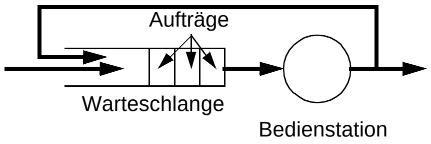
4 Prozessorvergabe in Monoprozessoren

4.1 Die operationelle Methode

Mehrere Aufträge (Prozesse) werden im Zeitmultiplex von einem Prozessor bearbeitet:



- **Frage: Welchen Einfluß übt die Auswahl-(Zuordnungs-)Strategie auf Verweilzeiten und Durchsatz aus?**

- **Systembeschreibung bei operationeller Betrachtung:**
Abarbeitung der Aufträge wird durch Warte- und Bedienfunktionen beschrieben.

- **Wartefunktion:**

$$W_i(t) = \begin{cases} 1 & \text{falls der Auftrag } P_i \text{ während des Intervalls} \\ & [t\Delta t, (t+1)\Delta t) \text{ auf Bedienung wartet,} \\ & \text{aber nicht bearbeitet wird} \\ 0 & \text{sonst} \end{cases}$$

- **Bedienfunktion:**

$$R_i(t) = \begin{cases} 1 & \text{falls der Auftrag } P_i \text{ während des Intervalls} \\ & [t\Delta t, (t+1)\Delta t) \text{ bedient wird} \\ 0 & \text{sonst} \end{cases}$$

Nebenbedingung 1

$$\forall i(1 \le i \le n \Rightarrow \exists t((0 \le t < T \wedge (R_i(t) = 1)) \wedge \forall t(0 \le t < T \Rightarrow (W_i(t)R_i(t) = 0))))$$

- **Abgeleitete Begriffe**
  - **Gesamtwartezeit des Auftrags P_i:**

$$W_i := \sum_{t'=0}^{T-1} W_i(t')\Delta t$$

  - **Gesamtbedienzeit des Auftrags P_i:**

$$R_i := \sum_{t'=0}^{T-1} R_i(t')\Delta t$$

  - **Restbedienzeit von P_i zum Zeitpunkt t:**

$$\vec{R}_i(t) := \sum_{t'=t}^{T-1} R_i(t')\Delta t$$

- **Von P_i zum Zeitpunkt t verbrachte Wartezeit:**

$$\overleftarrow{W}_i(t) := \sum_{t'=0}^{t-1} W_i(t')\Delta t$$

- **Restbedienzeit der zum Zeitpunkt t wartenden Aufträge:**

$$W\vec{R}(t) := \sum_{i=1}^{n} W_i(t)\vec{R}_i(t)$$

- **Bisherige Wartezeit der zum Zeitpunkt t bedienten Aufträge:**

$$R\overleftarrow{W}(t) := \sum_{i=1}^{n} R_i(t)\overleftarrow{W}_i(t)$$

- **Mittlere Restbedienzeit für die wartenden Aufträge:**

$$W\vec{R} := \frac{1}{T}\sum_{t'=0}^{T-1} W\vec{R}(t')$$

Prozessorvergabe - Monoprozessoren: Operationelle Methode

- **Mittlere bisherige Wartezeit der in Bedienung befindlichen Aufträge:**

$$R\overleftarrow{W} := \frac{1}{T}\sum_{t'=0}^{T-1} R\overleftarrow{W}(t')$$

- **Analog werden $R\overrightarrow{W}$, $W\overleftarrow{R}$ und $R\overrightarrow{R}$ definiert.**

- **Zahl der zum Zeitpunkt t wartenden Aufträge (= Warteschlangenlänge):**

$$Q(t) := \sum_{i=1}^{n} W_i(t)$$

- **Mittlere Warteschlangenlänge:**

$$E[Q] := \frac{1}{T}\sum_{t=0}^{T-1} Q(t)$$

Prozessorvergabe - Monoprozessoren: Operationelle Methode

- **Mittlere Wartezeit:**

$$E[W] := \frac{1}{n}\sum_{i=1}^{n} W_i$$

- **Mittlere Ankunftsrate: (= 1/mittleres Ankunftsintervall)**

$$\lambda := \frac{n}{T\Delta t}$$

- **Zahl der zum Zeitpunkt t im System befindlichen Aufträge:**

$$N(t) := \sum_{i=1}^{n} (W_i(t) + R_i(t))$$

- **Mittlere Zahl der im System befindlichen Aufträge:**

$$E[N] := \frac{1}{T}\sum_{t=0}^{T-1} N(t)$$

Prozessorvergabe - Monoprozessoren: Operationelle Methode

- **Verweilzeit des Auftrags P_i:**

$$U_i := \sum_{t=0}^{T-1} (W_i(t) + R_i(t))\Delta t$$

- **Mittlere Verweilzeit:**

$$E[U] := \frac{1}{n}\sum_{i=1}^{n} U_i$$

Prozessorvergabe - Monoprozessoren: Operationelle Methode

S4.1 **Satz**

1. $W\overrightarrow{R} = R\overleftarrow{W}$

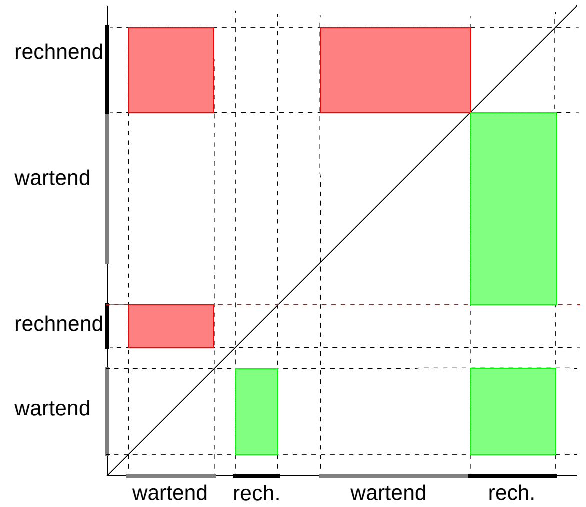
2. $R\overrightarrow{W} = W\overleftarrow{R}$

3. $R\overrightarrow{R} = \frac{1}{2T\Delta t}\sum_{i=1}^{n} R_i(R_i + \Delta t)$

4. $R\overleftarrow{W} + R\overrightarrow{W} = \frac{1}{T\Delta t}\sum_{i=1}^{n} W_i R_i$

BP 2 Prozessorvergabe - Monoprozessoren: Operationelle Methode

Beweis zu $\overrightarrow{WR} = \overleftarrow{RW}$: Betrachtung eines einzelnen Prozesses



Linke Seite:
Summe der roten Rechtecke

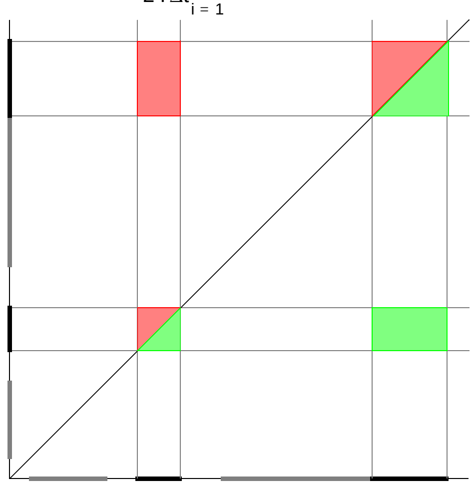
Rechte Seite:
Summe der grünen Rechtecke

Aus Symmetriegründen sind beide gleich

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

BP 2 Prozessorvergabe - Monoprozessoren: Operationelle Methode

Beweis zu $\overrightarrow{RR} = \frac{1}{2T\Delta t}\sum_{i=1}^{n} R_i(R_i + \Delta t)$



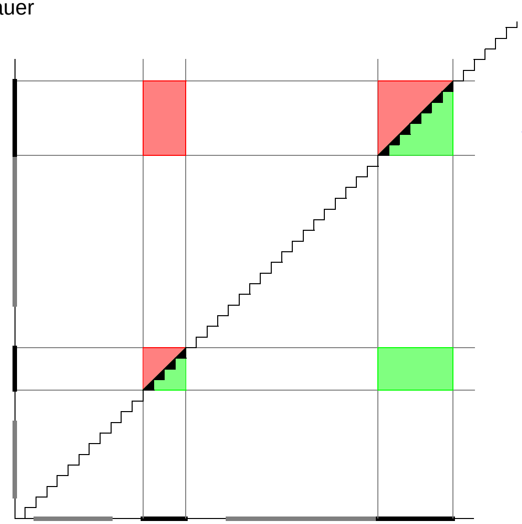
Linke Seite
Summe der roten Flächen

Im Grenzfall $\Delta t \rightarrow 0$
Summe der roten Flächen =
Summe der grünen Flächen

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

BP 2 Prozessorvergabe - Monoprozessoren: Operationelle Methode

genauer

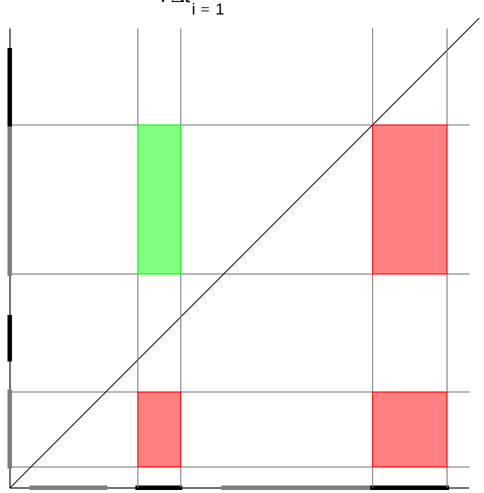


Stufe = Δt

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

BP 2 Prozessorvergabe - Monoprozessoren: Operationelle Methode

Beweis zu $\overleftarrow{RW} + \overrightarrow{RW} = \frac{1}{T\Delta t}\sum_{i=1}^{n} W_i R_i$



Linke Seite:

1. Summand =
Summe der roten Flächen

2. Summand =
Summe der grünen Flächen

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

S4.2 **Theorem von Little**

1. $E[Q] = \lambda E[W]$
2. $E[N] = \lambda E[U]$

Beweis:

Einfache Umformungen der Definitionen

Bezeichnungen

Ankunftszeitpunkt des Auftrags P_i

$$a_i := \min_{0 \le t < T} \{ t \mid W_i(t) + R_i(t) \neq 0 \}$$

Fertigstellungszeitpunkt des Auftrags P_i

$$e_i := \max_{0 \le t < T} \{ t \mid R_i(t-1) = 1 \}$$

Nebenbedingung 2

$$\forall i \forall t (0 \le t < T \Rightarrow (a_i \le t < e_i \Leftrightarrow (W_i(t) + R_i(t) = 1)))$$

S4.3 **Kleinrock**

Das System verfüge über eine einzige Bedienstation. Dann hat für alle Strategien, die

1. **nicht verdrängend sind (d. h.** $R\overrightarrow{W} = 0$**),**
2. **die Bedienstation nur unbenutzt lassen, wenn keine Aufträge im System sind und**
3. **Ankunfts- und Bedienzeiten nicht beeinflussen,**

die Größe $\sum_{i=1}^{n} R_i W_i$ **den gleichen Wert.**

Beweis:

1. $\overrightarrow{R}(t) := W\overrightarrow{R}(t) + R\overrightarrow{R}(t)$ **ist unter den gemachten Voraussetzungen ein sägezahnähnlicher Treppenzug, dessen Form unabhängig von der Zuordnungsstrategie ist.**
2. **Da nach Annahme** R_i **unabhängig von der Strategie ist, gilt dies wegen Satz 3.1 auch für** $R\overrightarrow{R}$
3. **Wegen 1. ist auch der Mittelwert** $\overrightarrow{R} = W\overrightarrow{R} + R\overrightarrow{R}$ **strategie-unabhängig und damit wegen 2. auch** $W\overrightarrow{R}$**. Aufgrund von Satz 3.1 gilt dies dann auch für** $R\overleftarrow{W}$**.**
4. **Nach Voraussetzung ist die Strategie nicht verdrängend, also** $R\overrightarrow{W} = 0$**.**

   Mit Satz 3.1 ergibt sich $RW = R\overleftarrow{W} + R\overrightarrow{W} = \frac{i}{T\Delta t} \sum_{i=1}^{n} R_i W_i$**, woraus wegen 3. die Behauptung folgt.**

S4.4 **Satz: Minimierung der mittleren Verweilzeit bei nicht verdrängenden Strategien**

In einem System mit einer Bedienstation, das zum Zeitpunkt 0 bereits alle Aufträge enthält (d. h. für alle i ist $R_i(0) + W_i(0) = 1$**) und das keine Verdrängung zuläßt, wird die mittlere Verweilzeit minimiert, wenn die Aufträge nach aufsteigenden Bedienzeitanforderungen abgearbeitet werden. Solche Vorgehensweisen werden als** *shortest job first (SJF)* **bezeichnet.**

S4.5 Satz: Minimierung der mittleren Verweilzeit bei verdrängenden Strategien

In einem System mit einer Bedienstation, das n Aufträge zu bearbeiten hat, minimiert die Strategie *preemptive shortest job first (PSJF)* **die mittlere Verweilzeit, falls Verdrängung und Zuordnung von Aufträgen keine Zeit beanspruchen.**

L4.5.1 Lemma

Eine Strategie S, die die mittlere Verweilzeit minimiert, erzeugt keine Leerstellen, d. h. in jedem Intervall $[t_1, t_2)$ ist ein Auftrag in Bearbeitung, falls während dieses Intervalls immer Aufträge im System sind.

L4.5.2 Lemma

Eine Strategie S, die U minimiert, erzeugt höchstens zu den Ankunftszeitpunkten a_i Verdrängungen.

Bezeichnung

Jedem Auftrag sei ein Zielzeitpunkt z_i zugeordnet, zu dem spätestens seine Fertigstellung erfolgen sollte.

S4.6 Satz: Minimierung der maximalen Verspätung ohne Verdrängung

Ein System mit einer Warteschlange und einer Bedienstation, das zum Zeitpunkt 0 bereits alle Aufträge enthält und keine Verdrängung zuläßt, minimiert die maximale Verspätung $\max_i\{L_i | (L_i = e_i - z_i)\}$**, wenn die Aufträge nach aufsteigender Zielzeit z_i abgearbeitet werden** *(deadline scheduling, earliest deadline first, EDF)*.

S4.7 Satz: Minimierung der maximalen Verspätung mit Verdrängung

In einem System mit einer Warteschlange und einer Bedienstation wird $\max_i\{L_i | (L_i = e_i - z_i)\}$ **minimiert, wenn zu jedem Zeitpunkt unter den im System vorhandenen Aufträgen einer mit kleinstem Zielzeitpunkt zugeordnet wird** *(preemptive deadline scheduling, preemptive earliest deadline first, PEDF)*.

Beweis: S sei ein Ablaufplan gemäß PEDF

O. B. d. A. Numerierung so, daß $i < j \Leftrightarrow z_i < z_j \vee (z_i = z_j \wedge e_i < e_j)$

k sei kleinster Index mit L_k maximal

1) Für alle i $L_i \le L_k \Rightarrow e_i - z_i \le e_k - z_k \Rightarrow z_k - z_i \le e_k - e_i$

2) $i < k \Rightarrow z_i \le z_k \Rightarrow 0 \le z_k - z_i \le e_k - e_i \Rightarrow e_i \le e_k$

$i \ne k$: $\Rightarrow e_i < e_k$

3) Beh.: Teilsystem $P_1, P_2, \ldots, P_k$ abgearbeitet wie S ist in diesem Teilsystem ebenfalls eine Abarbeitung nach PDS; es ist lediglich zu zeigen, daß in dem Teisystem keine Lücken entstehen.

Nach Definition der Indizierung: $k < r \Rightarrow z_k < z_r \vee (z_k = z_r \wedge e_k < e_r)$

Im ersten Fall wird P_r nur in Intervallen bearbeitet, während derer kein Auftrag des Teilsystems im System ist,

im zweiten Fall hätte P_r eine größere Verspätung als P_k, so daß er nicht eintreten kann.