SIMULATION IN DER NACHRICHTENVERKEHRSTHEORIE: PROBLEMSTELLUNGEN UND PROGRAMMIERSPRACHEN

von G. Kampe, P. Kühn und M. Langenbach-Belz
Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart

1. EINLEITUNG

In der Nachrichtenverkehrstheorie werden Probleme untersucht, die in Vermittlungssystemen für Fernsprech- oder Datenverkehr und in Rechnersystemen bei der Behandlung des Nachrichtenverkehrsflusses auftreten. Dabei wird neben der Analyse bestehender Systeme auch die Synthese neuer Systeme (Optimierung) vorgenommen. Ein System ist im wesentlichen durch seine Struktur, die Betriebsart und die statistischen Eigenschaften des Nachrichtenverkehrs gekennzeichnet. Als Untersuchungsmethoden sind Verkehrsmessungen, exakte oder näherungsweise Berechnungen sowie Simulationen auf Digitalrechnern üblich. Da die betrachteten Systeme oft sehr komplex sind, kann in vielen Fällen auch mit Großrechnern keine exakte Berechnung durchgeführt werden. Daher bildet die Simulation in der Nachrichtenverkehrstheorie überall dort ein unentbehrliches Hilfsmittel, wo entweder noch kein Rechenverfahren bekannt ist oder wo der Gültigkeitsbereich von Verfahren zur näherungsweisen Berechnung untersucht wird.

Der rechnerischen Analyse bzw. der Analyse durch Simulation geht die Modellbildung voraus. Die tatsächlichen Betriebsmittel und Vorgänge der Wirklichkeit werden dabei durch ein möglichst wirklichkeitsgetreues Modell beschrieben. Im zweiten Abschnitt der Arbeit werden zunächst die wesentlichsten Merkmale (Strukturelemente, Betriebsstrategien, Prozesse) von Modellen der Nachrichtenverkehrstheorie (kurz: verkehrstheoretische Modelle) vorgestellt, und es wird anhand dreier typischer Beispiele aus Vermittlungssystemen, Rechnersystemen sowie Netzen gezeigt, wie aus dem tatsächlichen System ein Modell entsteht. Es wird ferner kurz auf die Fragestellungen eingegangen, welche durch die Simulation beantwortet werden sollen.

Im dritten Abschnitt der Arbeit werden kurz die prinzipiellen Verfahren zur Simulation solcher Modelle anhand eines Beispiels aufgezeigt. Aufbauend auf den Erkenntnissen der vorangegangenen beiden Abschnitte wird im vierten Abschnitt der Arbeit auf Forderungen an Sprachen für Simulationsprogramme in der Nachrichtenverkehrstheorie eingegangen. Anhand eines einfachen Simulationsmodells (einstufiges Warte-Verlustsystem) wird schließlich im fünften Abschnitt eine Gegenüberstellung der folgenden vier Programmiersprachen vorgenommen:

FORTRAN IV
GPSS/360
SIMSCRIPT I.5
SIMULA 67.

Das betrachtete System wurde in jeder dieser vier Sprachen durch ein Simulationsprogramm nachgebildet. Die dabei über diese Sprachen gewonnenen Erfahrungen (z.B. Programmieraufwand, Speicherplatzbedarf, Programmlaufzeit) werden diskutiert.

2. PROBLEMSTELLUNGEN DER NACHRICHTENVERKEHRSTHEORIE

2.1 Allgemeines

In Nachrichtenvermittlungssystemen werden von Teilnehmern Verbindungswünsche (Anforderungen) gestellt, wodurch diese Teilnehmer aufgrund der gewählten Ziffern mit jeweils einem Zielteilnehmer automatisch verbunden werden sollen. Die äußeren Anreize lösen innerhalb des Vermittlungssystems eine Reihe von Vorgängen aus wie Verbinden des Teilnehmers mit einem freien Verbindungssatz, Anschalten an ein zentralisiertes Register, Aufnahme und Verarbeitung von Wählziffern, Einstellen der Koppelnetze zwischen rufendem und gerufenem Teilnehmer usw. Durch die Vielzahl der Verbindungswünsche und deren statistisch schwankende Eintreffzeitpunkte bzw. Belegungsdauern bedingt, können sich innerhalb des Vermittlungssystems Engpässe ergeben, welche sich durch Wartezeiten (Verzögerungen im Verbindungsaufbau) bzw. Besetztfälle äußern. Um einen hinreichend guten Service garantieren zu können, ist es erforderlich, die zentralen "Bedienungseinrichtungen" (Leitungen, Register, Markierer, Prozessoren, Koppelnetze usw.) ausreichend zu dimensionieren. Hierbei entstehen neben Fragen der notwendigen Anzahl von Einheiten vor allem Fragen hinsichtlich der optimalen Struktur und optimalen Betriebsstrategie, nach welcher die Bedienungseinrichtungen den Anforderungen zugeteilt werden.

Innerhalb von Rechnersystemen entstehen ganz ähnliche Probleme. Die einzelnen Anwenderprogramme stellen sehr unterschiedliche Anforderungen an die Betriebsmittel des Rechnersystems (Zentralprozessor, Ein/Ausgabeeinheiten, Hintergrundspeicher usw.). Größere Rechner werden heute weitgehendst nach der Multiprogramming-Betriebsweise organisiert. Die gleichzeitige Konkurrenz von mehreren Anwenderprogrammen bezüglich der Betriebsmittel kann ebenfalls auf interne Engpässe führen, welche durch passende Auslegung dieser Betriebsmittel bzw. intelligente Vergabe durch das Betriebssystem vermieden werden können.

Der Nachrichten-Weitverkehr für Ferngespräche, Fernschreiben bzw. Daten erfolgt über ausgedehnte Netze. Diese Netze bestehen aus Knoten mit Vermittlungs- und Steuereinrichtungen, welche über Nachrichtenkanäle untereinander verbunden sind. Die Struktur dieser Netze sowie die Betriebsstrategie haben einen entscheidenden Einfluß auf die Abwicklung des Nachrichtenverkehrs, insbesondere hinsichtlich momentaner Engpässe, welche durch Verkehrsspitzen bzw. Teilausfälle entstehen können.

2.2 Elemente verkehrstheoretischer Modelle

Im folgenden werden die wichtigsten Elemente zur vollständigen Beschreibung eines verkehrstheoretischen Modells zusammengestellt. Sie lassen sich in drei Kategorien einteilen: Strukturelemente, Betriebsstrategien und Prozesse.

2.2.1 Strukturelemente

Die Struktur eines Modells definiert die möglichen "Verkehrswege" von Anforderungen durch das System und gibt ferner die Art, Lage und Anzahl seiner Komponenten ("Strukturelemente") an. Die wichtigsten Strukturelemente sind in Tabelle 1 mit einer kurzen Definition zusammengefaßt. Während des Flusses von Anforderungen durch das System können Elemente wie Quellen, Bedienungseinheiten, Koppelpunkte usw. verschiedene diskrete Zustände einnehmen ("frei" oder "belegt" bzw. "blockiert" "offen" oder "geschlossen" usw.).

2.2.2 Betriebsstrategien

Die Betriebsstrategien eines Modells beschreiben die "Verkehrsregeln", nach welchen die Anforderungen durch das System laufen. Je nach Strukturelement lassen sich verschiedene Arten von Strategien unterscheiden vergl. Tabelle 2.

2.2.3 Prozesse

Der Nachrichtenverkehr zeichnet sich i.a. durch statistische Eigenschaften aus, wie z.B. die statistisch schwankenden Zeitabstände zwischen den Anforderungen, welche von einer großen Teilnehmerzahl an ein Nachrichtenvermittlungssystem gestellt werden bzw. die Dauern, in welchen von diesen Teilnehmern zentrale Einrichtungen (Leitungen, Register usw.) belegt werden. Im verkehrstheoretischen Modell lassen sich die statistisch schwankenden Eigenschaften durch den "Ankunftsprozess" bzw. den "Belegungsprozess" beschreiben, welcher durch die Wahrscheinlichkeitsverteilungsfunktion (VF) der Ankunftsabstände von Anforderungen bzw. der Belegungsdauern definiert wird. Messungen in realen Systemen haben ergeben, daß die meisten auftretenden Prozesse hinreichend genau durch wenige Standard-Verteilungsfunktionen beschrieben werden können.

Bei Ankunftsprozessen wird unterschieden, ob die Anforderungen von einer endlichen Anzahl q bzw. einer unendlich großen Anzahl von Quellen ($q \to \infty$) erzeugt werden. Dementsprechend wird die VF A(t) der Ankunftsabstände T_A entweder je freie Quelle oder für die Gesamtheit der Quellen definiert:

$$A(t) = P\left\{ T_A \leq t \right\} . \qquad (1)$$

Ein weiterer wichtiger Fall entsteht, wenn der Ankunftsprozeß von einer unendlich großen Zahl von Quellen erzeugt, bei Erreichen eines bestimmten Systemzustands jedoch "abgeschnitten" wird.

Entsprechend werden die Belegungsdauern T_H durch ihre VF H(t) beschrieben:

$$H(t) = P\left\{ T_H \leq t \right\} . \qquad (2)$$

Die häufigsten Typen von VF für Prozesse der Nachrichtenverkehrstheorie sind in Tabelle 3 zusammengestellt.

2.3 Beispiele verkehrstheoretischer Modelle

Die Modellbildung werde anhand dreier charakteristischer Beispiele aus Vermittlungssystemen, Rechnersystemen bzw. eines Netzknotens demonstriert. Es wird in jedem Beispiel außerdem kurz auf die Fragestellungen eingegangen, welche durch die Analyse beantwortet werden sollen.

2.3.1 Mehrstufige Koppelnetzwerke in Nachrichtenvermittlungssystemen

Bei der Durchschaltung von Verbindungen zwischen den Teilnehmern bzw. zwischen inneren Verbindungssätzen und zentralisierten Registern werden häufig mehrstufige Koppelanordnungen (Linksysteme) verwendet. Dabei werden die Verbindungen über mehrere Koppelmatrizen (Koppelvielfache) konjugiert durchgeschaltet.

Bild 1 zeigt ein Beispiel für ein Strukturmodell eines 4-stufigen Linksystems mit g_j Koppelmatrizen in Stufe j, $j = 1,2,3,4$. Die Koppelmatrizen aufeinanderfolgender Stufen sind über Zwischenleitungen nach

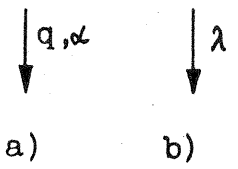
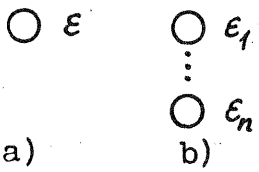
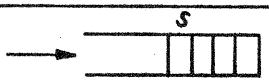
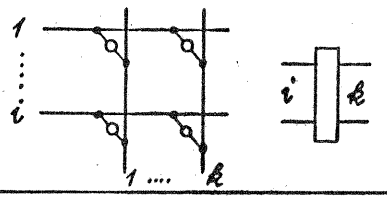
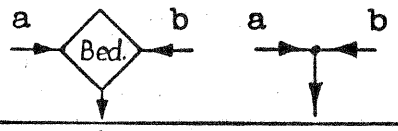
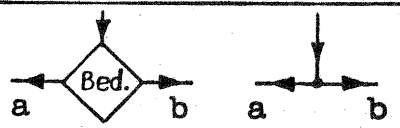
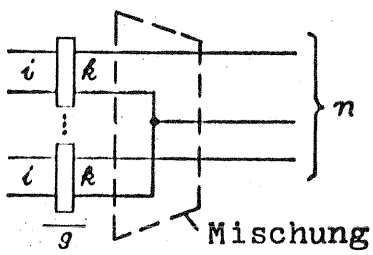
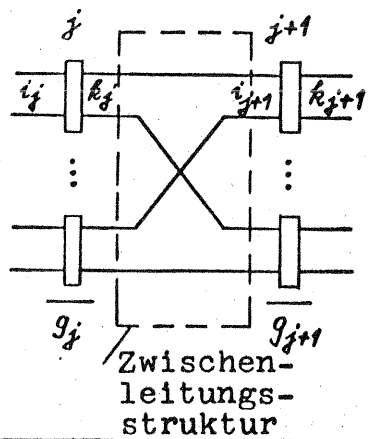
SYMBOL	ART	DEFINITION
	Verkehrs-quellen	Erzeugung von Anforderungen entsprechend eines "Ankunftsprozesses" a) $q < \infty$ Verkehrsquellen, Ankunftsrate α je freie Verkehrsquelle b) $q \to \infty$ Verkehrsquellen, gesamte Ankunftsrate λ
	Bedienungs-einheiten	Bedienung von jeweils einer Anforderung entsprechend eines "Belegungsprozesses" a) n=1 Bedienungseinheit, Enderate ε b) n>1 Bedienungseinheiten (Bündel, Gruppe), Enderaten $\varepsilon_1, \ldots, \varepsilon_n$
	Warte-speicher	Speicherung von max. s Anforderungen (Warteschlange)
	Koppelpunkt Schalter	Durchschalteelement für Anforderungen
	Koppel-matrix Koppel-vielfach	Element zur Durchschaltung zwischen i Eingängen und k Ausgängen. Jede Durchschaltung erfolgt über einen Koppelpunkt zwischen einem bestimmten Eingang und Ausgang
	Zusammen-führung	Element zur bedingten Zusammenführung des Verkehrs aus a bzw. b (Betriebsstrategie regelt Priorität)
	**Verzweigung**	Element zur bedingten Verzweigung des Verkehrs nach a bzw. b (Betriebsstrategie regelt Richtung)
	**Mischung**	Schema zur Zusammenführung (Konzentration) von Leitungen im Mischungsverhältnis (g.k):n > 1. Beschreibung durch Matrix MS[1:k, 1:g], wobei MS[x, y] = Nummer der Leitung an Ausgang x der Koppelmatrix y.
	Zwischen-leitungs-struktur	Schema zur Verbindung zwischen Ausgängen von Koppelmatrizen der Stufe j mit den Eingängen von Koppelmatrizen der Stufe j+1 ("Zwischenleitungen" oder "Linkleitungen") Verschiedenartige Verbindungsschemata dabei möglich: "geordnet aufgelegt", "zyklisch vertauscht aufgelegt", ferner auch Konzentration(Mischung) zwischen den Stufen möglich.

Tabelle 1. **Strukturelemente verkehrstheoretischer Modelle**

Tabelle 2. Betriebsstrategien verkehrstheoretischer Modelle

ART	Auswahlstrategien innerhalb einer Gruppe von Bedienungseinheiten	Auswahlstrategien innerhalb einer Warteschlange	Auswahlstrategien bei der Zusammenführung des Verkehrs	Auswahlstrategien bei der Verzweigung des Verkehrs
DEFINITION	Auswahl einer von mehreren freien Bedienungseinheiten aus einer Gruppe von Bedienungseinheiten	Auswahl einer von mehreren wartenden Anforderungen innerhalb einer Warteschlange	Auswahl einer von mehreren Herkunfts-Richtungen bzw. Herkunfts-Warteschlangen	Auswahl einer von mehreren Ziel-Richtungen bzw. Ziel-Warteschlangen
TYPISCHE BEISPIELE	- Sequentielles Absuchen mit/ohne Nullstellung - Zufällige Auswahl - Auswahl einer bestimmten Bedienungseinheit (Punktwahl) - Punktwahl mit mehrfachen Wiederholversuchen bei jeweils anderen Bedienungseinheiten	- Ankunftsreihenfolge (first-in, first-out FIFO) - Zufällige Auswahl (RANDOM) - Inverse Ankunftsreihenfolge (last-in, first-out LIFO) - Reihenfolge nach kürzester Bedienungszeit (shortest-job-first SJF) - Auswahl nach Prioritätsklassen (unterbrechende Priorität, nichtunterbrechende Priorität)	- Ankunftsreihenfolge - Zufällige Auswahl - Auswahl nach vorgegebenen Wahrscheinlichkeiten - Auswahl zyklisch fortschreitend - Auswahl nach momentanen Warteschlangenlängen - Auswahl nach Prioritätsklassen (unterbrechende Priorität, nichtunterbrechende Priorität, alternierende Priorität, Unterbrechungs-Distanz-Priorität, Unterbrechungs-Verzögerungs-Priorität)	- Auswahl einer bestimmten Richtung - Zufällige Auswahl einer Richtung - Sequentielles "Absuchen" von Richtungen nach dem "Überlaufprinzip" (Überlauf von Primär- auf Sekundärbündel, Primärspeicher auf Sekundärbündel, Primärspeicher auf Sekundärspeicher) - Auswahl nach Prioritätsklassen - Auswahl zyklisch fortschreitend

ART	DEFINITION
negativ-exponentielle VF	$P\{T \leq t\} = 1 - \exp(-\mu t)$, Mittelwert $E[T] = \frac{1}{\mu}$
konstante VF	$P\{T \leq t\} = \begin{cases} 0 & \text{für } 0 \leq t < 1/\mu \\ 1 & \text{für } t \geq 1/\mu \end{cases}$, $E[T] = \frac{1}{\mu}$
Erlang-k- VF	$P\{T \leq t\} = 1 - \exp(-k\mu t)\sum_{i=0}^{k-1} \frac{(k\mu t)^i}{i!}$, $E[T] = \frac{1}{\mu}$
Hyper-exponentielle VF k-ter Ordnung	$P\{T \leq t\} = 1 - \sum_{i=1}^{k} p_i \exp(-\mu_i t)$, $E[T] = \frac{1}{\mu} = \sum_{i=1}^{k} p_i \frac{1}{\mu_i}$ und $\sum_{i=1}^{k} p_i = 1$

Tabelle 3. Häufig auftretende Prozesse verkehrstheoretischer Modelle

einer vorgebbaren Zwischenleitungsstruktur ("Verdrahtungsschema") verbunden. Zwischen Stufe 1 und 2 ist zusätzlich noch eine Konzentration (Mischung) vorhanden. Die Koppelpunkte und Zwischenleitungen, welche an einer durchgeschalteten Verbindung beteiligt sind, bleiben während der gesamten Belegungsdauer (Gesprächsdauer bzw. Registerbelegungszeit) belegt. (Vergleiche durchgezogene Belegung in Bild 1).

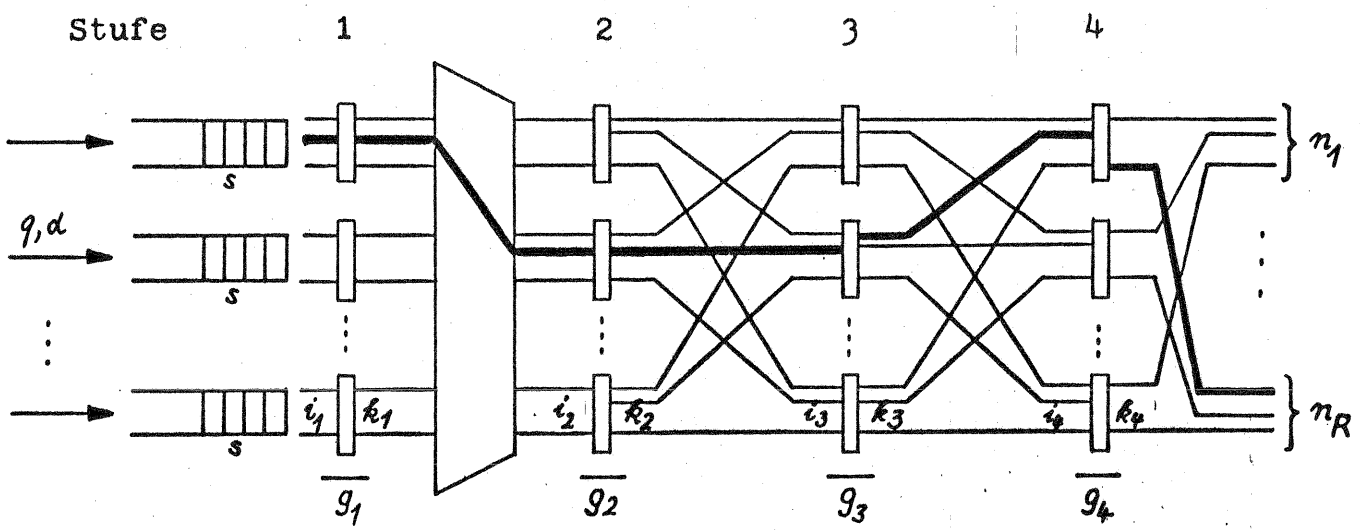


Bild 1. Mehrstufiges Koppelnetzwerk (Linksystem)

Am Ausgang des Linksystems erfolgt eine Verkehrsverzweigung in R Richtungen mit n_r Abnehmerleitungen (= Bedienungseinheiten), r=1,2,...,R. Die Anforderungen werden von g_1 Gruppen aus jeweils q Verkehrsquellen (Ankunftsrate α je freie Quelle) erzeugt. Je Gruppe ist ein Wartespeicher der Kapazität s vorgesehen.

Als Betriebsstrategien kommen verschiedene Auswahlstrategien für Bedienungseinheiten, Warteschlangen bzw. Anforderungen innerhalb von Warteschlangen in Frage, vergl. 2.2.2. Die Ankunftsabstände sind i.a. negativ-exponentiell, die Belegungsdauern entweder ebenfalls negativ-exponentiell (Gespräche) oder Erlang-k-verteilt (Aufnahme und Verarbeitung von Wählziffern im Register).

Als wichtigste Kenngrößen des Systems sind Warte- bzw. Verlustwahrscheinlichkeiten, mittlere Warteschlangenlängen, Mittelwerte und VF der Wartezeiten sowie die einzelnen Bündelbelastungen zu ermitteln.

2.3.2 Multiprogramming in Rechnersystemen

Eine wirkungsvolle Ausnutzung von Prozessoren (P) und E/A-Kanälen in Rechnersystemen ist durch Simultanarbeit und Konkurrenz von mehreren Programmen oder Programmteilen (Segmente, Seiten) im Hauptspeicher möglich. In Bild 2 ist ein Warteschlangenmodell angegeben, welches die verkehrstheoretisch wesentlichsten Teile eines Rechnersystems mit Multiprogramming und Paging wiedergibt.

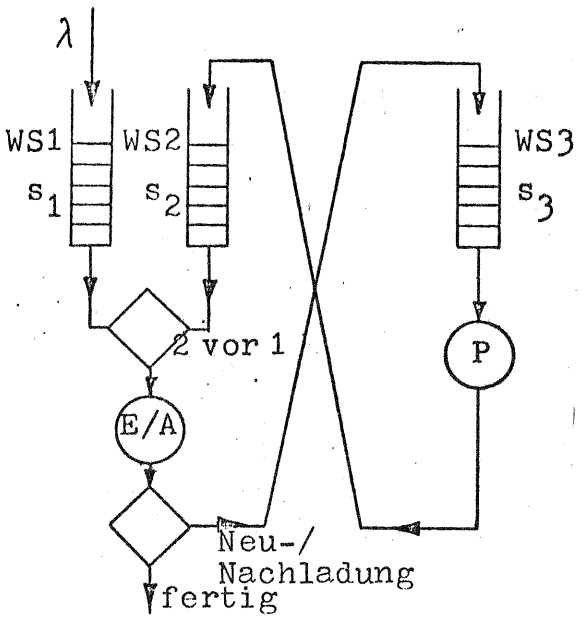


Bild 2. Multiprogramming-Rechnermodell

Neue Anforderungen (Programme) werden mit der Rate λ erzeugt und warten in einem Hintergrundspeicher (WS1), bis sie über den E/A-Kanal in den Hauptspeicher (WS3) mit einer Anfangsladung von Seiten transportiert werden. Der Prozessor P bearbeitet ein Programm bis zur Beendigung bzw. einer Unterbrechung infolge "page fault" und erzeugt - nach einer Verwaltungszeit - eine Anforderung an den E/A-Kanal, welche in WS2 warten kann bis zur Ausspeicherung des fertigen Programms bzw. Nachladung einer neuen Seite.

Als Betriebsstrategie müssen zunächst die Prioritätsregel bei der Zusammenführung (WS1, WS2) und die Verzweigungs-Bedingungen (nach E/A-Kanal) angegeben werden (vergl. Bild 2). Als Auswahlstrategien innerhalb der Warteschlangen kommen Prioritäten bzw. die normale Ankunftsreihenfolge in Frage. Die Ankunftsabstände neuer Programme seien z.B. negativ-exponentiell verteilt mit Ankunftsrate λ. Ein in der Praxis jedoch häufig zutreffender Fall ist die Annahme einer "gesättigten" Warteschlange WS1, wofür der Ankunftsprozess ohne Bedeutung ist. Die Belegungsdauern des E/A-Kanals sind ganze Vielfache einer Seitentransferzeit, die Prozessor-Belegungszeiten für ein ganzes Programm bzw. ein Programmteil sind i.a. hyperexponentiell verteilt.

Als interessierende Kenngrößen des Systems werden Mittelwerte und VF der Warte- und Verweilzeiten, Belastungen des Prozessors P, des E/A-Kanals und der einzelnen Wartespeicher sowie die mittlere Zahl von Unterbrechungen je Programm gemessen.

2.3.3 Netzknoten

Nachrichtennetze für den Fernsprech- und Datenverkehr sind nach einer Mischform aus reinen Stern- und Maschennetzen aufgebaut. Dadurch ist ein gewünschtes Ziel auf mehreren Wegen erreichbar und das Netz bei momentanen Engpässen durch Verkehrsspitzen bzw. Teilausfällen noch funktionsfähig. Zur wirtschaftlichen Ausnutzung der Übertragungsleitungen werden jedoch die Direktwege hoch belastet, während die Zweit- und Drittwege den "Überlaufverkehr" aufnehmen und deshalb weniger stark belastet werden dürfen.

In Bild 3 ist ein Ausschnitt aus einem Netz gezeigt, welches u.a. die Knoten i, j und k enthält. Der Verkehr von Knoten i nach j wird zunächst

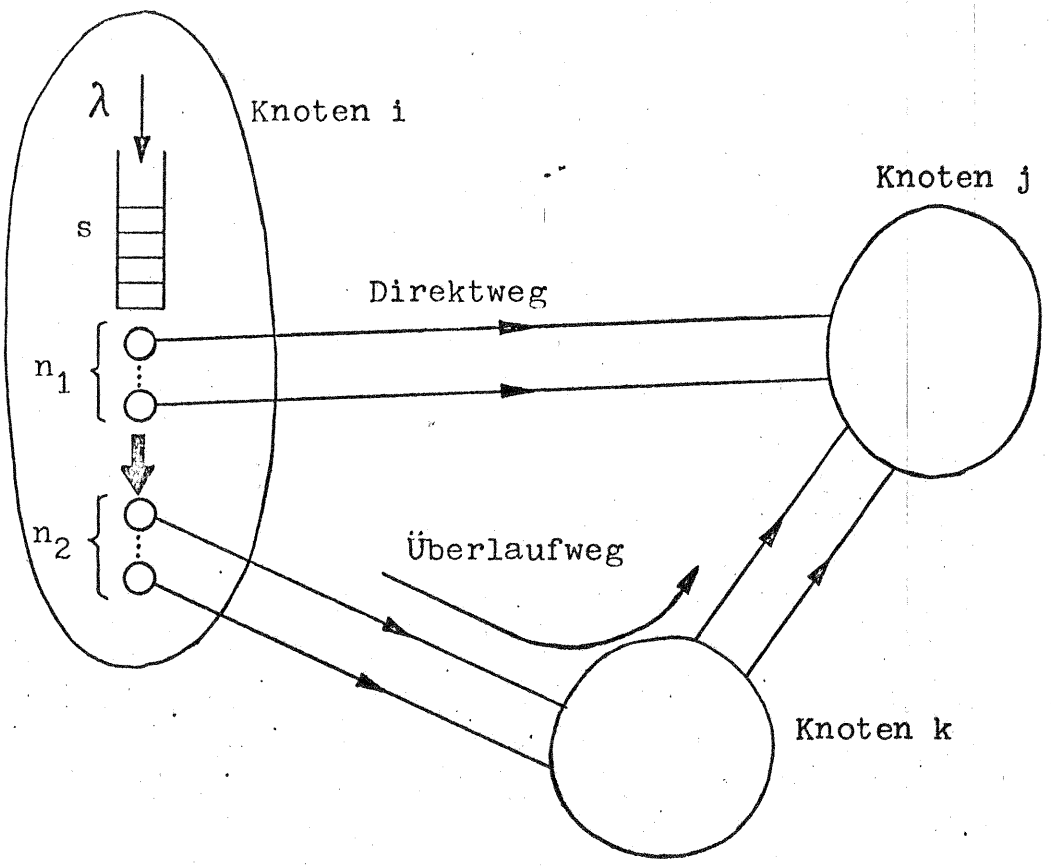


Bild 3. Netzknoten mit Direkt- und Überlaufweg

dem billigeren Direktweg mit n_1 Leitungen (Primärbündel) angeboten. Ist der Direktweg voll belegt, so läuft der Verkehr auf den teureren Überlaufweg mit n_2 Leitungen (Sekundärbündel) über und wird über Knoten k nach Knoten j geführt. Sind beide Wege belegt, so wird ein Platz im Wartespeicher (Kapazität s) belegt. Ist auch der Wartespeicher voll belegt, so wird eine ankommende Anforderung abgewiesen und geht verloren. Die Betriebsstrategien definieren den Überlauf vom Direktweg auf den Überlaufweg, die Auswahl von freien Leitungen innerhalb eines Bündels sowie die Auswahl wartender Anforderungen aus dem Wartespeicher. Die Ankunftsabstände seien z.B. negativ-exponentiell verteilt, die Belegungsdauern konstant (Datenverkehr).

Als Kenngrößen des Systems sind die Belastungen auf Direkt- und Überlaufweg, Warte- und Verlustwahrscheinlichkeit sowie Mittelwert und VF der Wartezeiten von Interesse.

3. SIMULATIONSVERFAHREN

3.1 Beschreibung des Beispiel-Modells

Mit den Elementen aus Abschnitt 2 lassen sich reale Systeme als Modelle nachbilden, wie dies durch die Beispiele in Abschnitt 2.3 angedeutet wurde. Im folgenden soll das Beispiel aus Abschnitt 2.3.3 näher betrachtet werden, um daran
- die Anwendung der Simulationstechnik zu beschreiben und
- Erfahrungen bei der Anwendung von Programmiersprachen zu zeigen (Abschnitt 5).

Bild 4 zeigt nochmals das Modell sowie eine Beschreibung, die sich auf die Struktur, die Betriebsstrategien und die Prozesse bezieht.

Ziel der Simulation ist es nun, dieses Modell so durch ein Simulationsprogramm auf einem Digitalrechner nachzubilden, daß die folgenden Meßgrößen als Ergebnisse gewonnen werden können:

- vor dem Wartespeicher:
  - In der Simulation realisierte Ankunftsrate der Anforderungen
  - Verlustwahrscheinlichkeit
- im Wartespeicher:
  - Wartewahrscheinlichkeit
  - Belastung
  - mittlere Wartezeit
  - Wartezeitverteilung
- in Primärbündel und Sekundärbündel jeweils:
  - Belastung
  - Zustandswahrscheinlichkeiten
  - Varianz der Anzahl gleichzeitig belegter Leitungen

3.2 Wahl des Simulationsverfahrens

Durch das Simulationsverfahren soll vor allem das d y n a m i s c h e Verhalten des Modells (der Ablauf) richtig abgebildet werden. Bild 5 zeigt dazu ein Ablaufdiagramm, welches erkennen läßt, daß es zwei Z e i t p u n k t e gibt, in denen eine Z u s t a n d s ä n d e r u n g stattfindet:

- Eine Anforderung kommt an
  - und belegt eine Leitung im Primärbündel
  - oder belegt eine Leitung im Sekundärbündel
  - oder belegt einen Warteplatz
  - (oder geht "verloren", was den Zustand des Modells nicht verändert)
- Eine Anforderung gibt eine Leitung frei
  - und eine wartende Anforderung rückt nach
  - oder die Leitung bleibt frei

Da der Ablauf des Geschehens im Modell durch diese Zustandsänderungen oder E r e i g n i s s e vollständig beschrieben werden kann, (wobei die Zeitspanne zwischen den einzelnen Zustandsänderungen durch Ankunfts- bzw. Belegungsprozess bestimmt wird,)genügt es, in der Simulation nur zu diesen d i s k r e t e n Zeitpunkten die Vorgänge im Modell nachzubilden (im Gegensatz zur kontinuierlichen Simulation z.B. auf einem Analogrechner). Wenn auch die Belegungsdauern negativ exponentiell verteilt wären, würde es zur Messung von Mittelwerten sogar genügen, nur die Wahrscheinlichkeiten für das Eintreten einer Zustandsänderung zu berücksichtigen (Monte-Carlo-Methode, Roulette-Methode,/1/). Im betrachteten Modell muß jedoch

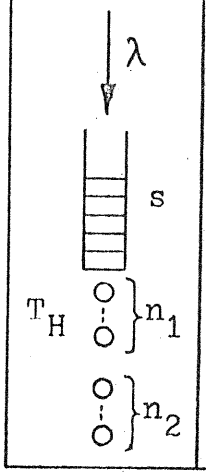
MODELL	MODELLBESCHREIBUNG		
	STRUKTUR	BETRIEBSSTRATEGIEN	PROZESSE
	Verkehrsquellen $q \to \infty$	Verlust, wenn alle Warteplätze belegt sind	Ankunftsrate λ Ankunftsabstände negativ-expon. verteilt
	Wartespeicher mit s Warteplätzen	Auswahl einer wartenden Anforderung nach FIFO	
	Primärbündel n_1 Leitungen	Absuchen sequentiell mit Überlauf von Primärbündel auf Sekundärbündel	Belegungsdauern im Primärbündel u. Sekundärbündel konstant $T_H = 1/\varepsilon$
	Sekundärbündel n_2 Leitungen		

Bild 4. Modellbeschreibung für das Simulationsbeispiel

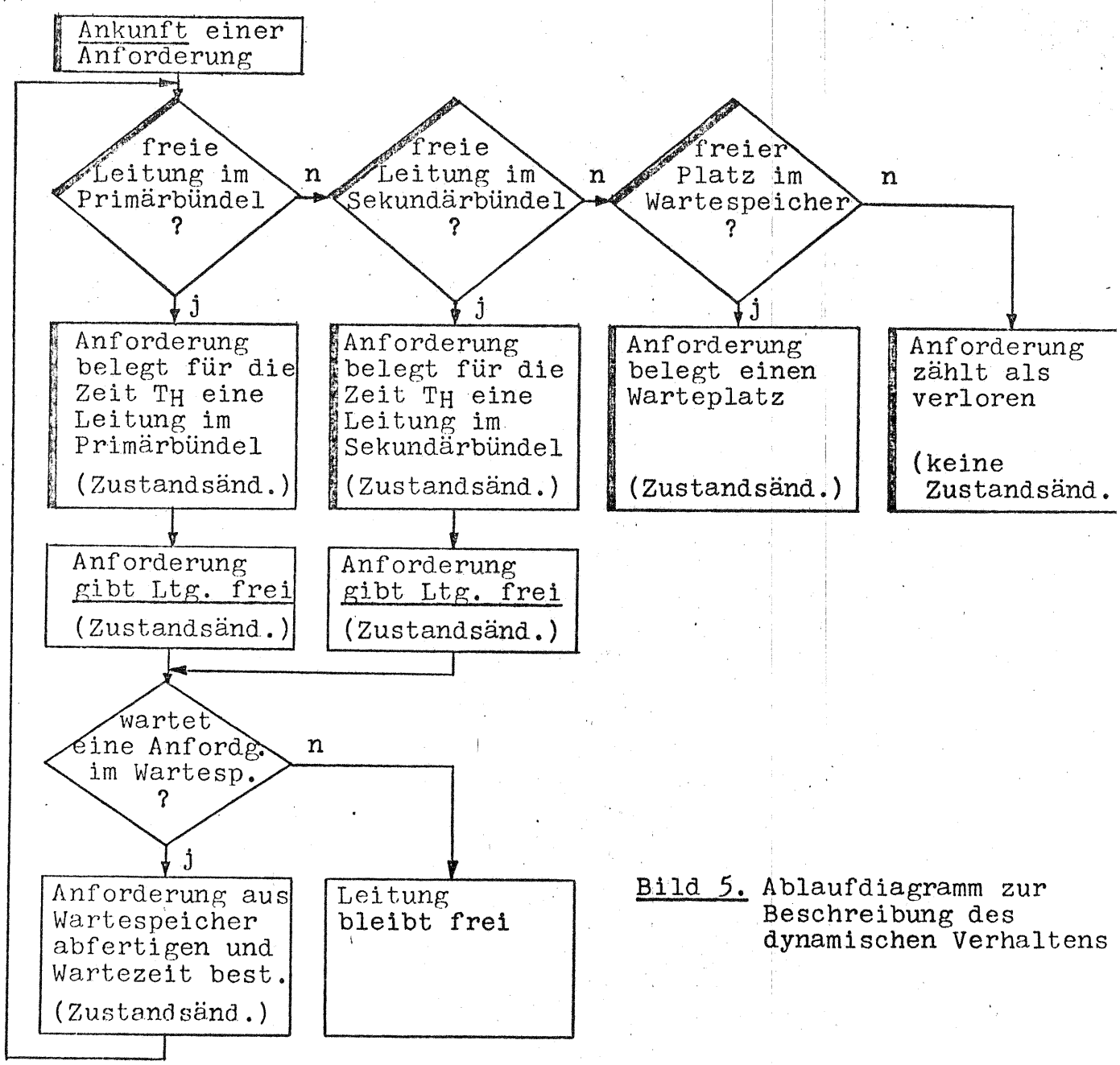


Bild 5. Ablaufdiagramm zur Beschreibung des dynamischen Verhaltens

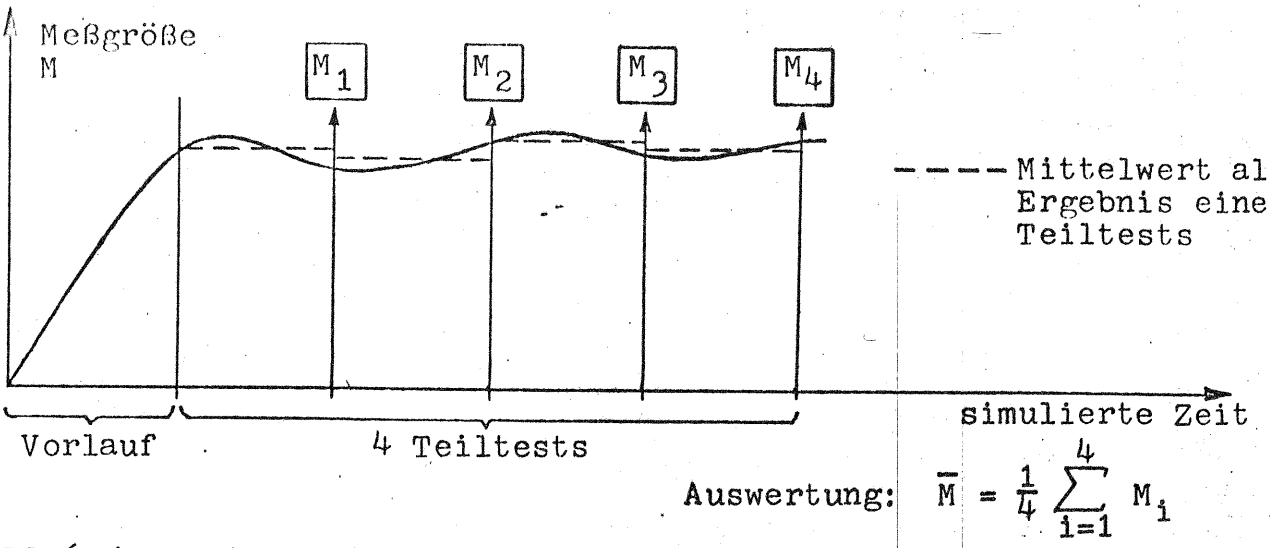


Bild 6. Auswertung einer Meßgröße

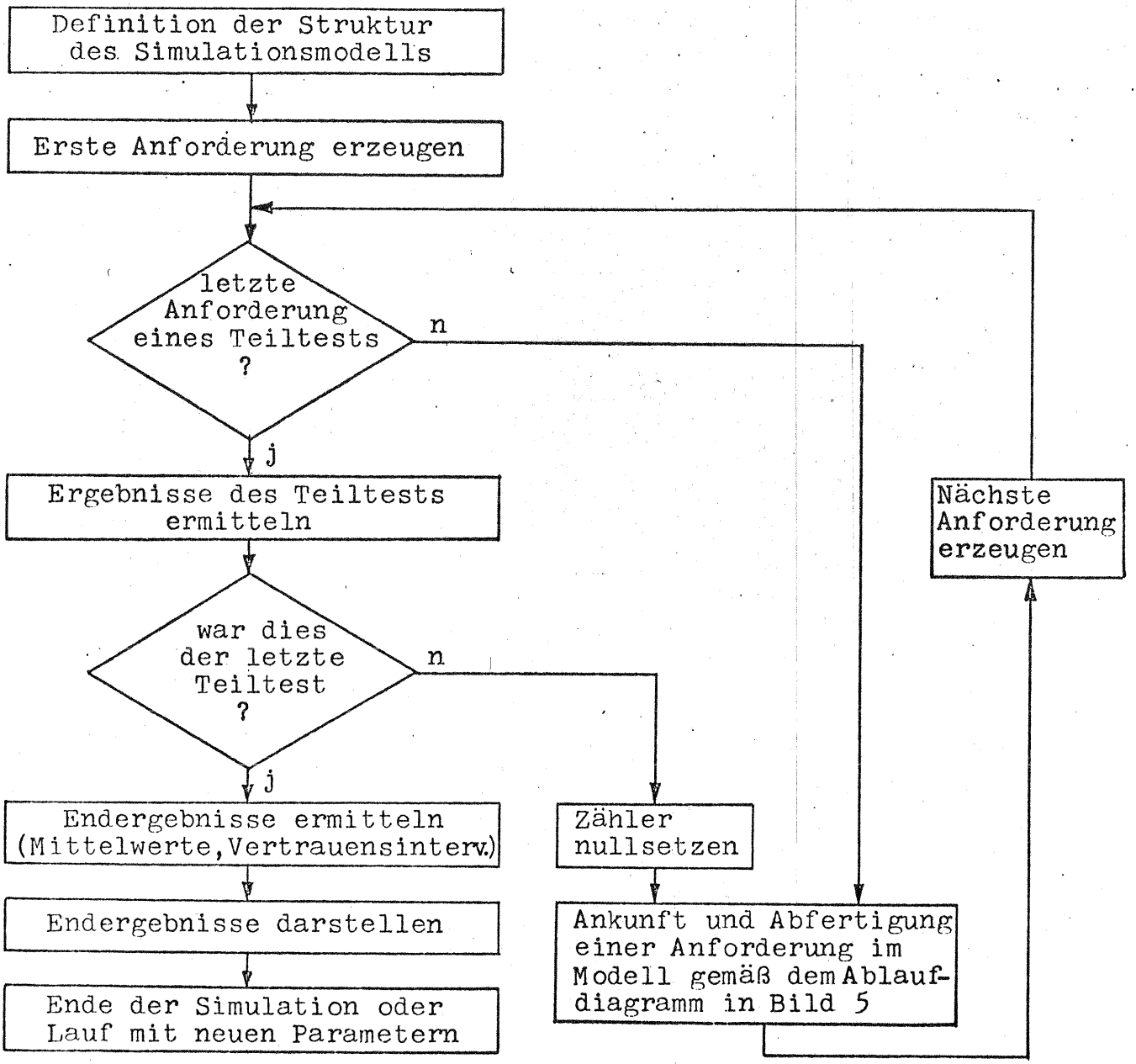


Bild 7. Ablaufdiagramm für das Rahmenprogramm

wegen der konstanten Belegungsdauern die z e i t t r e u e (event by event) Simulationsmethode /2/ angewandt werden, zumal außerdem noch individuelle Wartezeiten für die Berechnung der Wartezeitverteilung zu messen sind.

3.3 Durchführung und Auswertung der Simulation

Da die Zustandsänderungen im Modell durch die Ankunft und Abfertigung von Anforderungen vor sich gehen, wird sich erst nach einer gewissen Zeit (VORLAUF) ein stationärer Zustand im Modell einstellen, vgl. Bild 6 .Eine Aussage über die statistische Sicherheit, mit der das Endergebnis $\bar{M}$ (Mittelwert) einer Meßgröße ermittelt wird, läßt sich durch eine Unterteilung der Simulation in TEILTESTS machen. Aus den Teiltestergebnissen kann das Vertrauensintervall für die Meßgrößen errechnet werden (vgl./3/ bis /5/). Die Durchführung von VORLAUF, TEILTESTS und Auswertung erfolgt im sogenannten Rahmenprogramm (vgl. /6/,/7/), das in Bild 7 als Ablaufdiagramm dargestellt ist.

Im betrachteten Modell ist die Zahl der Quellen unendlich groß. Daher ist die Ankunftsrate der Anforderungen in jedem Augenblick gleich λ und der negativ-exponentiell verteilte Ankunftsabstand T_A ist mathematisch sehr einfach darstellbar als

$$T_A = - \frac{1}{\lambda} \ln(z)$$

mit der Zufallsvariablen z , die zwischen 0 und 1 gleichverteilt ist. Viele Beispiele in der Nachrichtenverkehrstheorie weisen jedoch eine endliche Zahl von Quellen auf. Die hierbei in der Simulation auftretenden Probleme werden ausführlich in /8/ diskutiert.

4. GRUNDSÄTZLICHE FORDERUNGEN AN SIMULATIONSSPRACHEN FÜR DIE NACHRICHTENVERKEHRSTHEORIE

In den Abschnitten 2 und 3 wurden die Problemstellungen und die Simulationsmethode erläutert, wie sie für die Nachrichtenverkehrstheorie typisch sind. Daraus ergeben sich die im folgenden aufgezählten Forderungen an eine "ideale" Simulationssprache (IS) für die Durchführung einer diskreten, zeittreuen Simulation gemäß Abschnitt 3.2.

4.1 Modellbeschreibung

Die Elemente der IS sollen den typischen Elementen (Abschnitt 2.2) der Modelle aus der Nachrichtenverkehrstheorie entsprechen und sich bequem miteinander verknüpfen lassen (z.B. bei Zwischenleitungsanordnungen). Variablennamen kann der Benutzer beliebig festlegen. M a k r o b e f e h l e sollen die Nachbildung der Betriebsstrategien sowie der Prozesse gestatten. Die simulierte Zeitachse ist durch ein selbständiges Zeitkontroll-Programm zu verwalten (Überwachung der Ereignisfolge), wobei Zeitgrößen als Gleitkommazahlen gelten.

4.2 Logische und arithmetische Operationen, Statistik

In der IS sind ganze Zahlen, Gleitkommazahlen und Boolesche Ausdrücke als Variable zuzulassen. In einer Funktions-Bibliothek sind die häufig benutzten Funktionen (z.B. ln, Wurzelfunktion) bereitzustellen. Für die gebräuchlichsten Verteilungsfunktionen sollen Prozeduren zur Verfügung stehen (vgl. Abschnitt 2.2.3). Die in der Simulation gewonnenen Meßdaten sind ohne großen Programmieraufwand statistisch auszuwerten (Mittelwerte, Vertrauensintervalle, Histogramme).

4.3 Eingabe, Ausgabe

Die Eingabe soll ohne ein vorgeschriebenes Format erfolgen. Die Anfangswerte und die Struktur des Modells seien einfach zu ändern. Für die Gestaltung der Drucker-Ausgabe (oft Tabellen) sollten bequeme Anweisungen zur Verfügung stehen und Ausgabebefehle für das Ablegen von Daten auf externen Speichern (Band, Platte) sollten nicht fehlen.

4.4 Testmöglichkeiten

Syntaktische und logische Fehler sowie eine falsche Reihenfolge im Ablauf der Ereignisse sind durch "Momentaufnahmen" (Ausdrucken der Werte aller Variablen) oder Prüfroutinen sichtbar zu machen. Der Compiler soll über eine detaillierte Fehlerdiagnose verfügen, und im Benutzerhandbuch ist die Abhilfe im Fehlerfall allgemeinverständlich zu erläutern.

4.5 Sonstiges

Die IS soll leicht erlernbar sein, sparsam mit Rechenzeit und Speicherplatz umgehen und eine übersichtliche Programmierung ermöglichen, welche gleichzeitig eine Modell-gerechte Formulierung des Simulationsproblems erlaubt.

5. ERFAHRUNGEN MIT 4 PROGRAMMIERSPRACHEN

Im folgenden Abschnitt wird gezeigt, welche Erfahrungen bei der Erstellung eines zeittreuen Simulationsprogrammes mit den Programmiersprachen FORTRAN IV , GPSS/360 , SIMSCRIPT I.5 und SIMULA 67 gemacht wurden. Als Simulationsmodell diente das Beispiel nach Bild 4 . Es sollen dabei weder die Simulationsprogramme (vgl. /9/ bis /12/) noch die Programmiersprachen (vgl. /13/ bis /21/) im einzelnen ausführlich erklärt werden. Für einen grundlegenden Vergleich von Programmiersprachen sei auf die Literaturstellen /22/ bis /32/ verwiesen.

5.1 Allgemeines über FORTRAN, GPSS, SIMSCRIPT, SIMULA

Während FORTRAN (wie ALGOL) eine universelle problemorientierte Programmiersprache darstellt, zählen die Simulationssprachen GPSS, SIMSCRIPT und SIMULA zu den speziellen problemorientierten Programmiersprachen und erfüllen in einigen Teilen die in Abschnitt 4 gestellten Forderungen. Insbesondere besitzen die drei letztgenannten Programmiersprachen einige Sprachelemente zur Beschreibung typischer Modellelemente: In Tabelle 4 ist eine Zusammenstellung einiger wichtiger Merkmale für die 3 Simulationssprachen angegeben. Bei der Beschreibung der Anwendung der einzelnen Sprachen werden diese Merkmale näher erläutert (Abschnitte 5.3 bis 5.5) .

Da nicht nur die Leistungsfähigkeit, sondern auch die Verfügbarkeit einer Programmiersprache über ihren Einsatz entscheidet, sind in Tabelle 5 einige Compiler-Hersteller für GPSS, SIMSCRIPT und SIMULA genannt. Ein FORTRAN-Compiler ist sicherlich für nahezu jeden Digitalrechner erhältlich. Für das Simulationsbeispiel wurden Compiler der Anlagen IBM 360/40 (GPSS/360) sowie CD 6600 (FORTRAN IV, SIMSCRIPT I.5 , SIMULA67) verwendet.

Tabelle 4. Einige Merkmale von GPSS, SIMSCRIPT, SIMULA

M E R K M A L	G P S S	S I M S C R I P T	S I M U L A
- bewegliche Einheit (z.B. Anforderung)	TRANSACTION	TEMPORARY ENTITY	PROCESS } Exemplar einer ACTIVITY
- Bedienungseinheit (z.B. Leitungsbündel)	FACILITY (1 Platz) STORAGE (>1 Platz)	PERMANENT ENTITY	PROCESS } Exemplar einer ACTIVITY
- Eigenschaft einer Einheit (z.B. Zeitpunkt, zu dem eine Anforderung in einen Wartespeicher gelangt)	PARAMETER	ATTRIBUTE	ATTRIBUTE
- Gruppe, in die Einheiten eingeordnet werden können (z.B. Wartespeicher)	CHAIN	SET	SET
- Ereignistyp	abhängig vom Block, den die TRANSACTION durchläuft	festgelegt im EVENT, einem abgeschlossenen Programmteil	festgelegt durch Operationsregeln im PROCESS, die in seiner aktiven Phase ausgeführt werden
- Zeitkontrolle	blockierte oder aktive TRANSACTION wird in internen CHAINS organisiert	EVENTS werden durch Anweisungen in den internen Kalender eingetragen	SEQUENCING SET zur Organisation der aktiven Phasen der PROCESSES

G P S S			S I M S C R I P T			S I M U L A	
Version/Herst./Serie			Version/Herst./Serie			Hersteller /	Serie
II	UNIVAC	1107 1108	?	PHILCO-FORD	210 211 212	BURROUGHS	B5500
III	CONTROL DATA CORP.	3600	I	IBM	7090 7094	CONTROL DATA CORP.	6400 6500 6600
III	IBM	7090 7094 7040 7044	I.5	CONTROL DATA CORP.	3600 3800 6400 6500 6600	IBM	360/ 370/
/360	IBM	360/ 370/	I.5	GENERAL ELECTRIC	625 635	UNIVAC	1107 1108
			I.5	IBM	360/		
			I.5	RCA Info-Systems	Spectra 70 ab Mod. 45		
			I.5	UNIVAC	490 1107 1108		
			II	IBM	360/		

Tabelle 5. Compiler für GPSS, SIMSCRIPT, SIMULA (nach /32/)

5.2 Anwendung von FORTRAN IV auf das Simulationsbeispiel

Da in FORTRAN keine Sprachelemente vorhanden sind, welche typische Modellelemente (Wartespeicher, Bedienungseinheiten) beschreiben, muß das Modell mit Hilfe von indizierten Variablen nachgebildet werden, die z.B. den Belegungszustand und die Belegungsstatistik festhalten. Die negativ-exponentiell verteilten Ankunftsabstände werden durch gleichverteilte Pseudozufallszahlen (vgl. Abschnitt 3.3) nachgebildet, die bei FORTRAN durch eine Bibliotheksfunktion aufgerufen werden können. Der Zeitablauf wird über ein Feld gesteuert, in welches jeweils das als nächstes anstehende Ereignis eingetragen wird. Die typischen Vorgänge im Verlauf der Simulation (Suchen einer freien Leitung, Belegen eines Warteplatzes, Auswerten eines Teiltests) können durch SUBROUTINES und FUNCTIONS nachgebildet werden, wodurch das Programm eine gewisse Übersichtlichkeit erhält. Im wesentlichen entspricht das Ablaufdiagramm des FORTRAN-Programms den in Bild 5 und Bild 7 gezeigten Ablaufdiagrammen. Die Beschreibung der Vorgänge in der Simulation durch jeweils mehrere Anweisungen bildet allerdings einen Nachteil für die Lesbarkeit des Programms.

Die Eingabe erfolgt bei FORTRAN IV formatgebunden, die Gestaltung der Ausgabe (z.B. Tabellen) ist für die Zwecke der Simulation etwas unhandlich. Logische und syntaktische Fehler werden ausführlich analysiert. Die Kontrolle des zeitlichen Ablaufs ist durch Druckbefehle möglich. FORTRAN IV ist gut dokumentiert und leicht erlernbar. Programmlaufzeit und Speicherplatzbedarf : siehe Abschnitt 5.6 .

5.3 Anwendung von GPSS/360 auf das Simulationsbeispiel

5.3.1 Modellbeschreibung bei GPSS/360

Die Simulationssprache GPSS/360 bildet das Modell und die darin ablaufenden Ereignisse durch ein Ablaufdiagramm nach, dessen Blöcke jeweils mit einem Makrobefehl beschrieben werden. Jeder Block ist durch eine Reihe von Parametern in seiner Wirkung festgelegt. Diese Wirkung übt der Block auf die beweglichen Teile des Modells, die TRANSACTIONS aus, sobald sie ihn durchlaufen bzw. in ihm verweilen (z.B. eine Anforderung belegt eine Leitung). Jede TRANSACTION kann sich zu einem bestimmten Zeitpunkt nur an e i n e r Stelle im Ablaufdiagramm aufhalten. TRANSACTIONS können in Wartespeicher eingereiht werden und unter vorschreibbaren Bedingungen diesen wieder verlassen. Eine TRANSACTION ist durch maximal 100 PARAMETER beschreibbar.

Der Ablauf der Ereignisse im Modell hängt bei GPSS/360 vom Zustand aller im Modell befindlichen TRANSACTIONS ab, wobei stets eine bestimmte TRANSACTION soweit durch das Ablaufdiagramm transportiert wird, bis sie entweder einen ADVANCE-Block erreicht (Belegungsdauer) oder z.B. aufgrund eines bereits belegt angetroffenen FACILITY-Blocks (Bedienungseinheit) am Weiterkommen gehindert wird. Dann sucht die Programmsteuerung die nächste bewegbare TRANSACTION im Modell.

Einige Sprachelemente, die im Simulationsbeispiel Verwendung fanden:

GENERATE-Block -erzeugt die Anforderungen (TRANSACTIONS) in zeitlichem Abstand entsprechend einer vorgebbaren FUNCTION (Wertepaare, welche die Stützstellen beschreiben.

STORAGE -mehrere Bedienungseinheiten, 1 Leitungsbündel .

QUEUE -Wartespeicher, in dem die vor einem STORAGE wartenden Anforderungen Platz finden.

TEST-Block -Abfrage.

TERMINATE -Löschen einer Anforderung nach Beendigung ihres Durchlaufs.

5.3.2 Logische und arithmetische Operationen, Statistik bei GPSS/360

Bezüglich der arithmetischen Operationen entstehen für das Simulationsbeispiel dadurch Schwierigkeiten, daß bei GPSS/360 Variable eine Boolesche Größe darstellen oder nur ganzzahlige Werte annehmen können. Dies bedeutet, daß alle Ergebnisse, die sich auf Wahrscheinlichkeiten beziehen, bei der Berechnung z.B. mit dem Faktor 10^5 multipliziert werden müssen. Hierdurch darf allerdings (bei Zwischenrechnungen) nicht die o b e r e Zahlenbereichsgrenze von $2^{31}-1$ überschritten werden. Ein weiterer Nachteil liegt darin, daß wichtige Bibliotheksfunktionen fehlen: Für die Berechnung des Vertrauensintervalls wird die Wurzelfunktion benötigt, die bei GPSS/360 durch Stützstellen einzugeben ist (Polygonzug) . Sehr vorteilhaft ist die automatische Erstellung von statistischen Daten bei Bedienungseinheiten und Wartespeichern. Z.B. werden für einen Wartespeicher die folgenden 8 Daten ermittelt:

- Belastung
- Gesamtzahl der TRANSACTIONS, die gewartet haben
- Gesamtzahl der TRANSACTIONS, die nicht warten mußten
- Wahrscheinlichkeit, daß nicht gewartet werden muß
- mittl. Wartezeit aller durchgelaufenen TRANSACTIONS
- mittl. Wartezeit aller wartenden TRANSACTIONS
- momentante Anzahl von wartenden TRANSACTIONS
- maximale Anzahl von wartenden TRANSACTIONS

5.3.3 Eingabe, Ausgabe bei GPSS/360

Die wesentlichen Eingabedaten zur Beschreibung des Simulationsbeispiels sind im Simulationsprogramm in aufeinanderfogenden Karten in einem gesonderten Programmteil (MACRO) zusammengefaßt worden. Für die Ausgabe auf einem Drucker stehen bei GPSS/360 zwei Typen von Befehlen zur Verfügung:
- Auswahl, Betiteln und Kommentieren von Statistiken
- graphische Ausgabe (Diagramme)

5.3.4 Testmöglichkeiten bei GPSS/360

Die automatisch erstellten Statistiken erleichtern das Austesten wesentlich. Die meisten Programmierfehler werden außerdem durch den Compiler analysiert und durch eine Codenummer gekennzeichnet.

5.3.5 Sonstiges bei GPSS/360

Der kompakte übersichtliche Aufbau von GPSS/360 trägt zur schnellen Erlernbarkeit der Sprache bei. Der Übergang vom Ablaufdiagramm zum Programm ist direkt durchführbar. Die Lesbarkeit des Programms leidet allerdings ein wenig darunter, daß der Programmierer an die vorgeschriebenen Parameternamen gebunden ist und nicht eigene Parameterbezeichnungen Modell-bezogen wählen kann. Programmlaufzeit und Speicherplatzbedarf: siehe Abschnitt 5.6 .

5.4 Anwendung von SIMSCRIPT I.5 auf das Simulationsbeispiel

5.4.1 Modellbeschreibung bei SIMSCRIPT I.5

Die Struktur dieser Simulationssprache erfordert eine klare Aufteilung des Modells in einen statischen Teil (definiert durch PERMANENT ENTITIES und deren ATTRIBUTES) und einen dynamischen Teil, d.h. Programmabschnitte, welche die Durchführung von Zustandsänderungen gestatten (EVENTS). Für das Simulationsbeispiel werden gemäß Abschnitt 3.2 zwei EVENTS benötigt:
- Ankunft einer Anforderung mit der Abfrage, ob sie sofort abgefertigt werden kann oder wartet oder verloren geht;
- Freigabe einer Leitung mit der Abfrage, ob eine Anforderung wartet und diese Leitung aufs neue belegt.

Zur Organisation der simulierten Zeitachse steht ein "innerer Kalender" zur Verfügung, in den der Zeitpunkt für das Stattfinden eines Ereignisses entweder schon zu Beginn der Simulation (EXOGENOUS EVENT) oder erst im Verlauf der Simulation in Abhängigkeit von gewissen Bedingungen eingetragen wird (ENDOGENOUS EVENT). Das jeweilige Ereignis findet statt, indem die Anweisungen des betreffenden Programmteils ausgeführt werden. Als "bewegliche" Elemte des Modells werden TEMPORARY ENTITIES erzeugt, die durch TEMPORARY ATTRIBUTES beschrieben werden und sich in Gruppen (SETS) einordnen lassen (FIFO, LIFO oder abhängig vom Zahlenwert eines TEMPORARY ATTRIBUTE). Die Definition aller Größen (außer der lokalen Variablen) und eine Aufteilung in TEMPORARY VARIABLES, PERMANENT VARIABLES, SETS und FUNCTIONS erfolgt anhand eines DEFINITION-Formulars, in dem auch Angaben über den Speicherbedarf jeder Variablen (Ganzworte, Halbworte, Drittelworte) zu machen sind.

Einige Sprachelemente, die im Simulationsbeispiel Verwendung fanden:

CREATE ruf - erzeugt eine Anforderung (TEMPORARY ENTITY), die unter dem Namen "ruf" angesprochen werden kann.

CAUSE ankft AT 2.0 - veranlaßt, daß das Ereignis "Ankunft einer Anforderung" für den Zeitpunkt 2.0 in den inneren Kalender eingetragen wird.

FILE ruf IN wart - ordnet die TEMPORARY ENTITY "ruf" in den Wartespeicher "wart" ein.

DESTROY ruf - zerstört die TEMPORARY ENTITY "ruf" und deren TEMPORARY ATTRIBUTES. Hierdurch werden die betreffenden Speicherzellen frei für andere, neue TEMPORARY ENTITIES (dynamische Speicherplatzverwaltung).

5.4.2 Logische und arithm. Operationen, Statistik bei SIMSCRIPT I.5

Die Verwandtschaft von SIMSCRIPT und FORTRAN zeigt sich darin, daß die Möglichkeiten für arithmetische Operationen in beiden Sprachen gleich sind, allerdings mit der Ausnahme, daß SIMSCRIPT keine Boolesche Variablen kennt. Zur Herstellung von statistischen Daten gibt es bei SIMSCRIPT Anweisungen, welche die laufende Integration von Meßwerten über der Zeit sowie die Bildung von z.B. Mittelwert, Varianz oder Standardabweichung aus einer Werteserie erlauben. Als Funktionen stehen die in der FORTRAN-Unterprogramm-Bibliothek genannten zur Verfügung. Zusätzlich können eigene FUNCTIONS definiert oder Funktionsverläufe durch Wertepaare (Stützstellen) eingegeben werden.

5.4.3 Eingabe, Ausgabe bei SIMSCRIPT I.5

Die Werte aller PERMANENT VARIABLES zu Beginn der Simulation und damit auch der Anfangszustand des Modells werden in einem INITIALIZATION-Formular festgelegt. Das vorgeschriebene Format erfordert dabei einen erhöhten Aufwand beim Erstellen der Eingabedaten. Die Gestaltung der Ausgabe auf einem Drucker wird in "reports" festgelegt, die eine übersichtliche Formulierung der Druckbefehle gestatten. Allerdings benötigen die vom Compiler erzeugten zugehörigen Hilfsprogramme einen verhältnismäßig großen Speicherplatz. Schreibbefehle für den Transport von Daten auf externe Speicher sind vorhanden.

5.4.4 Testmöglichkeiten bei SIMSCRIPT I.5

Da das Simulationsprogramm aus selbständigen Teilen (EVENTS, SUBROUTINES) besteht, kann kein geschlossenes Ablaufdiagramm für den gesamten zeitlichen Simulationsablauf erstellt werden. Durch Druckanweisungen innerhalb eines EVENT läßt sich jedoch der richtige zeitliche Ablauf überprüfen. Der Compiler druckt beim Übersetzen ausführliche Fehlerangaben aus. Die Fehlerdiagnose während des Simulationslaufs war allerdings im vorliegenden Fall nicht immer zufriedenstellend.

5.4.5 Sonstiges bei SIMSCRIPT I.5

Für FORTRAN-Kenner ist SIMSCRIPT leicht erlernbar. Die freie Namensgebung macht das Programm gut lesbar. In der neueren Version SIMSCRIPT II /15/ sind einige Nachteile, vor allem bezüglich der streng formalen Dateneingabe, behoben. Programmlaufzeit und Speicherplatzbedarf : siehe Abschnitt 5.6 .

5.5 Anwendung von SIMULA 67 auf das Simulationsbeispiel

5.5.1 Modellbeschreibung bei SIMULA 67

Die Elemente im Simulationsmodell, welche während der Simulation Veränderungen erfahren o d e r Veränderungen auslösen, werden durch PROCESSES nachgebildet, die individuell durch ATTRIBUTES gekennzeichnet sind und deren Verhalten durch eine Folge von zugehörigen Anweisungen (Operationsregeln) festgelegt ist. Dabei bewirken die Anweisungen unter anderem, daß der PROCESS sich in einer aktiven Phase (entsprechend einem stattfindenden Ereignis,EVENT) oder in einer passiven Phase befindet. Eine Ablaufsteuerung (SEQUENCING SET) besorgt die zeitliche Abfolge der aktiven Phase bei allen im System befindlichen PROCESSES, wobei immer nur 1 PROCESS aktiv sein kann. PROCESSES mit gleicher Wirkungsweise (d.h. gleicher Folge von Operationsregeln, aber unterschiedlichen ATTRIBUTE-Werten) sind durch ein und denselben Programmabschnitt (ACTIVITY) beschrieben. Die Variable ELEMENT ermöglicht einen Bezug auf einen PROCESS. ELEMENTS können in Gruppen (SETS) eingeordnet werden. Die Vereinbarung der ACTIVITIES,ATTRIBUTES etc. erfolgt (wie bei ALGOL) in einem Vereinbarungsteil jeweils zu Beginn der ineinander verschachtelten Blöcke.

Einige Sprachelemente, die im Simulationsbeispiel Verwendung fanden:

'ACTIVATE''NEW' ruf 'AT' 2.0	- veranlaßt den Aufruf des PROCESS "ruf" (durch 'NEW' ruf) und seine Aktivierung,d.h. die Ausführung seiner Operationsregeln, zum Zeitpunkt 2.0
CURRENT.INTO(wart)	- das ELEMENT des gerade aktiven PROCESS (repräsentiert eine Anforderung) wird in den Wartespeicher gebracht.

Ein individueller PROCESS wird beendet, d.h. Speicherplatz wird wieder frei, sobald in der Ausführung der ihn beschreibenden Operationsregeln die Anweisung 'END' erreicht wird.

5.5.2 Logische und arithm. Operationen, Statistik bei SIMULA 67

Die Entwicklung von SIMULA aus ALGOL 60 ist unter anderem daran erkennbar, daß beide Sprachen dieselben logischen und arithmetischen Operationen aufweisen und die Syntax von ALGOL 60 fast vollständig in SIMULA übernommen wurde. Eine Vielzahl von Verteilungsfunktionen kann in SIMULA 67 direkt über Prozedurnamen aufgerufen werden, wodurch die Nachbildung von Prozessen sehr erleichtert wird. Zur Erstellung statistischer Daten stehen Prozeduren zur Verfügung, welche z.B. Werte über der Zeit integrieren oder Histogramme errechnen.

5.5.3 Eingabe,Ausgabe bei SIMULA 67

Die Eingabe erfolgt völlig formatfrei: einzulesende Daten müssen nur durch mindestens eine Leerstelle getrennt sein. Für die Gestaltung der Ausgabe steht eine große Zahl von Anweisungen zur Verfügung (bequeme Tabellenausgabe).

5.5.4 Testmöglichkeiten bei SIMULA 67

Sowohl während der Übersetzung als auch während des Simulationslaufs werden Fehler ausführlich analysiert und ausgedruckt. Der richtige zeitliche Ablauf kann mit Druckbefehlen an den entscheidenden Programmstellen kontrolliert werden.

5.5.5 Sonstiges bei SIMULA 67

Für ALGOL-Kenner ist SIMULA relativ bequem erlernbar, wobei das Verständnis des Grundkonzeptes dieser Sprache zunächst Schwierigkeiten bereiten kann. Natürlich muß berücksichtigt werden, daß bei SIMULA 67 neben einer Anwendung als Simulationssprache allgemein eine handliche Listenverarbeitung möglich ist, wodurch die Sprache im Ganzen umfangreich wird. Das erstellte Simulationsprogramm ist durch die freie Wahl der Variablennamen und die Blockstruktur der Sprache gut lesbar. Programmlaufzeit und Speicherplatzbedarf: siehe nächster Abschnitt .

5.6 Vergleichsresultate

Zusammenfassend kann bezüglich einer Eignung für Simulationsprobleme in der Nachrichtenverkehrstheorie festgestellt werden, daß die drei Simulationssprachen GPSS, SIMSCRIPT und SIMULA folgendermaßen zu beurteilen sind:

- Strukturelemente sowie der Ablauf der Vorgänge im Modell lassen sich sehr bequem in GPSS nachbilden (direkte Umsetzung des Ablaufdiagramms).
- Betriebsstrategien sind vor allem in SIMSCRIPT günstig zu verwirklichen.
- Prozesse können nur einfach nachgebildet werden, wenn die häufigsten Verteilungsfunktionen zur Verfügung stehen. Der Mangel an Bibliotheksfunktionen bei GPSS ist besonders ungünstig, während SIMULA vorbildlich eine ganze Reihe von Verteilungsfunktionen bereitstellt.
- Die Erstellung statistischer Daten erfolgt bei GPSS automatisch (erhöhter Rechenzeitbedarf), doch können Variable nur ganzzahlige Werte annehmen. SIMSCRIPT und SIMULA stellen ausreichende Hilfsmittel zur Statistikerstellung bereit.
- Für die Eingabe von Daten und Anfangsbedingungen ist die formatfreie Form bei SIMULA besonders hilfreich.
- Alle drei Simulationssprachen verfügen, soweit die im Simulationsbeispiel benutzten Compiler ein solches Urteil zulassen, über eine ausreichende Fehlerdiagnose.
- Die Ausgabe gestaltet sich bei SIMSCRIPT durch report-Formulare sehr einfach und übersichtlich.
- Der Lernaufwand nimmt von GPSS über SIMSCRIPT bis SIMULA zu, doch hängt dies auch vom jeweils zur Verfügung stehenden Handbuch ab.

Zur Feststellung von Programmlaufzeit und Speicherplatzbedarf wurden die in FORTRAN IV, GPSS/360, SIMSCRIPT und SIMULA erstellten Programme mit denselben Eingabeparametern versehen und damit jeweils ein

Simulationslauf durchgeführt. Bild 6 zeigt die Ergebnisse. Der Vergleich mit GPSS/360 ist nicht ganz zutreffend, da hierbei ein anderer Rechner (IBM 360/40) als bei den übrigen drei Programmen eingesetzt wurde. Neben der Wahl des Compilers beeinflussen selbstverständlich auch noch andere Gegebenheiten grundsätzlich das Vergleichsergebnis: die Geschicklichkeit und Erfahrung des Programmierers die Komplexität des Modells, der Umfang der gewünschten statistischen Auswertung und die Anforderungen an die Ausgabe der Ergebnisse.

PROGRAMMIER-SPRACHE	SPEICHER-PLATZ (Worte à 60 bit)	ÜBERSETZUNGS-ZEIT (sec)	ZENTRAL-RECHNER-ZEIT (sec)	E/A-ZEIT (sec)	LOCH-KARTEN
FORTRAN IV	14 000	2.0	9.7	13.1	450
GPSS/360	30 000	79.2	2 049.5		340
SIMSCRIPT	33 000	48.5	34.2	58.5	350
SIMULA	17 700	6.3	379.4	19.5	360

Bild 8. Vergleichswerte für das Simulationsbeispiel
$s = 5$, $n_1 = 3$, $n_2 = 2$, VORLAUF: 500 Anforderungen
10 TEILTESTS: je 5000 Anforderungen

Unter Berücksichtigung aller bisherigen Erfahrungen, die naturgemäß beschränkt sein müssen, ergibt sich der Eindruck, daß für viele Probleme der Nachrichtenverkehrstheorie zur Zeit SIMSCRIPT und SIMULA als geeignete Simulationssprachen betrachtet werden können.

LITERATURVERZEICHNIS

/1/ KOSTEN, L.: On the Measurement of Congestion Quantities by Means of Fictitious Traffic. HET PTT-Bedrijf (1948/49), 15-25

/2/ NEOVIUS, G.: Artificial Traffic Trials Using Digital Computers. Ericsson Technics 11(1955), 279-291

/3/ LOTZE, A.: Über die statistische Sicherheit von Verkehrsmessungen. NTZ 11(1958)1, 5-7

/4/ SCHMETTERER, L.: Einführung in die mathematische Statistik. Springer-Verlag, Wien, New York, 2.Aufl.1966

/5/ KÜMMERLE, K.: Ein Vorschlag zur Berechnung der Vertrauensintervalle bei Verkehrstests. A.E.Ü.23(1969)10, 507-511

/6/ HUBER, M., WAGNER, W.: Simulation von Nachrichtenvermittlungssystemen. Aus: Nicht-numerische Informationsverarbeitung, Herausgeber: R. Gunzenhäuser, Springer-Verlag, Wien, New York, 1968

/7/ WAGNER, H. DIETRICH, G.: Bestimmung der Verkehrsleistung von Wartesystemen durch künstlichen Fernsprechverkehr NTZ 17(1964)6, 273-279

/8/ WEISSCHUH, H.: Theoretical Aspects of Finite Source Input Process Simulation. 7th International Teletraffic Congress, Stockholm, 1973

/9/ RAKOTOMANGA, D.: Simulationsprogramm in FORTRAN für ein Warte-Verlustsystem mit Überlauf. Monografie, Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart, 1970

/10/ THIEL, K.-H.: Simulationssprache GPSS. Monografie, Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart, 1970

/11/ PFEIFFER, H.: Nachbildung der Datenübertragung zwischen zwei Vermittlungsrechnern mit Hilfe der Simulationssprache SIMSCRIPT, Teil 1. Monografie, Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart, 1970

/12/ DITZEL, D.: Simulationssprache SIMULA. Monografie, Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart, 1972

/13/ IBM: General Purpose Simulation System/ 360 User's Manual. IBM Nr. H20-0326-3, 1968

/14/ MARKOWITZ, H., HAUSNER, B., KARR, H.: SIMSCRIPT: A Simulation Programming Language. The Rand Corporation, Santa Monica, Calif., RM-3310, Nov. 1962

/15/ KIVIAT, P.J., VILLANUEVA, R., MARKOWITZ, H.: The SIMSCRIPT II Programming Language. Prentice-Hall, Inc., New Jersey, 1968

/16/ KAMPE, G.: SIMSCRIPT. Vieweg-Verlag, Braunschweig, 1971

/17/ RAEHMEL, Ch.-A.: SIMSCRIPT-eine Simulationssprache. Computerpraxis 5(1972)11, 321-327

/18/ DAHL, O.J., NYGAARD, K.: SIMULA, a language for programming and description of discrete event systems. Introduction and User's Manual. Norwegian Computing Center, Oslo, 1966

/19/ CONTROL DATA CORPORATION: 6400/6500/6600 Computer Systems: SIMULA Reference Manual. Nr. 60234800 Rev.A, Palo Alto, 1969

/20/ NIEDEREICHHOLZ, J.: Einführung in die Simulationssprache SIMULA. El.Datenverarbeitung 12(1970)5, 204-207

/21/ BOESCH, R., BENN, J.: SIMULA 6000 - Beispiele. Rechenzentrum der FIDES-Treuhandvereinigung, Publ.36, 1. Ausgabe 1970

/22/ TOCHER, K.D.: Review of Simulation Languages. Operations Research Quarterly, 16(1965)2, 189-217

/23/ FENNEL, H.J.: Simulation und die Simulationssprachen SIMSCRIPT und GPSS II. El.Datenverarbeitung 7(1965)3, 130-140

/24/ TEICHROEW, D., LUBIN, J.F.: Computer Simulation - Discussion of Technique and Comparison of Languages. Comm.ACM 9(1966)10, 723-741

/25/ KRASNOW, H.S.: Dynamic Presentation in Discrete Interaction Simulation Languages. In "Digital Simulation in Operations Research" S.H.HOLLINGDALE (Ed.), The English Universities Press Ltd., London, 1967

/26/ WEINERT, A.E.: A SIMSCRIPT-FORTRAN case study. Comm.ACM 10(1967)12, 781-792

/27/ BUXTON, J.N.(Ed.): Simulation Programming Languages. North-Holland Publishing Company, Amsterdam, 1968.

/28/ KIVIAT, P.J.: Digital Computer Simulation: Computer Programming Languages. The Rand Corporation, Santa Monica, Calif., RM-5883-PR, Jan.1969

/29/ NOSKA, T.: Ermittlung der für Planung und Betrieb von Hüttenwerken geeigneten Simulationssprachen. Dissertation an der Montanistischen Hochschule Loeben, 1970

/30/ BERGER, B., SEIBT, D., STRUNZ, H.: Bibliographie des Betriebswirtschaftlichen Institutes für Organisation und Automation an der Universität Köln zum Thema "Programmiersprachen". El.Datenverarbeitung 11(1969)5, 225-234; 7, 330-341; 8, 387-397

/31/ LÜTTGENS, H.-G., SEIBT, D.: Ergänzungen zu Literatur /28/. Angewandte Information 13(1971)3, 137-144

/32/ EMSHOFF, J.R., SISSON, R.L.: Simulation mit dem Computer. Verlag Moderne Industrie, München, 1972