GMR-GI-GfK Fachtagung Prozessrechner '77 Augsburg, 7.-8. März 1977, Informatik Fachberichte, 7, Springer, Berlin

Über warteZeitoptimale Abfertigungsstrategien für Prozeßrechner*)

Bernhard Walke
AEG-TELEFUNKEN, Forschungsinstitut
Postfach 1730, 7900 Ulm

Für Prozeßrechner im mittleren und oberen Leistungsbereich bietet die Berechnung der Verkehrsflüsse ein wertvolles Hilfsmittel für die Dimensionierung der Rechner-Betriebsmittel. Ein besonders wichtiges Problem ist die Vergabe von Prioritäten als Grundlage für eine, hinsichtlich geforderter Antwortzeiten optimal abgestimmte, Ablaufsteuerung. Es werden unterschiedliche Belastungsfälle betrachtet, Optimierungsziele vorgestellt und Regeln für die optimale Ablaufsteuerung angegeben. Ergebnisse verschiedener Strategien werden verglichen. Es zeigt sich, daß die allgemein übliche Anwendung statischer Prioritäten nicht für alle Belastungsfälle vertretbar ist. Vielfach bieten die von Universalrechnern her bekannten internen Prioritäten erhebliche Vorteile.

1. Modell eines Prozeßrechners

Wir betrachten im folgenden nur den ereignisgesteuerten Betrieb eines Prozeßrechners. Dabei treffen zu verarbeitende Daten (=Anforderungen) zu nicht vorherbekannten Zeitpunkten ein. Jede Anforderung stößt einen Rechenprozeß im Rechner an. Die Bediendauer des Rechenprozesses ist entweder exakt vorausbekannt oder durch eine Wahrscheinlichkeitsverteilung beschreibbar. Es ist Aufgabe des Rechners, die einzelnen, um seine Betriebsmittel konkurrierenden, Rechenprozesse gemäß vorgegebener Strategien zu bedienen. Dabei werden bestimmte Rechenprozesse bevorzugt bedient, so daß vorgegebene Optimierungsziele möglichst gut eingehalten werden. Ein wesentliches Kennzeichen des ereignisgesteuerten Betriebs ist, daß Bearbeitungstermine nicht strikt eingehalten werden können. Die Verweildauer (=Wartezeit plus Bediendauer) einer Anforderung hängt vielmehr stark davon ab, wieviele Rechenprozesse gleichzeitig bedient werden wollen. Ihre Zahl kann stark schwanken. Als Optimierungsziele kommen deshalb nur Forderungen für die Momente der Verweildauer -bzw. Wartezeit- Verteilungsfunktion (1. Moment = Erwartungswert, 2. zentrales Moment = Varianz) und für ihre Quantile in Frage. Beispielsweise kann man fordern, daß 95% aller Wartezeiten von Rechenprozessen, die von einer bestimmten Sorte von Anforderungen ausgelöst wurden, kleiner als T_1 sein sollen (z.B. T_1=10-facher Bediendauer-Erwartungswert). Das 95% -Quantil ist dann gleich T_1.

Nicht alle Anforderungen müssen zu gleichwichtigen Rechenprozessen führen. Beispielsweise kann man N Typen von Anforderungen unterscheiden. Ein Anwendungsfall eines Prozeßrechners könnte dann durch mehrere Vorgaben für Optimierungsziele beschrieben werden:

*) Die Arbeit wurde zu 50% durch das 2. DV-Programm der Deutschen Bundesregierung gefördert. Erschienen in [3].

Anforderungen vom Typ i sollten eine erwartete Verweildauer V_i nicht überschreiten und das x %-Quantil sollte T_i sein, mit $1 \leq i \leq N$.

Zur Nachbildung und Berechnung solcher Belastungsfälle eignet sich das einfache in Bild 1 gezeigte Modell eines Prozeßrechners. Anforderungen vom Typ i treffen entsprechend einem Ankunftsprozeß beim Rechner ein und werden dort in einer Warteschlange i verwaltet. Wir unterstellen im folgenden unabhängige Ankunftsprozesse mit je negativ exponentiell verteilten Ankunftsabständen (Poisson- oder Markoff-Prozesse genannt). Mit Hilfe einer Abfertigungsstrategie ist festgelegt, an welchem der wartenden Rechenprozesse der Prozessor jeweils arbeitet. In jeder Warteschlange gilt die Strategie FCFS (Tabelle 1). Das Modell geht davon aus, daß alle wartenden Rechenprozesse wahlfrei bearbeitet werden können, also im Zentralspeicher des Prozeßrechners verfügbar sind. In der Praxis benötigt man zusätzlich Zentralspeicher-Belegungsstrategien, die dafür sorgen, daß Rechenprozesse rechtzeitig geladen werden, damit sie der Prozessor im Bedarfsfall verfügbar hat. Solche Strategien werden hier nicht betrachtet.

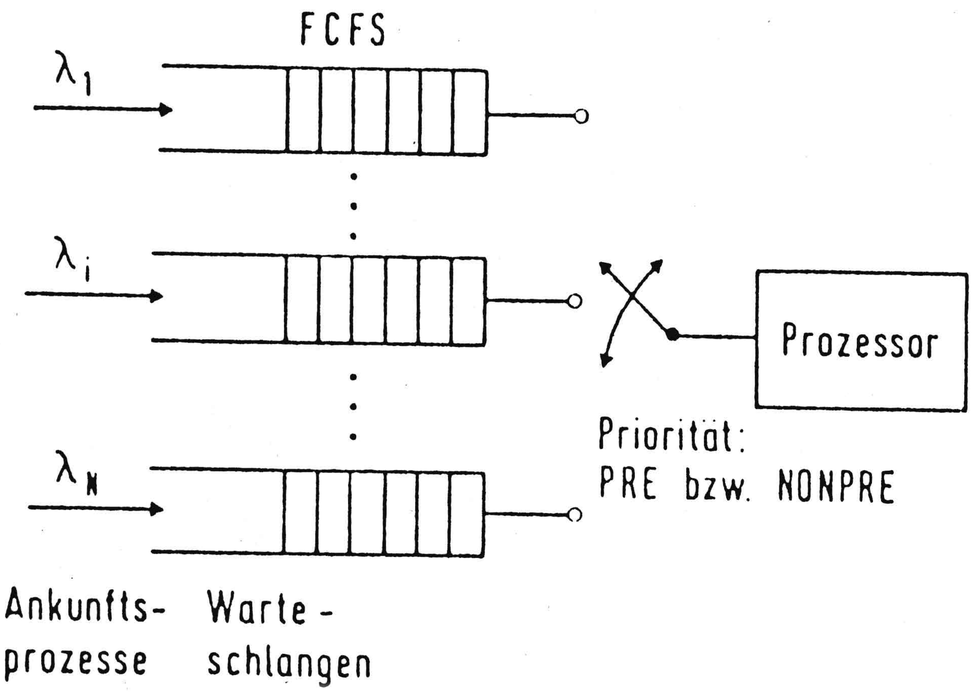


Bild 1:
Verkehrstheoretisches Modell eines Prozeßrechners.
Poisson-Ankunftsprozesse, beliebige Bediendauer-Verteilungen.

Die Bediendauer eines Rechenprozesses kann, muß aber nicht, zusammenhängend abgearbeitet werden. Ein Rechenprozeß gilt als beendet, wenn er den Prozessor freiwillig abgibt. In der Prozeßrechner-Praxis verursacht eine Anforderung u.U. einen Rechenprozeß, der nicht zusammenhängend durch den Prozessor abgearbeitet werden kann, z.B. wenn Rechen- und Transportphasen abwechseln. Im Rahmen des Modells wird jede Rechenphase als individueller Rechenprozeß betrachtet.

2. Abfertigung mit statischen Prioritäten

Prozeßrechner verwenden fast ohne Ausnahme dieselbe Abfertigungsstrategie zur Bearbeitung von Rechenprozessen durch den Prozessor. Diese Strategie setzt voraus, daß jedem Rechenprozeß eine Zahl zugeordnet worden ist, die seine Wichtigkeit (Priorität) für die Bearbeitung durch den Prozessor angibt. Der Rechenprozeß mit höchster Priorität

(kleinster Zahl) ist der wichtigste. Nichtunterbrechende Priorität (nonpreemptive priority) liegt vor, wenn der Prozessor eine begonnene Bearbeitung nicht mehr unterbricht und nur nach Bearbeitungsende den nächsten Rechenprozeß mit Hilfe der Priorität auswählt. Unterbrechende Priorität (preemptive priority) liegt vor, wenn der Prozessor die Bearbeitung eines Prozesses sofort unterbricht, falls ein anderer mit höherer Priorität den Prozessor verlangt. Wir betrachten hier nur den Fall, daß der unterbrochene Prozeß später, ohne Verlust der schon für ihn geleisteten Prozessorarbeit, fortgesetzt wird (PRE-resume).

Man spricht bei der beschriebenen Abfertigungsstrategie von der Verwendung externer statischer Prioritäten ohne bzw. mit Unterbrechung. Im Modell, Bild 1, könnte i die statische Priorität sein. Den allgemeineren Fall zeigt das Modell in Bild 2. Dort gibt es N Prioritäten vom Typ PRE (man spricht von N als der Zahl der Prioritätsebenen), und je Ebene gibt es R Prioritäten vom Typ NONPRE. Jede der Warteschlangen ij hat einen individuellen unabhängigen Markoff-Ankunftsprozeß mit der Ankunftsrate λ_{ij}. Läßt man zu, daß einzelne Raten $\lambda_{ij}=0$ sind, dann kann die Zahl nichtunterbrechender Prioritäten pro Ebene verschieden groß sein. Die Abarbeitung von in einer Warteschlange wartenden Anforderungen erfolgt im allgemeinen in der Reihenfolge des Eintreffens (FCFS). Z.B. wird in der Ebene i=2 die gesamte Warteschlange j=1 vor j=2 abgearbeitet.

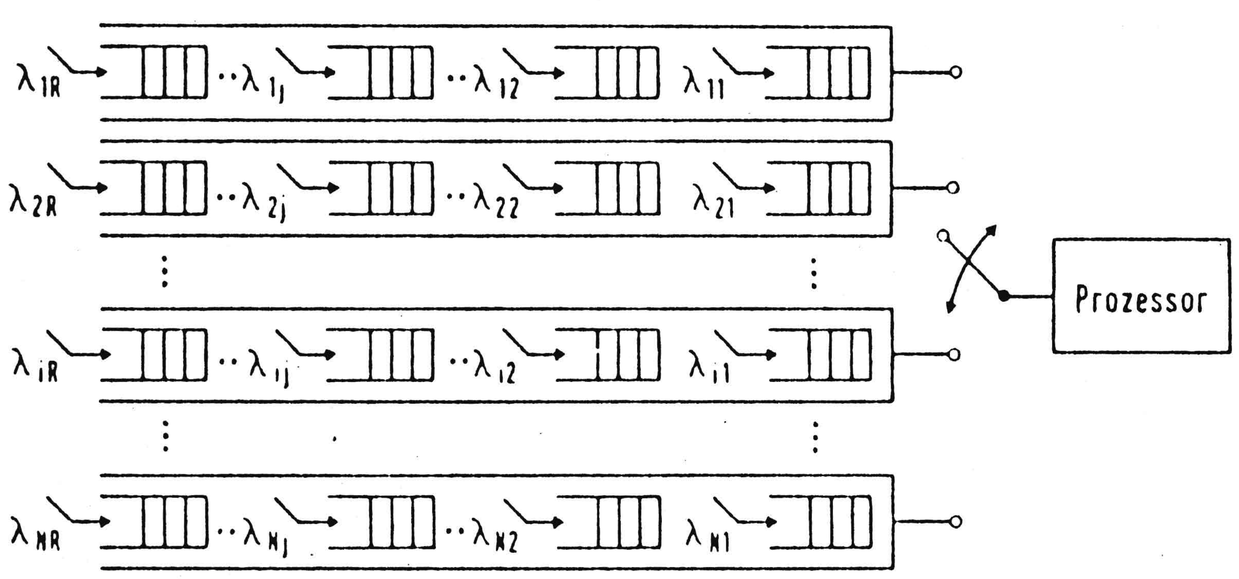


Bild 2:
Verkehrstheoretisches Modell mit N rein unterbrechenden Prioritäten und R Warteschlangen in jeder Ebene i. Bei Anwendung statischer Prioritäten sind die Prioritäten 1 bis R nichtunterbrechend und in jeder Warteschlange gilt die Abfertigungsstrategie FCFS.

2.1 Zur Optimalität statischer Prioritäten

Fordert man für einen gegebenen Anwendungsfall eines Prozeßrechners, daß alle Programmierer statische Prioritäten benutzen müssen, dann entzieht man sich damit einer Kontrolle des optimalen Einsatzes des Prozessors, wie auch der übrigen Betriebsmittel. Als Optimierungsziel bleibt nur die verschwommene Forderung, daß alles zufriedenstellend "läuft". Jede beliebige Prioritätenvergabe, die diese Forderung erfüllt, könnte als optimal bezeichnet werden. Möglicherweise

könnte eine irgendwie andere Festlegung der statischen Prioritäten, insbesondere auch des Typs (PRE bzw. NONPRE) der Prioritäten das Problem ebenso, oder besser lösen. Besser könnte heißen, daß z.B. die Zahl von Unterbrechungen pro Zeiteinheit deutlich kleiner würde, der Rechner jetzt mehr Kapazität für zusätzliche Aufgaben frei hätte, oder daß ein Prozessor kleinerer Leistung ausreichen würde.

Beispiel: Modell nach Bild 1, N=2, Ankunftsabstände und Bediendauern seien negativ exponentiell verteilt (= Markoff-Prozesse) mit den Erwartungswerten $1/\lambda_i$ bzw. β_i, rein unterbrechende Prioritäten (PRE). Wir interessieren uns für die mittlere Wartezeit W_i der Priorität i in Abhängigkeit von der Prozessor-Gesamtbelastung

$$\rho_{\leq N} = \sum_{i=1}^{N} \rho_i \quad \text{mit } \rho_i = \lambda_i \beta_i. \qquad (2.1)$$

Unter der Voraussetzung $\rho_{\leq N} < 1$ ist das Angebot ρ_i die Wahrscheinlichkeit, daß der Prozessor von der Priorität i belegt ist.
Sind die Bediendauer-Erwartungswerte β_i unterschiedlich, wie in Bild 3 (rechts)angegeben, so hängt die bezogene mittlere Wartezeit W_i/β_i (Ordinate) sehr von der Vergabe der statischen Prioritäten ab.

Die gezeigten Ergebnisse sind exakt und geschlossen berechenbar [1]. Erhalten die Rechenprozesse mit dem kleinen Bediendauererwartungswert β_a die höhere Priorität, so ergibt sich der Wartezeitverlauf für die erste/zweite Priorität durch die Kurven $1_o/2_o$. Erhalten Rechenprozesse mit dem größeren Bediendauer-Erwartungswert β_b die höhere Priorität, so erhält man für die erste, bzw. zweite Priorität die Kurven 1_p bzw. 2_p. Beide Arten der Prioritätsvergabe könnten einen Belastungsfall der Praxis zufriedenstellend lösen,

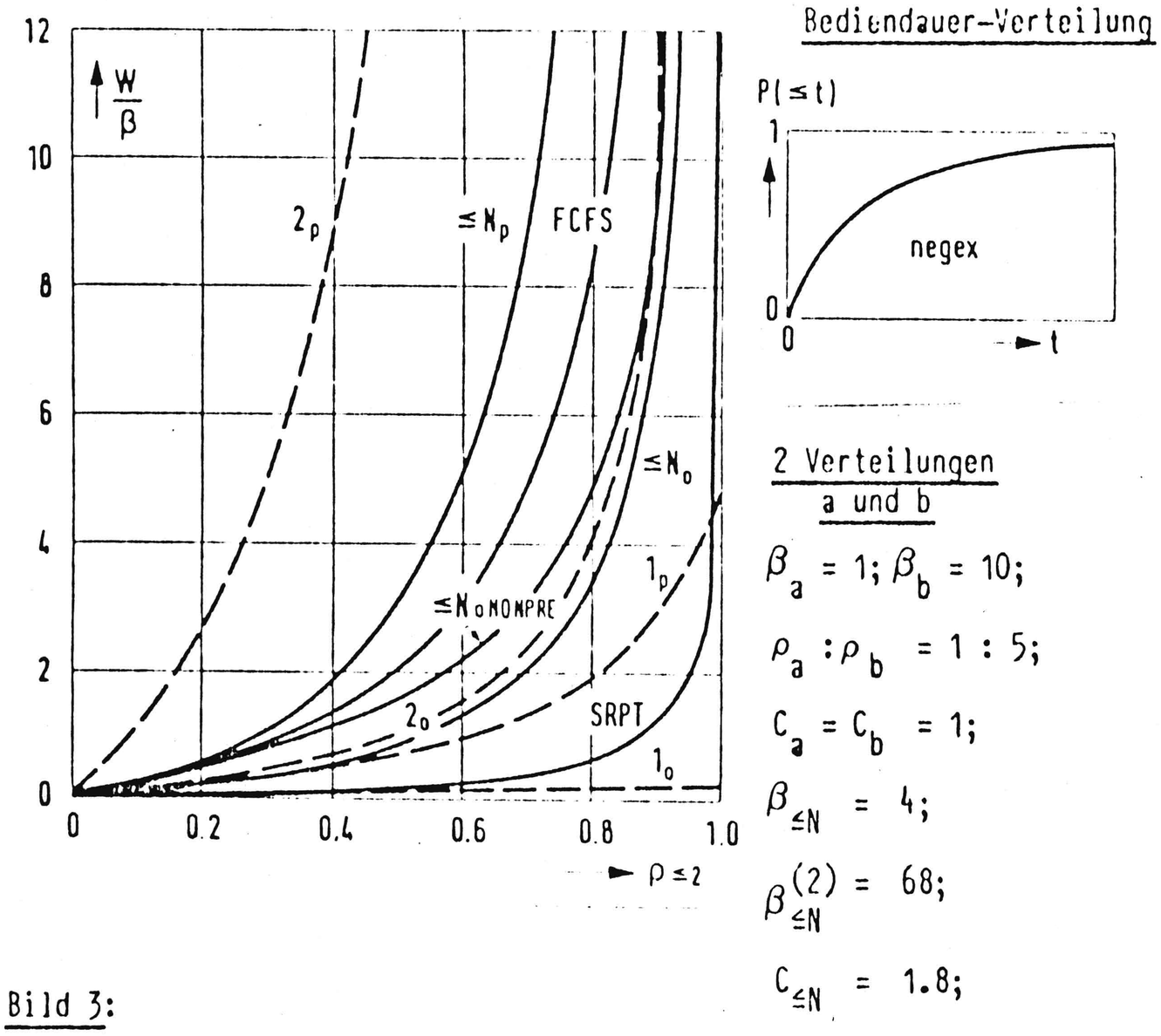


Bild 3:
Bezogene mittlere Wartezeit über dem Gesamtangebot. Modell, Bild 1, mit unterbrechenden Prioritäten. Die Kurvenpaare $1_o/2_o$ bzw. $1_p/2_p$ geben die bezogenen mittleren Wartezeiten W_i/β_i (für i=1,2) der statischen Prioritäten 1 und 2 bei optimaler bzw. pessimaler Prioritätenvergabe wieder. Die Kurven $\leq N_{Index}$ geben die bezogene gemeinsame mittlere Wartezeit $W_{\leq N}/\beta_{\leq N}$ bei statischen, die Kurven FCFS und SRPT bei internen Prioritäten wieder.

z.B. die Forderungen erfüllen, daß die bezogenen mittleren Wartezeiten für Rechenprozesse mit $\beta_a = 1$ und $\beta_b = 10$ je die Bedingung $W/\beta \leq 2$ erfüllen müssen. Man liest ab, daß die Kurven 1_o und 2_o diese Forderung fürein Gesamtangebot $\rho_{\leq N} \leq 0.65$ erfüllen. Bei ungünstiger Prioritätsvergabe (→ Kurven $1_p, 2_p$) muß $\rho_{\leq N} \leq 0.18$ sein (Kurve 2_p schneidet $W/\beta = 2$). Bei günstiger Vergabe der statischen Prioritäten kann die 3.6-fache Prozessor-Gesamtbelastung zugelassen, oder aber ein Prozessor mit entsprechend verminderter Leistung verwendet werden. Das Beispiel läßt sich für beliebige Bediendauer-Verteilungen vorführen. Bei sehr vielen Rechenprozessen wird die Frage der optimalen Prioritätenvergaben mit dem Ziel, rechenprozess-individuelle Zeitforderungen einzuhalten (zu unterbieten) und die gesamte zulässige Prozessorbelastung $\rho_{\leq N}$ zu maximieren praktisch unlösbar. Die Zahl der Alternativen ist zu groß.

Die allgemein akzeptierte Voraussetzung, die Verwendung statischer Prioritäten sei das am besten geeignete Mittel zur Ablaufsteuerung für Prozeßrechner, erweist sich, so gesehen, als fragwürdig. Es ist bekannt, daß statische Prioritäten zu rigoros unterschiedlicher Behandlung von Prozessen mit verschiedenen Prioritäten führen (vergleiche das obige Beispiel). Das muß nicht unbedingt erwünscht sein. Darüber hinaus kann man zeigen, daß unter bestimmten Voraussetzungen statische Prioritäten generell einen deutlich leistungsstärkeren Prozessor erfordern, als sogenannte interne Prioritäten, ohne daß dabei wesentliche Unterschiede im Echtzeit-Verhalten feststellbar wären. Mit solchen Beispielen wollen wir uns beschäftigen.

3. Minimierung der gemeinsamen mittleren Wartezeit

Bei vielen Anwendungsfällen von Prozeßrechnern gibt es einzelne Rechenprozesse, für die keine unterschiedlichen Wartezeit-Forderungen begründbar sind, die also zweckmäßig in eine Gruppe zusammengefaßt werden sollten. Entsprechend Bild 2 teilt man solchen Prozessen dieselbe Ebenenpriorität i zu. Verschiedene Gruppen solcher Prozesse haben verschiedene Ebenenprioritäten. Unterschiedliche Warteschlangen in einer Ebene für Prozesse mit unterschiedlichen Eigenschaften (Bediendauerverteilung, Ankunftsrate) sind hier zunächst nur ein Hilfsmittel der Verwaltung; ihre Vorteile werden später deutlich. Innerhalb einer Ebene unterscheiden sich Rechenprozesse durch die Zugehörigkeit zu einer der Warteschlangen j. Die Abarbeitungsreihenfolge konkurrierender Anforderungen innerhalb jeder Warteschlange j und zwischen den Warteschlangen einer Ebene i kann mit Hilfe extern festgelegter statischer Prioritäten, oder aber durch interne Priori-

täten gesteuert werden. Bei internen Prioritäten wird die momentane Priorität jedes Rechenprozesses dynamisch (abhängig vom Bearbeitungsfortschritt) ermittelt (Vergl. Abschn. 3.2).

Interne Prioritäten haben sich im Teilnehmer-Betrieb von Universalrechnern bewährt. Dort besteht ebenfalls die Aufgabe, für eine Gruppe von Rechenprozessen geforderte Wartezeiten (Mittel- und Maximalwerte) mit bekannter Wahrscheinlichkeit zu garantieren. Hat man statistische, oder sogar genaue Vorkenntnisse über die Bediendauern einzelner Rechenprozesse, dann kann man für alle denkbaren Belastungsfälle Strategien mit internen Prioritäten angeben, die zu einer minimalen mittleren Wartezeit aller Prozesse einer Gruppe führen. Nur in Sonderfällen erreichen Strategien mit statischen Prioritäten dasselbe günstige Ergebnis.

Im folgenden beschäftigen wir uns mit der Abarbeitung einer Gruppe von Rechenprozessen, wobei einzelne Rechenprozesse auch unterschiedliche Bediendauerverteilungen und Ankunftsraten, aber gleiche Forderungen für die Wartezeitverteilung haben können. Die Einordnung in ein Prioritätenschema mit unterbrechenden Ebenen -Prioritäten für verschiedene Gruppen- wird hier nicht behandelt. In der heutigen Prozeßrechner-Praxis werden unterschiedliche Rechenprozesse einer Gruppe grundsätzlich mit Hilfe statischer Prioritäten abgearbeitet. Wir wollen die zugehörigen Resultate für die Wartezeiten denen von Abfertigungsstrategien mit internen Prioritäten gegenüberstellen. Zunächst beschäftigen wir uns mit der Prioritätszuweisung bei statischen Prioritäten mit dem Ziel, eine minimale gemeinsame Wartezeit zu erzielen.

Das r-te Moment der gemeinsamen Wartezeitverteilung ist definiert zu

$$W_{\leq N}^{(r)} = \Big(\sum_{i=1}^{N} \lambda_i W_i^{(r)} \Big) / \lambda_{\leq N}, \qquad (3.1)$$

wobei $W_i^{(r)}$ das r-te Moment der Wartezeitverteilung der Priorität i ist und $\lambda_{\leq N} = \sum_{i=1}^{N} \lambda_i$ die Summe aller Ankunftsraten. Das r-te Moment der gemeinsamen Bediendauerverteilung $\beta_{\leq N}^{(r)}$ ergibt sich aus Gl.(3.1), indem man W durch β ersetzt. Für r=1 erhält man den Erwartungswert und schreibt $W_{\leq N}$, $\beta_{\leq N}$. Das Modell, Bild 1, wird zur Bearbeitung von N Rechenprozessen einer Gruppe verwendet. Die Ankunftsprozesse seien Markoff'sch, die Bediendauerverteilungen beliebig.

3.1 Optimale statische Prioritäten

Verwendet man statische Prioritäten vom Typ nichtunterbrechend (NONPRE), dann wird $W_{\leq N}$ minimal, wenn

$$\beta_1 \leq \beta_2 \leq \ldots \beta_i \ldots \leq \beta_N \qquad (3.2)$$

erfüllt ist, wobei i=1 die höchste Priorität ist [2]. Bei rein unterbrechenden Prioritäten (PRE) müssen die Bediendauer-Verteilungen negativ exponentiell verteilt sein, damit auch hier diese Regel (Gl.3.2) gilt. Bild 3 enthält für unser Beispiel Rechenergebnisse für die bezogene gemeinsame mittlere Wartezeit $W_{\leq N}/\beta_{\leq N}$ (Kurven $\leq N_o$, $\leq N_p$), wobei der Index o für optimale Prioritätenzuordnung, p für ungünstigste (pessimale) steht.

Sind die Bediendauern beliebig verteilt, dann kann keine Regel für die Anwendung statischer unterbrechender Prioritäten angegeben werden, die zu einer minimalen gemeinsamen mittleren Wartezeit führt. Man kann zeigen, daß unterbrechende Prioritäten zu einer kleineren Wartezeit $W_{\leq N}$ als nichtunterbrechende führen, wenn der Variationskoeffizient $C_{\leq N}$ der gemeinsamen Bediendauerverteilung von zu bearbeitenden Prozessen groß ($C_{\leq N} > 1$) ist. $C_{\leq N}$ ist definiert zu

$$C^2_{\leq N} = \beta^{(2)}_{\leq N} / \beta^2_{\leq N} - 1. \qquad (3.3)$$

Im Beispiel, Bild 3, ist $C_{\leq N} = 1.8$. Die gemeinsame mittlere Wartezeit bei optimalen nicht-unterbrechenden Prioritäten (Kurve $\leq N_{oNONPRE}$) ist größer als bei unterbrechenden Prioritäten (Kurve $\leq N_o$).

Untersuchungen haben ergeben, daß bei $C_{\leq N} < 1$ die Regel Gl.(3.2) auch bei unterbrechenden statischen Prioritäten nahezu zur minimalen Wartezeit $W_{\leq N}$ führt. Bei $C_{\leq N} > 1$ versagt sie völlig. Für solche Belastungsfälle kann die gemeinsame mittlere Wartezeit nur minimiert werden, wenn unterbrechende Prioritäten nicht extern statisch, sondern während der Bearbeitung intern im Rechner ermittelt werden.

3.2 Interne Prioritäten

Interne Prioritäten werden bei an sich gleichrangigen Anforderungen angewandt. Man unterscheidet die Fälle: vorausbekannte bzw. durch Wahrscheinlichkeitsverteilungen gegebene Bediendauern.

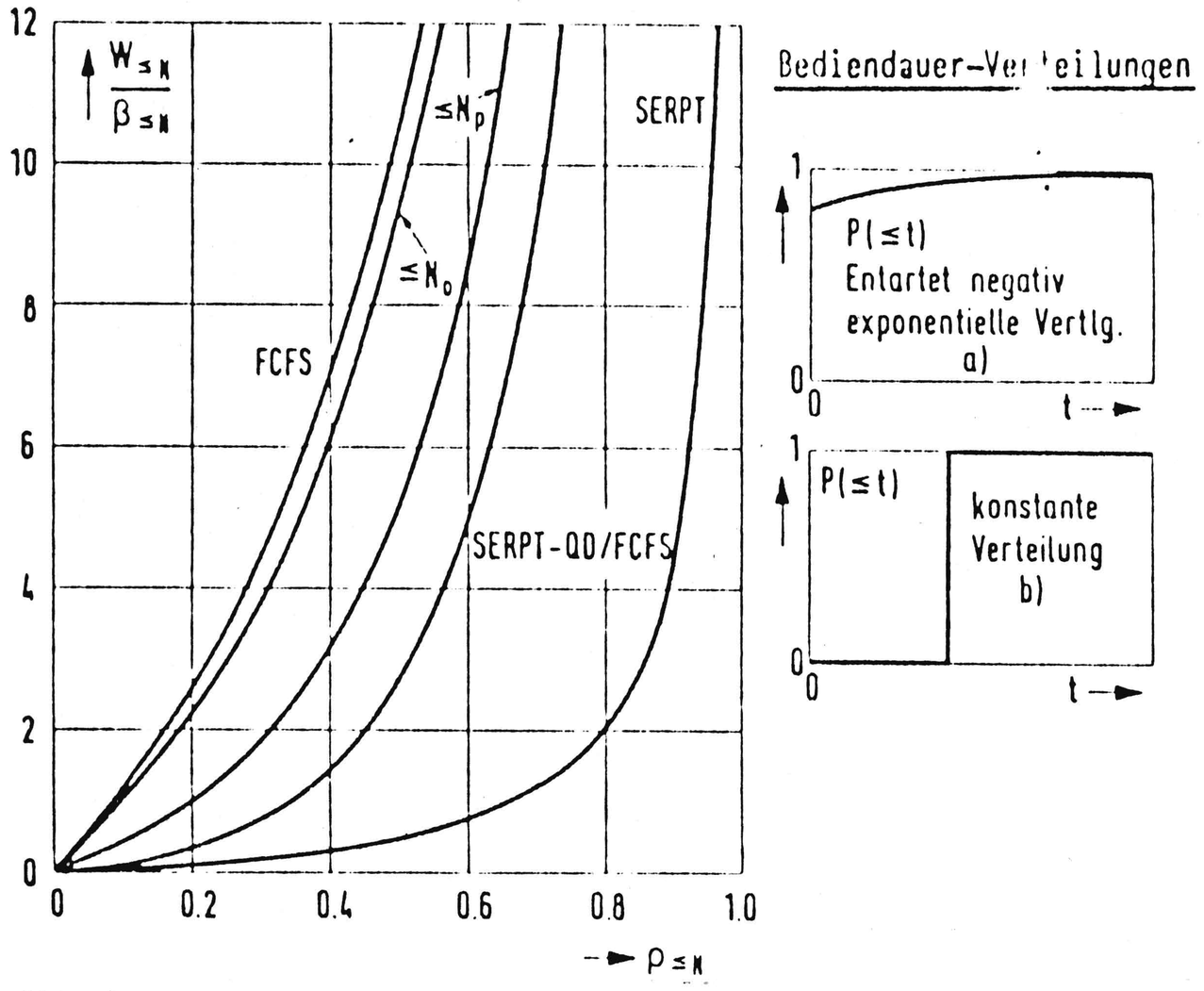
Bei der Berechnung interner Prioritäten wird der individuelle Bearbeitungsfortschritt jedes Rechenprozesses berücksichtigt, daraus die bekannte oder erwartete Restbediendauer ermittelt und die höchste unterbrechende Priorität für eine begrenzte Frist dem Prozeß zuge-

wiesen, der baldmöglichst fertigstellbar erscheint. Solche Berechnungen sind genau dann nötig, wenn eine weitere Anforderung eingetroffen ist, oder ein Rechenprozeß seine Frist überschritten hat, also (bei genügend großer Frist) selten.
Innerhalb einer Gruppe (vergleiche Bild 2, Ebene i) muß man beachten, ob man nacheinander für ein und denselben Rechenprozeß ij eingetroffene Anforderungen nacheinander abfertigen will (FCFS pro Schlange ij), oder ob auch Überholen erlaubt sein soll, wobei evtl. mehrere Anforderungen einer Schlange j angefangen aber nicht fertigbearbeitet sein dürfen. Deshalb unterscheiden wir interne Prioritäten ohne und mit Überholen. Ist Überholen verboten, dann kommen nur alle auf dem ersten Platz einer Schlange j der Gruppe (Ebene i) wartenden Anforderungen für eine Bearbeitung in Frage. Beispiele interner unterbrechender Prioritäten mit/ohne Überholen sind:

Name	Bediendauer-Vorkenntnisse	Überholen:	Kriterium
SRPT	alle	ja	shortest remaining processing time first
FCFS	beliebig	nein	first come first serve
LCFS	beliebig	ja	last come first serve
SERPT	statistische	ja	shortest expected remaining processing time first
SERT-FCFS	statistische	nein	wie SERPT
SERT-QD-FCFS (queue-length dependent)	statistische	nein	wie SERPT, bei gleich wichtigen wird die längste Schlange bevorzugt bearbeitet

Tabelle 1: Abfertigungsstrategien

Bild 4 zeigt Ergebnisse für ein Belastungsbeispiel (siehe rechts im Bild) mit N=10 Rechenprozessen, wobei je 5 dieselbe Bediendauerverteilung und Ankunftsrate haben. Entsprechend der Regel, Gl.(3.2), müßten die 5 Rechenprozesse mit der Verteilung a) die höchsten 5 statischen Prioritäten 1 bis 5 erhalten. Das entsprechende Resultat (Kurve $\leq N_o$) ist jedoch ungünstiger, als die gegenteilige statische Prioritätenvergabe (Kurve $\leq N_p$). Also gilt die Regel tatsächlich nicht für diesen Belastungsfall.



Bediendauer-Verteilungen

10 Verteilungen, je 5 von a) und b)

$\beta_a = 1$, $\mu = 0.04$, $p = 0.96$;

$C_a = 7$; $\beta_{\leq N} = 1.1$; $\beta^{(2)}_{\leq N} = 25.72$;

$\beta_b = 1{,}2$; $C_{\leq N} = 4.5$;

$\rho_1 = \ldots \rho_5$; $\rho_6 = \ldots \rho_{10}$;

$\rho_a : \rho_b = 1 : 1.2$

Bild 4:

Gemeinsame mittlere Wartezeit bei N = 10 unterbrechenden statischen Prioritäten (Kurven $\leq N_o$ und $\leq N_p$) und bei internen Prioritäten (Kurven FCFS, SERPT-QD/FCFS und SERPT). Die mit SERPT-QD/FCFS bezeichnete Kurve wurde durch Simulation gewonnen.

Interne Prioritäten ohne Überholen (SERPT-QD-FCFS) oder gar mit Überholen (SERPT) erzielen deutlich bessere Ergebnisse. Beispielsweise kann bei $\rho_{\leq N} = 0.5$ und schrittweiser Verbesserung der Strategie die bezogene Wartezeit $W_{\leq N}/\beta_{\leq N}$ von 9 über 5 über 3 auf 0.5 verringert werden. Anders gesehen kann z.B. ein geforderter Wert $W_{\leq N}/\beta_{\leq N} = 6$ bei $\rho_{\leq N} = 0.4$, 0.53, 0.63 oder gar 0.92 noch garantiert werden, was heißt, daß der Prozessor zu 60, 47, 37 oder bloß 8 % leerstehen muß. Im Extremfall könnte die Rechnerbelastung um 0.92/0.4 = 2.3 gesteigert werden, oder die Prozessorleistung um den Faktor 2.3 verringert werden, ohne daß die normierte Wartezeit verändert würde. Beim Verringern der Prozessorleistung nimmt allerdings die mittlere Verweilzeit zu. Die neue mittlere Bediendauer $\beta_{\leq N}$ wird größer und ebenso die neue mittlere Wartezeit $W_{\leq N}$. Die Koordinatenbezifferung in Bild 4 bleibt jedoch unverändert.

Die Kurve SERPT ist mit Hilfe des Modells, Bild 1, mit N = 15 unterbrechenden statischen Prioritäten berechenbar. Entsprechend der Strategie SERPT müssen die ersten 5 Prioritäten mit den Anforderungen der Verteilung a) belegt werden, die kurz rechnen ($\beta_i = 0$), wobei $\rho_i = p\rho_a/5 = 0$ (i=1,...,5) sein muß. Die Prioritäten 6 bis 10 sind mit Prozessen der Verteilung b) belegt ($\rho_6 = \ldots = \rho_{10} = \rho_b/5$). Die Prioritäten 11 bis 15 haben Angebote $\rho_{11} = \ldots \rho_{15} = (1-p)\rho_a/5$ und negativ exponentielle Bediendauern mit $\beta_i = \mu^{-1}$ (i=11,...,15).

Die kleinstmögliche Wartezeit erhält man mit der Strategie SRPT, vergleiche Bild 3. Das Ergebnis dieser Strategie ist für das Beispiel in Bild 4 noch günstiger, als es die von Bild 3 übertragene Kurve angeben würde. Die Strategie FCFS schneidet in beiden Beispielen, Bilder 3 und 4, schlecht ab.

Eine bezüglich der erwarteten gemeinsamen Wartezeit $W_{\leq N}$ optimale Abarbeitung führt, wie Bild 5 zeigt, hier auch zu verbesserten Wartezeitverteilungsfunktionen. An der Ordinate steht die Wahrscheinlichkeit, daß eine Wartezeit $w \leq t$ ist, an der Abszisse steht die auf die gemeinsame mittlere Bediendauer $\beta_{\leq N}$ bezogene Zeit t. Die gezeigten - durch den halblogarithmischen Maßstab bedingten Geraden sind Näherungen für den exakten Verlauf der Wartezeitverteilungen, wobei in der Näherung nur die beiden ersten Momente berücksichtigt sind.

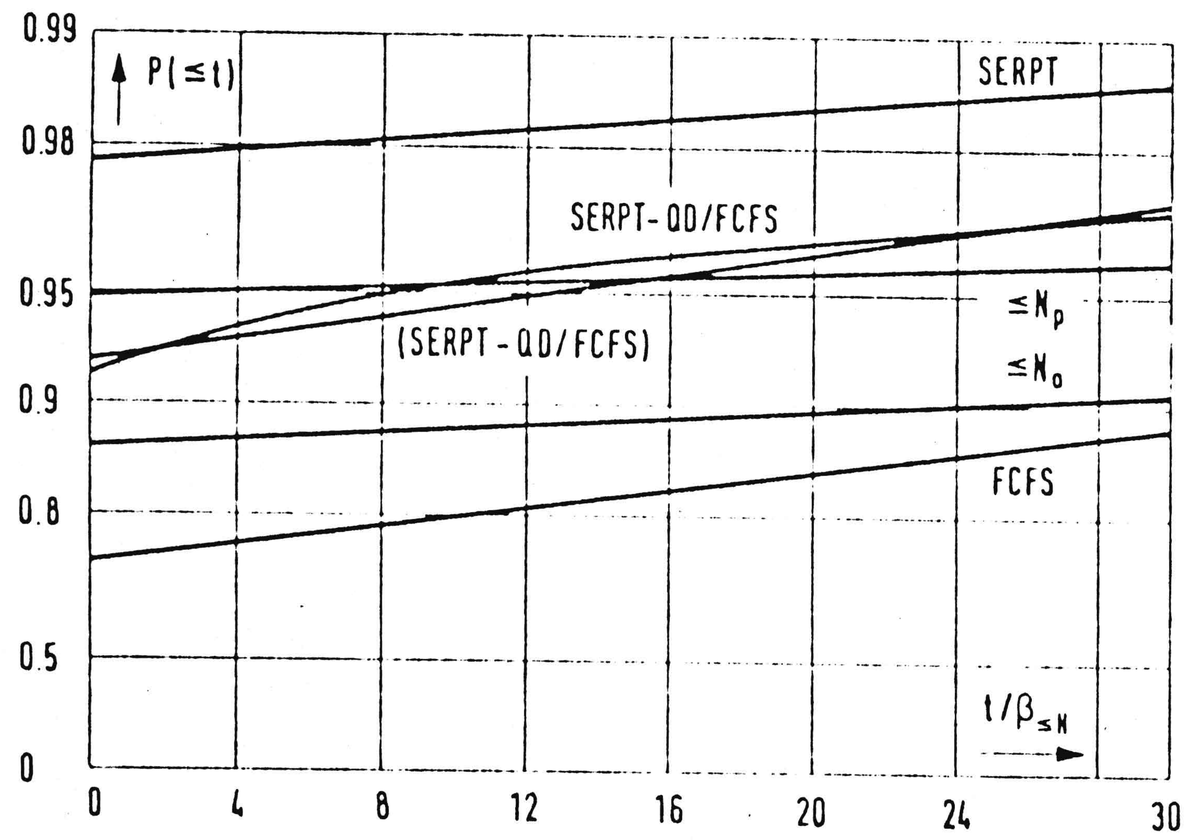


Bild 5:
Wartezeitverteilungen für das Beispiel, Bild 4. Gesamtangebot $\rho_{\leq N} = 0.55$. Die mit SERPT-QD/FCFS bezeichnete Kurve wurde durch Simulation gewonnen.

Als Näherungsfunktion ist ein Spezialfall der hyperexponentiellen Verteilung 2. Ordnung, nämlich eine entartet negativ exponentielle Verteilung verwendet worden:

$$P(\leq t) = 1-(1-p)\exp(-\mu t). \quad (3.4)$$

Ihre Parameter sind, mit C aus Gl. (3.3 mit W anstelle von β):

$$p = (C^2+1)/(C^2-1); \qquad \mu = 2[W(1+C^2)]^{-1} \quad (3.5)$$

Die Qualität der Näherung kann durch Vergleich mit dem exakten Verlauf am Beispiel der Strategie SERPT-QD-FCFS beurteilt werden: Die Kurve ist ein Simulationsergebnis, die Gerade eine Näherung nach Gl.(3.4).

Man vergleicht Wartezeitverteilungen verschiedener Strategien über Quantile, oder durch Vergleich der Wahrscheinlichkeiten, daß eine bestimmte Wartezeit, z.B. $t/\beta_{\leq N} = 20$, unterschritten wird. Je größer die entsprechende Wahrscheinlichkeit, umso günstiger ist die Verteilungsfunktion.

4. Schlußbemerkungen

Prozeß- und Universalrechner sind nebeneinander entwickelt und auf ihr heutiges Leistungsniveau gebracht worden. Obwohl in mancher Hinsicht enge Verwandtschaft, wenn nicht Identität besteht, verwendet man bis heute bei Prozeßrechnern fast ausschließlich statische Prioritäten. Die dabei resultierenden Nachteile, auch für das Echtzeitverhalten selbst, können erheblich sein. Es erscheint lohnend, den theoretischen Kenntnisstand über Abfertigungsstrategien bei Universalrechnern (insbesondere Teilnehmer-Rechensystemen) auch bei Prozeßrechnern zu nutzen. Das gilt nicht nur für den Prozessor, sondern auch andere Betriebsmittel wie Zentralspeicher, Transportkanäle, Dateien usw. Statische Prioritäten bergen die Gefahr, daß man das Echtzeitverhalten eines Prozeßrechners vor allem mit geballter Hardware-Leistung bei weitgehend ungenutzten Betriebsmitteln (d.h. $\rho_{\leq N} << 1$) sicherstellt.

Bisher sind keine Meßergebnisse zur Charakterisierung von Aufgabenprofilen für Prozeßrechner bekannt geworden. Die vorgelegten Ergebnisse unterstreichen die Notwendigkeit solcher Messungen. Eine Entscheidung für oder gegen die Vewendung von internen Prioritäten bei Prozeßrechnern kann erst nach ausgiebigen Messungen in unterschiedlichen Anwendungen gefällt werden.

Literatur

[1] T.W. Gay, P.H. Seaman:
Composite Priority Queue, IBM Journ. Res. & Development, Jan. 1975, pp. 78-81.

[2] R.W. Conway, W.L. Maxwell, L.W. Miller:
Theory of scheduling, Addison Wesley Publishing Company, Reading, Massachusetts, Palo Alto/London/Don Mills, Ontario, 1967.

[3] B.Walke (vorliegender Aufsatz):
GI, VDI/VDE-GMR, GfK Fachtagung "Prozeßrechner 1977", Augsburg, 7./8. März 1977, Informatik Fachberichte, Springerverlag Berlin, Heidelberg 1977.