellt. Die Ereignisdichtefunktion gibt an, wie oft eine Rechenzeitanforderung in einem Intervall **I** stattfindet. Die Phase beschreibt die zeitliche Verschiebung einer Anforderung in Form einer Rechtsverschiebung.

> Runtergebrochen gesagt rechnet man Folgendes aus: Wie viele Rechenzeitanforderungen passen in ein Intervall.

7. Stellen Sie die Rechenzeitanforderungsfunktion tC,ges(I) formelmäßig dar und weisen Sie die 'schritthaltende Verarbeitung' nach.

> A: Das kleinste gemeinsame Vielfache der Prozesszeiten der beiden Tasks beträgt 10. Die Bedingung für die Bestimmung der Intervalle ist Folgende: 0 < I < kgV(5, 10). Demnach sind die einzigen Intervalle I: {5}. Es macht jedoch Sinn auch die 10 in die Intervalle zu beziehen, also I: {5, 10}.

$T_1(5; 2) \qquad T_2(10; 5)$

$$t_{C,ges}(I) = \sum \left\lfloor \frac{I + t_{P_{min,i}} - t_{D_{max,i}} - t_{Ph,i}}{t_{P_{min,i}}} \right\rfloor \cdot t_{E_{max,i}}$$

$$= \left\lfloor \frac{I + 5\cancel{ms} - 5\cancel{ms} - 0ms}{5ms} \right\rfloor \cdot 2ms + \left\lfloor \frac{I + 10\cancel{ms} - 10\cancel{ms} - 0ms}{10ms} \right\rfloor \cdot 5ms$$

$$= \left\lfloor \frac{I}{5ms} \right\rfloor \cdot 2ms + \left\lfloor \frac{I}{10ms} \right\rfloor 5ms$$

Hyperperiode: $0 < I < kgV(5, 10) = 10\,ms$

$I_{T1}: \{5, 10\}, \quad I_{T2}: \{5\} \Rightarrow I: \{5, 10\}$

$$t_{c,ges}(5) = \left\lfloor \frac{5ms}{5ms} \right\rfloor \cdot 2ms + \left\lfloor \frac{5ms}{10ms} \right\rfloor \cdot 5ms = 2ms \leq 5 \; (\checkmark)$$

$$t_{c,ges}(10) = \left\lfloor \frac{10ms}{5ms} \right\rfloor \cdot 2ms + \left\lfloor \frac{10ms}{10ms} \right\rfloor \cdot 5ms = 9ms \leq 10 \; (\checkmark)$$

$\downarrow$

eigentlich nicht nötig, da $0 < I < 10$ und somit "10" nicht mehr im Intervall

Code

Code zur Anwendung der Gesamt-Rechenzeitanforderungsfunktion. Wenn die Rechenzeitanforderung für ein einzelnes Task simuliert werden soll, bietet es sich an die Intervalle manuell zu setzen und das `range` in der `for`-Schleife durch das maximale Intervall zu ersetzen.

```python
import numpy
import math
import csv

kGV = 10
taskset = [(5, 2), (10, 5)]

intervals = set()
for i in taskset:
    for j in range(1, int(kGV / i[0])):
        intervals.add(j * i[0])
intervals = list(intervals)
intervals.sort()

t = numpy.zeros(kGV)
remainingTasks = taskset.copy()
t_ges = 0

for i in range(kGV):
    for task in taskset:
        t_Pmin = task[0]
        t_Emax = task[1]
        if i % t_Pmin == 0 and i in intervals:
            t_ges += t_Emax
    t[i] = t_ges

rows = [[str(i), str(value)] for (i, value) in enumerate(t)]
print(rows)

with open('taskset', 'w') as f:
    write = csv.writer(f)
    write.writerows(rows)
```

Realzeitnachweis eines Steuerungssystems

Beschreibung

Ein (Einprozessor-)Steuerungssystem mit 3 Tasks ist durch folgende Daten gekennzeichnet:

Taskname	tP,min in ms	tE,min in ms	tE,max in ms	tD,min in ms	tD,max in ms	ρmax,i (Auslastung)
A	80	10	30	0	60	0.375
B	80	20	30	0	50	0.375
C	160	28	32	0	155	0.2

Die Tasks sind voll unterbrechbar.

Aufgaben

1. Berechnen Sie

   1. pmax,A (ein Job)

      Auslastung (S. 18)

      $$\rho_{max,A} = \frac{t_{Emax,A}}{t_{Pmin,A}} = \frac{30ms}{80ms} = \underline{\underline{0{,}375}}$$

   2. tPmin,B

      $$t_{Pmin,B} \Rightarrow \frac{t_{Emax,B}}{t_{Pmin,B}} = \rho_{max,B} \quad | \div \rho_{max,B} \,| * t_{Pmin,B}$$

      $$\Leftrightarrow \frac{t_{Emax,B}}{\rho_{max,B}} = t_{Pmin,B}$$

      $$\Leftrightarrow \frac{30ms}{0{,}375} = \underline{\underline{80\ ms}}$$

   3. pmax (Gesamtauslastung für alle drei Jobs)

      $$\rho_{max,ges} = \sum \rho_{max,(A,B,C)} = 0{,}375 + 0{,}375 + \underbrace{0{,}2}_{\frac{32ms}{160ms} = \rho_{max,C}}$$

      $$= 0{,}95$$

      $$= 95\%$$ → Auslastung gesamt des Rechners durch Task A, B, C

   4. Ist eine schritthaltende Verarbeitung prinzipiell möglich?

      > A: Ja, eine schritthaltende Verarbeitung ist möglich, da die in Unteraufgabe 3 berechnte Gesamtauslastung pmax unter 1 ist.

2. Prioritätenvergabe

   1. Wie lautet die Faustformel zur Vergabe von Prioritäten?

      > A: Je kürzer die Ausführungs- **und** die Prozesszeit eines Tasks ist, desto höher ist dessen Priorität. Andere Scheduling-Verfahren haben wiederum andere Faustregeln, wie beispielsweise EDF, bei dem die Task, die am ehesten fertig sein muss, die höchste Priorität erhält.

   2. Bei welchem der drei Rechenprozesse ist eine eindeutige Vergabe der Prioritäten möglich? Welche Priorität bekommt dieser Job?

      > A: Beim Prozess C lässt sich direkt sagen, dass er die niedrigste Priorität erhält, da dieser die größte minimale Prozesszeit und die maximale Ausführungszeit aller drei

> Prozesse besitzt.

3. Realzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings für das bestehende System. Gehen Sie hierzu von den folgenden Prioritäten aus: A=1 (hoch), B=2 (mittel) und C=3 (niedrig).

    1. Geben Sie die Rechenzeitanforderungsfunktion tC,1(t) für Jobs der höchsten Priorität an.

    $$A: \; t_{c,1}(t) = \left\lceil \frac{t}{80\ ms} \right\rceil \cdot 30\ ms$$

    $$J = \{A\}$$

    2. Geben Sie die Rechenzeitanforderungsfunktion tC,2(t) für Jobs mittlerer Priorität an.

    $$B: \; t_{c,2}(t) = \left\lceil \frac{t}{80\ ms} \right\rceil \cdot 30ms + \left\lceil \frac{t}{80ms} \right\rceil \cdot 30ms$$

    $$J = \{A, B\}$$

    3. Geben Sie die Rechenzeitanforderungsfunktion tC,3(t) für Jobs der niedrigsten Priorität an.

    $$C: \; t_{c,3}(t) = \left\lceil \frac{t}{80\ ms} \right\rceil \cdot 30m + \left\lceil \frac{t}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t}{160\ ms} \right\rceil \cdot 32ms$$

    $$J = \{A, B, C\}$$

    4. Berechnen Sie die maximale Reaktionszeit tRmax,1, tRmax,2 und tRmax,3

$$A: \quad t^{(l+1)} = \sum_{j=1}^{1} \left\lceil \frac{t^{(l)}}{t_{P_{min},j}} \right\rceil \cdot t_{E_{max},j} = \left\lceil \frac{t^{(l)}}{t_{P_{min},1}} \right\rceil \cdot t_{E_{max},1} = \left\lceil \frac{t^{(l)}}{80ms} \right\rceil \cdot 30ms$$

$*_1 \; t_{E_{max},A} = 30ms$

$$t^{(l+1)} = \left\lceil \frac{t^{(0)}}{80ms} \right\rceil \cdot 30ms = \left\lceil \frac{30ms}{80ms} \right\rceil \cdot 30ms = 1 \cdot 30ms = 30ms$$

>> die $t_{R_{max},1}$ beträgt 30 ms

$$B: \quad t^{(l+1)} = \sum_{j=1}^{2} \left\lceil \frac{t^{(l)}}{t_{P_{min},j}} \right\rceil \cdot t_{E_{max},j} = \left\lceil \frac{t^{(l)}}{t_{P_{min},1}} \right\rceil \cdot t_{E_{max},1} + \left\lceil \frac{t^{(l)}}{t_{P_{min},2}} \right\rceil \cdot t_{E_{max},2}$$

$$= \left\lceil \frac{t^{(l)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(l)}}{80ms} \right\rceil \cdot 30ms$$

$*_2 \; t_{E_{max},A} = 30ms$

$t_{E_{max},B} = 30ms$

$\Rightarrow 30ms$

$$t^{(l+1)} = \left\lceil \frac{t^{(0)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(0)}}{80ms} \right\rceil \cdot 30ms = \left\lceil \frac{30ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{30ms}{80ms} \right\rceil \cdot 30ms = 60ms$$

$$t^{(l+1)} = \left\lceil \frac{t^{(1)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(1)}}{80ms} \right\rceil \cdot 30ms = \left\lceil \frac{60ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{60ms}{80ms} \right\rceil \cdot 30ms = 60ms$$

>> die $R_{max,2}$ beträgt 60 ms

$$C: \quad t^{(l+1)} = \sum_{j=1}^{3} \left\lceil \frac{t^{(l)}}{t_{P_{min},j}} \right\rceil \cdot t_{E_{max},j} = \left\lceil \frac{t^{(l)}}{t_{P_{min},1}} \right\rceil \cdot t_{E_{max},1} + \left\lceil \frac{t^{(l)}}{t_{P_{min},2}} \right\rceil \cdot t_{E_{max},2} + \left\lceil \frac{t^{(l)}}{t_{P_{min},3}} \right\rceil \cdot t_{E_{max},3}$$

$$= \left\lceil \frac{t^{(l)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(l)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(l)}}{160ms} \right\rceil \cdot 32ms$$

$*_3 \; t_{E_{max},A} = 30ms$

$t_{E_{max},B} = 30ms$

$t_{E_{max},C} = 32ms$

$\Rightarrow 32ms$

$$t^{(l+1)} = \left\lceil \frac{t^{(0)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(0)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(0)}}{160ms} \right\rceil \cdot 32ms$$

$$= \left\lceil \frac{32ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{32ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{32ms}{160ms} \right\rceil \cdot 32ms = 92ms$$

$$t^{(l+1)} = \left\lceil \frac{t^{(1)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(1)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(1)}}{160ms} \right\rceil \cdot 32ms$$

$$= \left\lceil \frac{92ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{92ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{92ms}{160ms} \right\rceil \cdot 32ms = 152ms$$

$$t^{(l+1)} = \left\lceil \frac{t^{(2)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(2)}}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{t^{(2)}}{160ms} \right\rceil \cdot 32ms$$

$$= \left\lceil \frac{152ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{152ms}{80ms} \right\rceil \cdot 30ms + \left\lceil \frac{152ms}{160ms} \right\rceil \cdot 32ms = 152ms$$

>> die $t_{R_{max},3}$ beträgt 152ms

5. Zeigen Sie anhand der 2. Realzeitbedingung (Rechtzeitigkeitsbedingung), dass schritthaltende Verarbeitung nicht möglich ist.

> A: Die zweite Realzeitbedingung besagt, dass die Reaktion t_R innerhalb der Deadline t_D stattgefunden haben muss. (S. 229, S. 222) Es gilt: `t_Dmin <= t_Rmin <= t_Rmax <= t_Dmax`.
>
> Task A: 0 <= 0 <= 30 <= 60. Passt!
> Task B: 0 <= 0 <= 60 <= 50. **Passt nicht!**
> Task C: 0 <= 0 <= 152 <= 155. Passt!

> Task B überschreitet mit seiner maximalen Reaktionszeit die maximale Deadline. Damit ist keine schritthaltende Verarbeitung mehr möglich.

4. Was können Sie unter Beibehaltung des prioritätengesteuerten Schedulings ändern, damit dennoch eine schritthaltende Verarbeitung gewährleistet wird? Geben Sie für diesen Fall die maximale Reaktionszeit tRmax,A für Job A an.

   > A: Damit eine schritthaltende Verarbeitung wieder möglich ist, sollten folgende Punkte beachtet werden:
   >
   > 1. Die Prioritäten sollten dieselben bleiben. D.h. A > B > C
   > 2. Je kleiner die Prozess- und Ausführungszeit, desto höher die Priorität (Siehe Frage 2.1). D.h. nur die Zeiten von A dürfen verändert werden, da B sonst bei kleineren Zeiten eine höhere Priorität als A erhalten würde. Die Veränderung von C spielt hier keine Rolle, da die Berechnung von t_Rmax(B) nicht von C abhängig ist.

   > Durch mehrfaches Experimentieren mit der Ausführungszeit von A wurde festgestellt, dass die größtmögliche Ausführungszeit von A 20 ms betragen sollte, damit eine schritthaltende Verarbeitung möglich ist.
   > Dadurch ergibt sich eine maximale Reaktionszeit von t_Rmax,B = 50 ms.
   > Damit ergibt sich eine maximale Reaktionszeit t_Rmax,A von 20 ms.

   Code zur iterativen Berechnung der maximalen Reaktionszeit

```python
import math

taskset = [(80, 30), (80, 30)]
t = 30

for _ in range(10):
    _t = 0
    print(t)
    for task in taskset:
        _t += math.ceil(t / task[0]) * task[1]
    t = _t
```

5. Realzeitnachweis bei Einsatz eines Deadline-Schedulings

   1. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion tC,ges(I) an.

Task A Task B Task C

$$t_{c,ges}(I) = \sum \left\lfloor \frac{I + t_{P_{min,i}} - t_{D_{max,i}} - t_{Ph,i}}{t_{P_{min,i}}} \right\rfloor \cdot t_{E_{max,i}}$$

$$= \left\lfloor \frac{I + 80ms - 60ms - 0ms}{80ms} \right\rfloor \cdot 30ms + \left\lfloor \frac{I + 80ms - 50ms - 0ms}{80ms} \right\rfloor \cdot 30ms$$

$$+ \left\lfloor \frac{I + 160ms - 155ms - 0ms}{160ms} \right\rfloor \cdot 32ms$$

$$= \left\lfloor \frac{I + 20ms}{80m} \right\rfloor \cdot 30ms + \left\lfloor \frac{I + 30ms}{80ms} \right\rfloor \cdot 30ms + \left\lfloor \frac{I + 5ms}{160ms} \right\rfloor \cdot 32ms$$

2. Für welche I müssen Sie tC,ges(I) konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und dazu die zu untersuchenden Punkte in diesem Bereich.

11.1

$$^{*1}\; 0 < I < \mathrm{kgV}(t_{P_{min,i}}) + \max(t_{Ph,i})$$

$$0 < I < \mathrm{kgV}(80ms, 80ms, 160\,ms) + (0,0,0) = \underline{\underline{160ms}}$$

$\rightarrow$ Hyperiode $]0;160[$: I ist im Bereich $]0;160[$ zu untersuchen

I_A: {80 ms}
I_B: {80 ms}
I_C: {160 ms}

I: {80ms, 160ms}

eigentlich nicht wegen Bedingung *1

> A: Die Bedingung für den Intervallbereich könnte man als Folgende formulieren: `0 < min(I) <= max(I) < kgV(80, 160)` Da 160 [ms] die minimale Prozesszeit von C ist, sollte das eigentlich auch noch untersucht werden. Folgende Punkte können untersucht werden: {80, 160}. Bei den 160 [ms] sind wir uns unsicher gewesen, ob diese noch mit einbezogen werden sollen, da 160 < kgV(80, 160) = 160, damit nicht passen würde. Aber zum Zeitpunkt 160 [ms] wird C ausgeführt. Vielleicht können wir hierzu eine Rückmeldung erhalten.

3. Führen Sie den Realzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

> A: Eine schritthaltende Verarbeitung ist möglich, da die folgende Bedingung für alle Intervalle erfüllt wird: `tC,gesamt(I) <= I`. Die folgende Abbildung zeigt die Durchführung des Realzeitnachweises.

11.2

Hinreichender Schedulingtest:

$$U = \sum_{j=1}^{n} \frac{t_{Emax,j}}{\min(t_{Dmax,j}, t_{Pmin,j})} \leq 1$$

$$U = \frac{30}{60} + \frac{30}{50} + \frac{32}{155} \leq 1$$

$$= 1{,}30645 \leq 1 \quad ↯$$

=> Hinreichender Test nicht erfüllt.
Es sollte der notwendige Schedulingtest durchgeführt werden.

Notwendiger Schedulingtest:

$$t_{C,ges}(I) = \sum_{j=1}^{n} \left\lfloor \frac{I + t_{Pmin,j} - t_{Dmax,j} - t_{Ph,j}}{t_{Pmin,j}} \right\rfloor \cdot t_{Emax} \leq I$$

$$t_{C,ges}(80ms) = \left\lfloor \frac{80 + 80 - 60}{80} \right\rfloor \cdot 30ms + \left\lfloor \frac{80 + 80 - 50}{80ms} \right\rfloor \cdot 30ms + \left\lfloor \frac{80 + 160 - 155}{160} \right\rfloor \cdot 32ms$$

$$= 60ms \leq 80ms \quad ✓$$

$$t_{C,ges}(160ms) = \left\lfloor \frac{160 + 80 - 60}{80} \right\rfloor \cdot 30ms + \left\lfloor \frac{160 + 80 - 50}{80} \right\rfloor \cdot 30ms + \left\lfloor \frac{160 + 160 - 155}{160} \right\rfloor \cdot 32ms$$

$$= 60ms + 60ms + 0ms = 120ms \leq 160ms \quad ✓$$

hier sind wir uns nicht sicher (wie in 5.2 beschrieben), haben das Intervall trotzdem berechnet

Index der Kommentare

10.1 Die Zeiten sind fix, da kann nichts geändert werden.
Wenn die Prioritäten aber wie folgt angepasst werden, ist eine schritthaltende Abarbeitung möglich: B > A > C.

11.1 Es müssen ALLE Sprungstellen untersucht werden. Somit ergibt sich die Menge [50ms, 60ms, 130ms, 140ms, 155ms]

11.2 Durch Folgefehler die restlichen Sprungstellen nicht beachtet. Könnt ihr ja für euch nochmal durchrechnen.