

Universität Stuttgart

Institut für Nachrichtenvermittlung und Datenverarbeitung
Prof. Dr.-Ing. habil. Dr. h.c. P.J. Kühn

70. Bericht über verkehrstheoretische Arbeiten

Modellierung und Parallele Simulation von Verbindungsnetzen

von
Michael Jurczyk
1996

D 93

© 1996 Institut für Nachrichtenvermittlung und Datenverarbeitung, Universität Stuttgart
Druck: E. Kurz & Co., Druckerei + Reprografie GmbH., Stuttgart
ISBN 3-922403-80-8



University of Stuttgart

Institute of Communications Switching and Data Technics
Prof. Dr.-Ing. habil. Dr. h.c. P.J. Kühn

70th Report on Studies in Congestion Theory

Modelling and Parallel Simulation of Interconnection Networks

by
Michael Jurczyk

1996

Abstract

In order to achieve significant performance in parallel processing and broadband communication systems it is necessary to support fast data exchange via the interconnection network(s) incorporated into those systems. Network parameters that are essential for fast communication (e.g., network topology and architecture) have to be evaluated and selected prior to the network implementation through the use of analysis and/or simulation. The parameter selection and the resulting network are strongly dependent upon the assumed network traffic. Networks that exhibit high performance under uniform traffic might suffer from a severe performance degradation under nonuniform traffic patterns (e.g., hot-spot traffic) due to saturation trees of full buffers within the network, which might result in a performance degradation of the whole parallel system. This thesis therefore concentrates on nonuniform traffic patterns in one class of interconnection networks, the multistage interconnection networks (e.g. Delta and Generalized-Cube networks). Effects of static and dynamic nonuniform traffic patterns in interconnection networks are investigated, a scheme to classify networks with respect to the achievable performance is presented, and methods for performance enhancement in those networks are proposed. The performance evaluation is based on parallel network simulation running on SIMD computers.

The implementation of a network simulator on a large-scale SIMD machine gives the programmer multiple choices of how to map the network structure onto the machine. A major distinction between the various mapping strategies is the *mapping granularity.* In this thesis, two mapping granularities, the link mapping (a network link is mapped onto one PE) and the box mapping (a whole switch box is mapped onto one PE), are implemented on the MasPar MP-1 SIMD computer and their performance is compared. It is shown that although the link mapping inherits higher parallelism, it achieves at most only half the performance of the box mapping simulator. With the box mapping simulator, 10^9 data packets can be simulated in 40 minutes on the MasPar system. Also, the parallel simulator is 150 times faster than a serial simulator implemented on a SUN workstation.

Nonuniform traffic can degrade the overall performance of multistage interconnection networks substantially. In this thesis, the performance degradation is traced back to blocking effects that are not present under uniform traffic patterns within a network. This blocking phenomenon is termed *higher order Head-of-Line-blocking* (*HOL_k-blocking*). Methods to determine the HOL-blocking order of multistage networks in order to classify the networks are presented together with construction methods of higher order HOL-blocking switch boxes. The performance of networks under nonuniform traffic as a function of their HOL-blocking characteristics is studied by analysis and simulation. It is shown that network bandwidth and packet delay improve under nonuniform traffics with increasing HOL-blocking order of a network. Also, it is discussed how the classification scheme can be used to optimize a recently proposed switch box architecture. It is shown that minor design modifications increase the HOL-blocking order of networks

consisting of those switches, which substantially increase the network performance under nonuniform traffic patterns.

Higher order HOL-blocking networks can effectively deal with static nonuniform traffic patterns. However, they cannot alleviate dynamic network overloads under temporary nonuniform traffic scenarios. Therefore, a switch box architecture for multistage interconnection networks is proposed that minimizes the degrading effects of saturation trees on the uniform background traffic under temporary nonuniform traffic patterns. Switches can be implemented with little hardware overhead. Three transfer priority mechanisms are introduced and their effectiveness in saturation tree effect minimization under temporary hot-spot traffic is discussed. The proposed alternating transfer priority mechanism can minimize the uniform traffic delay and the length of the network overload phase under temporary nonuniform traffic scenarios, while it does not degrade performance under pure uniform traffic patterns. To also account for hot-spot producing program segments that require a minimized hot-spot phase length, an adaptive transfer mechanism is introduced.

The proposed priority switch box architecture can not only be used in parallel processing systems to control network behavior but also in ATM-communication systems to transmit time-critical data streams like MPEG-coded video streams. Time-critical connections are often influenced and delayed by background traffic in conventional ATM-networks. This can be fully eliminated when the proposed priority switch is used within the ATM-network and the time-critical data stream is given highest transfer priority. This results in minimal delay, minimal delay jitter, and low cell loss probability of the time-critical data stream.

Inhaltsverzeichnis

Inhaltsverzeichnis iii

Formelzeichen vii

1 Einführung und Stand der Technik 1

1.1 Grundlagen der Computerkommunikation 1

1.2 Parallelrechnerstrukturen 2

1.2.1 Die Rechnerklasse SIMD 4

1.2.2 Die Rechnerklasse MIMD 5

1.3 Organisation von Verbindungsnetzen in Parallelrechnern 6

1.4 Organisation von Verbindungsnetzen in Übermittlungssystemen 7

1.5 Topologien von Verbindungsnetzen 9

1.5.1 Direkte Verbindungsnetze 9

1.5.2 Indirekte Verbindungsnetze 10

1.6 Betriebsaspekte 11

1.6.1 Operationsmodus 11

1.6.2 Kommunikationsflexibilität 12

1.6.3 Steuerungsstrategien 12

1.6.4 Vermittlungsverfahren 13

1.6.5 Blockierungen 15

1.7 Realisierungsaspekte 16

1.7.1 Schnittstellenstruktur 16

1.7.1.1 Schnittstellen in direkten Netzen 16

1.7.1.2 Schnittstellen in indirekten Netzen 18

1.7.1.3 Kommunikationsprozessoren 18

1.7.2 Router- und Koppelelement-Architektur 18

1.7.2.1 Steuerungsstrategie 19

1.7.2.2 Verbindungsstruktur 19

1.7.3 Technologische Realisierung 20

1.8 Datenverkehre in Verbindungsnetzen 21

1.9 Übersicht über die Arbeit 22

2 Indirekte Verbindungsnetze 24

2.1 Modell indirekter Netze 24

2.2 Indirekte Einpfadnetze 25

2.2.1 Banyan-Netze 25
2.2.2 Delta-Netze 26
2.2.2.1 Topologie 27
2.2.2.2 Routing 28
2.2.2.3 Permutationen 30
2.2.2.4 Topologische Äquivalenz von Netzen 31
2.3 Indirekte Mehrpfadnetze 34
2.3.1 Clos-Netze 34
2.3.2 Beneš-Netze 35

3 Koppelelementarchitekturen und VLSI-Realisierungsaspekte 37
3.1 Eingangspuffer 37
3.2 Ausgangspuffer 40
3.2.1 Interne Paketvermittlung 41
3.2.2 Externe Paketvermittlung 42
3.3 Verteilte Pufferung 43
3.4 Zentralpufferung 43
3.5 Prioritätsmechanismen in ATM-Koppelelementen 45
3.5.1 Verdrängungssystem 45
3.5.2 Schwellwertverfahren 45
3.6 VLSI-Realisierungsaspekte 46
3.6.1 Zentralpufferorganisation mit paralleler Pufferstruktur 46
3.6.2 Zentralpufferorganisation mit Schieberegisterstruktur 47

4 Datenverkehrsmodelle 52
4.1 Verkehrsmodelle für Parallelrechner 52
4.1.1 Gleichverteilter Verkehr 52
4.1.2 Hot-Spot-Verkehr 53
4.1.2.1 Statischer Hot-Spot-Verkehr 53
4.1.2.2 Normalverteilter Hot-Spot-Verkehr 54
4.1.3 Permutationsverkehr 55
4.1.3.1 Statischer Bit-Reverse-Permutationsverkehr 56
4.1.3.2 Temporärer Bit-Reverse-Permutationsverkehr 59
4.2 MPEG-Codierte Videoströme in ATM-Kommunikationssystemen 59

5 Leistungsbestimmung von Netzen 62
5.1 Leistungsparameter von paketvermittelnden Verbindungsnetzen 62

5.2 Analyseverfahren zur Leistungsbestimmung 63
5.3 Simulationstechniken zur Leistungsbestimmung 64
5.3.1 Aspekte der parallelen Simulation 65
5.3.1.1 Parallele ereignisgesteuerte Simulation 65
5.3.1.2 Parallele zeitgesteuerte Simulation 67
5.3.2 Zeitgesteuerte Simulation von synchronen Delta-Netzen auf SIMD-Parallelrechnern 67
5.3.2.1 Abbildungsgranularität 68
5.3.2.2 Kommunikationsaufwand 69
5.3.2.3 Speicherbedarf 69
5.3.2.4 Zusatzaufwand für Koppelelementoperationen 69
5.3.2.5 Leistung von parallelen zeitgesteuerten Simulatoren 70

6 Leistungsverhalten, Modellierung und Optimierung von Delta-Netzen unter ungleichverteilten Datenverkehren 72

6.1 Effekte von ungleichverteilten Verkehren in Delta-Netzen 72
6.1.1 Statisch ungleichverteilte Verkehre 73
6.1.2 Temporär ungleichverteilte Verkehre 76
6.2 Maßnahmen zur Verminderung von Sättigungsbaumeffekten in Delta-Netzen unter ungleichverteilten Verkehren 81
6.2.1 Kombinierende Netze 81
6.2.2 Flußkontrollmechanismen 82
6.2.3 Erweiterte Koppelelementarchitekturen 84
6.3 Netzklassifizierung mittels der HOL-Blockierungen höherer Ordnung 84
6.3.1 Konventionelle HOL-Blockierungen 85
6.3.2 HOL-Blockierungen höherer Ordnung 87
6.3.3 Klassifikation von Delta-Netzen unter Verwendung des Konzepts der HOL-Blockierungen höherer Ordnung 92
6.3.4 VLSI-Realisierungsaspekte von Koppelelementen für HOL_H-Netze 95
6.3.4.1 Realisierungsaspekte der Koppelelementausgangspuffer 95
6.3.4.2 Realisierungsaspekte der Koppelelementeingangslatches 101
6.3.4.3 Realisierungsaspekte der Koppelelementeingangspuffer 103
6.3.5 HOL_H-Verbindungsnetze unter gleichverteilten Verkehren 104
6.3.6 Leistungsbewertung von HOL_H-Verbindungsnetzen unter statisch ungleichverteilten Verkehren 107
6.3.6.1 Analyse von HOL_H-Verbindungsnetzen unter statisch ungleichverteilten Verkehren 107

6.3.6.2 Leistung von HOL_H-Verbindungsnetzen unter statischem Hot-Spot-Verkehr 109
6.3.6.3 Leistung von HOL_H-Verbindungsnetzen unter statischem Bit-Reverse-Permutationsverkehr 112
6.3.7 Leistungsbewertung von HOL_H-Verbindungsnetzen unter temporär ungleichverteilten Verkehren 114
6.3.7.1 Leistung von HOL_H-Verbindungsnetzen unter normalverteiltem Hot-Spot-Verkehr 114
6.3.7.2 Leistung von HOL_H-Verbindungsnetzen unter temporärem Bit-Reverse-Permutationsverkehr 116
6.3.8 Optimierung einer Koppelelementarchitektur unter Zuhilfenahme des HOL-Blockierungskonzepts 117
6.4 Koppelelementarchitektur zur Verkehrssteuerung in Delta-Netzen 119
6.4.1 Prioritätskoppelelementarchitektur 119
6.4.2 Transferprioritätsmechanismen und deren Leistungsfähigkeit in Parallelrechnernetzen 122
6.4.2.1 Niedrige Transferpriorität der heißen Pakete 122
6.4.2.2 Hohe Transferpriorität der heißen Pakete 123
6.4.2.3 Alternierende Transferpriorität der heißen Pakete 125
6.4.2.4 Adaptive Alternierende Transferpriorität der heißen Pakete 131
6.4.3 Leistungsfähigkeit der Prioritätskoppelelementarchitektur in ATM-Netzen unter MPEG-codierten Videoströmen 132

7 Zusammenfassung und Ausblick 137

Literaturverzeichnis 141

Formelzeichen

A	Netzauslastung
A_{ub}	obere Schranke der Netzauslastung
B	Koppelelementgröße
C	Gesamtpufferkapazität eines Koppelelements
C_{min}	minimale Gesamtpufferkapazität eines HOL_H-Koppelelementes
D	Länge eines Koppelelementpuffers
D_{min}	minimale Länge eines Koppelelementpuffers in einem HOL_H-Koppelelement
D_2	Grenzwert der Pufferlänge für niedrig-priorisierte Zellen in partial buffer sharing ATM-Koppelelementen
h	Anteil des ungleichverteilten Verkehrs am Gesamtverkehr
H	HOL-Blockierungsordnung
H_{max}	maximal erreichbare HOL-Blockierungsordnung eines Netzes
j	Nummer einer Netzstufe
K	Prioritätsparameter
k	Länge eines Wegsegments in einem Netz
L	Netzlinkbandbreite
l_i	Nummer eines Eingangsports einer Netzstufe
N	Netzgröße
N_{ib}	Anzahl der parallelen Eingangspuffer eines Koppelelementeingangsports
N_{ob}	Anzahl der parallelen Ausgangspuffer eines Koppelelementausgangsports
N_S	Anzahl der Prozessoren eines Parallelrechners
P	Anzahl der Prozessoren, die eine Simulatorabbildung benötigt
P_{HL}	Wahrscheinlichkeit, daß ein Datenpaket mit einer gleichverteilten Zieladresse an einen bestimmten Eingangslink der Stufe s–H in einem Delta-Netz adressiert ist
P_{HP}	Wahrscheinlichkeit, daß ein Paket auf einem heißen Weg vom Netzeingang bis zu einem Eingangslink von Stufe s–H in einem Delta-Netz propagiert
P_{NHP}	Wahrscheinlichkeit, daß ein Paket nicht auf einem heißen Weg vom Netzeingang bis zu einem Eingangslink von Stufe s–H in einem Delta-Netz propagiert
PS_i	Menge von Wegsegmenten, die am Eingangsport l_i einer Netzstufe enden
R	Tiefe des Sättigungsbaums
s	Stufenanzahl eines Netzes
T	Anzahl von zusätzlichen Transistoren, um ein Koppelelement mit Zentralpuffer in ein HOL_H-Koppelelement zu überführen
T_h	Länge der Hot-Spot-Phase
T_o	Länge der Überlastphase
U	Anzahl der Koppelelemente, die bei der Teilnetzabbildung auf einem Rechnerknoten simuliert werden

Z	Größe des Zentralpuffers in einem Koppelelement mit Zentralpufferung
Z_{min}	minimale Größe des Zentralpuffers in HOL_H-Koppelelementen
λ	Verkehrsrate
λ_v	Videorate an einem Netzausgang
λ_0	minimale Verkehrsrate, bei der ein Sättigungsbaum entsteht
λ_{max}	maximale Verkehrsrate, bei der Sättigungsbäume im Prioritätsnetz nicht entstehen
Δ	Differenz von Verkehrsrate und minimale Verkehrsrate, bei der ein Sättigungsbaum entsteht ($\Delta = \lambda - \lambda_0$)
μ	Mittelwert der normalverteilten Hot-Spot-Verteilung
σ	Streuung der normalverteilten Hot-Spot-Verteilung

1 Einführung und Stand der Technik

In Parallelrechnern und Breitbandübermittlungssystemen kommt dem Austausch von Daten eine primäre Bedeutung zu. Eine der leistungsbestimmenden Komponenten dieser Systeme ist deshalb das Verbindungsnetz. Die Hauptaufgabe von Übermittlungssystemen besteht darin, möglichst viele gleichzeitige Verbindungen zwischen Gesprächsteilnehmern (z. B. in einem Telefonnetz) aufzubauen. Die hierzu verwendeten Verbindungsnetze müssen dies in definierten Parametergrenzen erfüllen können. Netze in Parallelrechnern sorgen für den schnellen und sicheren Datenaustausch unter den Recheneinheiten. Insbesondere bei der Ausführung von Parallelrechnerprogrammen, die kommunikationsintensiv sind, kann ein Verbindungsnetz mit suboptimalen Leistungscharakteristiken zum leistungsbegrenzenden Element des gesamten Parallelrechners werden. Um leistungsstarke Parallelrechner zu erhalten, muß deshalb sichergestellt sein, daß das verwendete Verbindungsnetz die zu erwartenden Datenverkehre schnellstmöglich und sicher vermitteln kann.

1.1 Grundlagen der Computerkommunikation

Als eine Voraussetzung für den Einsatz und die Auswahl eines Verbindungsnetzes soll hier zunächst in allgemeiner Weise die Computerkommunikation diskutiert werden, insbesondere das ISO OSI-Schichtenmodell, durch das der Kontext für viele Anforderungen an Verbindungsnetze geschaffen wird.

In Kommunikationssystemen wie dem Internet werden Verbindungen oft über eine große Anzahl von Zwischenknoten und unterschiedlichste Transportmedien aufgebaut, welche vom lokalen Ethernet bis zur transatlantischen Glasfaserleitung reichen. Um einen solchen Datenaustausch zwischen Computern zu ermöglichen, ist es erforderlich, daß sich die Teilnehmer der Kommunikationsverbindung über die Bedeutung der ausgetauschten Daten einig sind. Manche Bits des Datenstroms sind Nutzdaten, andere dienen der Datensicherung, und wieder andere werden für die Festlegung der Verbindung benötigt. Diese Einigung geschieht auf der Basis von **Protokollen**. Dabei ist ein einzelnes Protokoll meist nicht ausreichend.

Um die unterschiedlichen Protokollaufgaben zu standardisieren und damit die Entwicklung von Protokollen zu vereinfachen, wurde das **ISO OSI-Schichtenmodell** (ISO: International Standards Organization; OSI: Open Systems Interconnection) [Iso84] standardisiert, dessen sieben Schichten in Abbildung 1.1 dargestellt sind. Die Schichten sind von 1 bis 7 numeriert, deren wesentliche Aufgaben hier nur kurz dargestellt werden sollen. Ausführliche Diskussionen hierzu finden sich beispielsweise in [Tan81]. Ein Anwendungsprozeß tauscht Daten nur mit der obersten Schicht 7, der **Anwendungsschicht** (**application layer**) aus; der Inhalt dieser Schicht wird daher von den Benutzern bestimmt. Schicht 6, die **Datendarstellungsschicht** (**presentation layer**), führt Datentransformationen wie Kompression und Verschlüsselung durch. Die **Kom-**

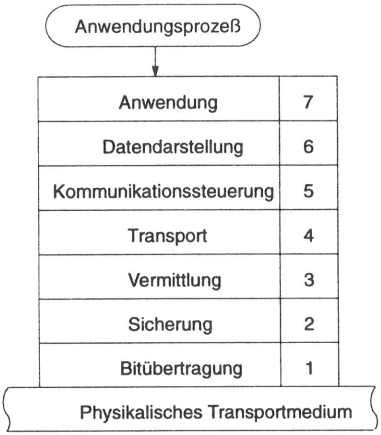


***Abbildung 1.1:** ISO OSI-Schichtenmodell.*

munikationssteuerung (session layer) handhabt Auf- und Abbau sowie Verlauf der Kommunikation zwischen Prozessen. Die **Transportschicht (transport layer)** stellt zuverlässige Verbindungen zwischen den Endteilnehmern sicher und verbirgt alle Details des Kommunikationsweges. Die **Vermittlungsschicht (network layer)** ist für die Wegesuche verantwortlich. **Die Sicherungsschicht (data link layer)** stellt einen zuverlässigen Datentransport sicher, auch wenn während des Datentransports Fehler auftreten. In der **Bitübertragungsschicht (physical layer)** werden die Daten an die Gegebenheiten des physikalischen Transportmediums angepaßt.

Diese Protokollstruktur ermöglicht die Kommunikation zwischen zwei Anwenderprozessen, unabhängig von eventuellen Zwischenknoten oder Transportmedien. Für die Protokolle auf Schichten 1 bis 3 sind Zwischenknoten sichtbar; dort kommunizieren die Schichten zwischen Endteilnehmern und Zwischenknoten. Auch Schichten 4 bis 7 kommunizieren virtuell direkt miteinander, jeweils unter Verwendung der darunterliegenden Protokollschichten.

Je nach Anwendung müssen nicht alle Schichten in einem System realisiert werden (Funktionen können durch Hardware oder durch Software realisiert sein). Insbesondere die Funktionen der unteren Schichten wie Bitübertragung und Sicherung werden oft aufgrund der hohen Übertragungsleitung durch Hardwarekomponenten übernommen.

1.2 Parallelrechnerstrukturen

Sowohl in der Forschung als auch im kommerziellen Bereich sind Parallelrechner unterschiedlichster Architektur unter Ausnutzung der unterschiedlichsten Methoden des Datenaustausches

entworfen und auch realisiert worden. Da die Auswahl und Eignung einer Kommunikationsstrategie fundamental von der Rechnerarchitektur und den darauf abgebildeten Algorithmen abhängt, werden die wesentlichen Parallelrechnerkonzepte hier kurz diskutiert. Ausführliche Diskussionen über die Strukturen, Eigenschaften und Fallbeispiele von Parallelrechnern finden sich beispielsweise in [AtS88, Dun90, Hwa93, HwB84].

Parallelrechner zeichnen sich durch eine Anzahl von Prozessoren aus, die parallel Berechnungen ausführen und somit gemeinsam an einem Problem arbeiten können. Ein Verbindungsnetz ermöglicht die Kommunikation zwischen den einzelnen Recheneinheiten. Grundsätzlich existieren zwei Organisationsformen von Parallelrechnern: die PE-zu-PE- und die Prozessor-zu-Speicher-Organisation. In der **PE-zu-PE-Parallelrechnerorganisation** bildet ein Prozessor-Speicherpaar eine Einheit: das **PE** (**processing element**). Die *N* PEs sind, wie in Abbildung 1.2 gezeigt, durch ein Verbindungsnetz miteinander verbunden.

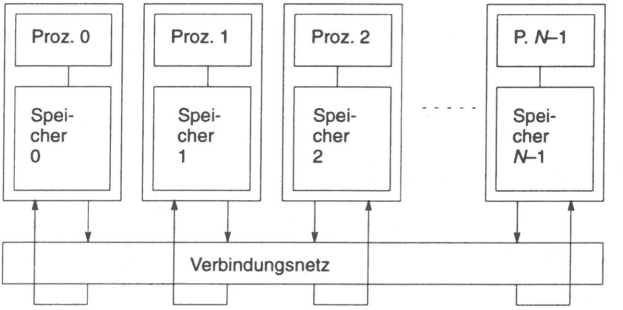


Abbildung 1.2: *PE-zu-PE-Organisation eines Parallelrechners.*

Da ein Zugriff auf einen globalen Speicher programmtechnisch vorteilhaft ist, wird dies auch bei der PE-zu-PE-Struktur angestrebt und hat zu Konzepten des virtuell gemeinsamen Speichers [StZ90] geführt.

In der **Prozessor-zu-Speicher-Organisation** sind die Prozessor- und die Speichermodule getrennt angeordnet. Die *N* Prozessoren können, wie in Abbildung 1.3 gezeigt, über ein bidirektionales Verbindungsnetz mit den *M* Speichermodulen verbunden sein. In manchen Systemen sind den Prozessoren Caches zugeordnet.

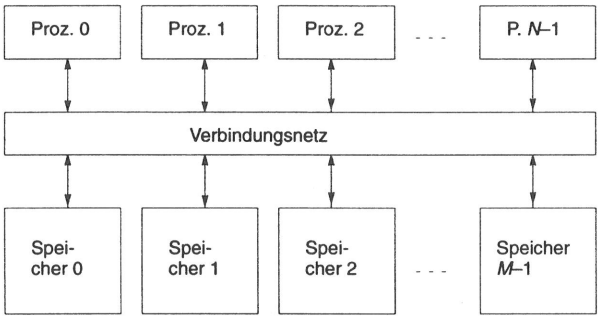


Abbildung 1.3: *Prozessor-zu-Speicher-Organisation eines Parallelrechners.*

Unabhängig von ihrer Organisation lassen sich nach Flynn Parallelrechner in zwei Hauptklassen einteilen [Fly66]: die SIMD und die MIMD Klassen. Die Unterschiede und Gemeinsamkeiten dieser Klassen werden nun kurz vorgestellt.

1.2.1 Die Rechnerklasse SIMD

Das Systemmodell der Rechnerklasse **SIMD (single instruction stream – multiple data stream)** ist in Abbildung 1.4 gezeigt. In SIMD-Rechnern, die überwiegend in der PE-zu-PE-Struktur organisiert sind, sendet eine übergeordnete Kontrolleinheit Maschinenbefehle zeitsynchron an alle Prozessoren [JuS95b]. Diese Befehle werden synchron von den Prozessoren ausgeführt, wobei jeder Prozessor auf seine lokalen Daten zugreift und diese bearbeitet. Die Kommunikation zwischen den Prozessoren erfolgt ebenfalls synchron über das Verbindungsnetz; das Netz muß deshalb für synchrone Datentransfers ausgelegt sein. Zur Ausführung von Fallunterscheidungen, wie z. B. *if..then..else* Konstrukten, werden die Prozessoren durch entsprechende Maskierung aktiv oder passiv geschaltet. Nur aktive PEs führen Befehle aus, während passive PEs die Befehle ignorieren. Typische Vertreter dieser Rechnerklasse sind die MasPar MP-1 [Bla90], die Connection Machine CM-2 [TuR88] und die MPP [Bat80].

In der Regel bestehen diese Rechner aus einfach aufgebauten und relativ leistungsschwachen Prozessoren. Auch ist die Speicherkapazität pro PE klein. Die Leistung bezieht ein SIMD-Rechner aus der hohen Anzahl von parallel arbeitenden Prozessoren. Die MasPar MP-1 z. B. besteht aus 16K PEs mit einer Datenwortbreite von 4 bit und einer Speicherkapazität von 16KByte pro PE [Bla90, Nic90], während die Connection Machine CM-2 mit 64K PEs, bestehend aus jeweils einem 1-bit Prozessor und einem 64KByte RAM-Modul, ausgerüstet ist [Hil85, TuR88].

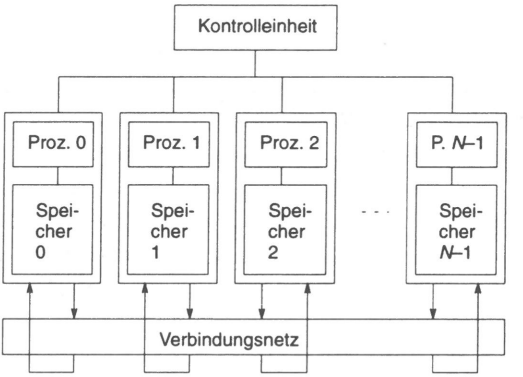


Abbildung 1.4: *Systemmodell eines SIMD-Parallelrechners mit PE-zu-PE-Organisation.*

1.2.2 Die Rechnerklasse MIMD

In Systemen der Rechnerklasse **MIMD (multiple instruction stream – multiple data stream)** arbeiten eine begrenzte Anzahl leistungsstarker Prozessoren unabhängig voneinander gleiche oder verschiedene Programme ab. Diese können in der Prozessor-zu-Speicher- wie auch in der PE-zu-PE-Struktur organisiert sein. Daten werden zwischen den Prozessoren und den Speichern bzw. zwischen den PEs asynchron über das Verbindungsnetz ausgetauscht. Beispiele von MIMD-Rechnern sind die Intel Paragon XP/S [EsK93], die IBM RP3 [PfB85], das EDS System [HaL90] und die BBN Monarch [ReC90].

Ähnlich wie bei SIMD-Rechnern ist bei MIMD-Rechnern neben dem Datenaustausch die Synchronisation zwischen Prozessoren eine der wesentlichen Aufgaben, die vom Verbindungsnetz unterstützt werden müssen. Der Verkehr durch das Netz ist daher durch asynchrone Transfers und ungleichmäßige Verkehrsverteilungen gekennzeichnet.

Die Synchronisation zwischen Prozessoren ist bei MIMD-Systemen eine der aufwendigsten Probleme und erfordert in der Regel erhebliche Kommunikation durch das Netz. Daher wurden einerseits Verbindungsnetze entworfen, die effiziente Synchronisation erlauben (siehe kombinierende Netze in Abschnitt 6.2.1), und andererseits wurden Maschinen konstruiert, die dynamisch zwischen den SIMD- und MIMD-Betriebsarten wechseln können und so die inhärente SIMD-Synchronisation nutzen können. Beispiele sind PASM [SiS81], TRAC [SeU80] und die Connection Machine CM-5 [PoT93]. Das CM-5-System nutzt Hardware-Barrier-Synchronisation, ein der SIMD-Methode ähnliches Prinzip [GhC94, ScD87].

1.3 Organisation von Verbindungsnetzen in Parallelrechnern

Die Struktur des Parallelrechners hat einen entscheidenden Einfluß auf die Organisation des zu verwendenden Verbindungsnetzes. In einem Rechner mit Prozessor-zu-Speicher-Organisation muß das Netz Schreib- und Leseoperationen zu und von den Speichermodulen unterstützen. Bei Schreiboperationen sendet der Prozessor die zu speichernden Daten zusammen mit der Speicheradresse zu den Speichermodulen. Bei einem Lesezugriff muß der Prozessor zuerst die Leseadresse zu den Speichermodulen senden. Das ausgewählte Speichermodul sendet dann das gelesene Datenwort zum Prozessor zurück. Dieser Mechanismus kann durch ein bidirektionales Netz (Abbildung 1.5a) oder durch getrennte Hin- und Rücknetze (Abbildung 1.5b) implementiert werden. Im bidirektionalen Netz können sich allerdings die Hin- und Rücknachrichten gegenseitig beeinflussen (z. B. sich gegenseitig blockieren oder aufhalten), während dies im Fall der getrennten Netze nicht auftreten kann.

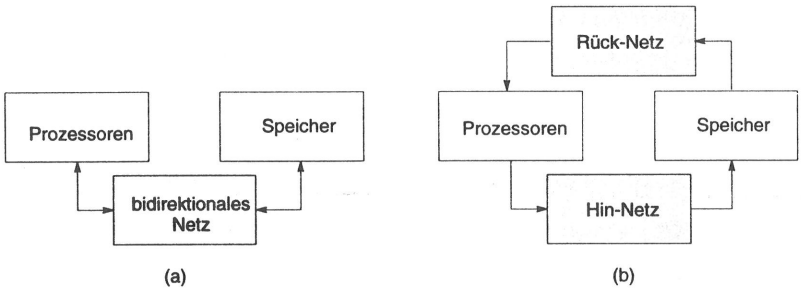


Abbildung 1.5: *Parallelrechnersystem in Prozessor-zu-Speicher-Organisation mit (a) bidirektionalem Netz und (b) mit Hin- und Rück-Netz.*

In Parallelrechnern mit der PE-zu-PE-Organisation kann die Kommunikation unidirektional erfolgen. Hierfür können unidirektionale gefaltete Netze (Abbildung 1.6a) und auch bidirektionale einseitige Netze (Abbildung 1.6b) verwendet werden.

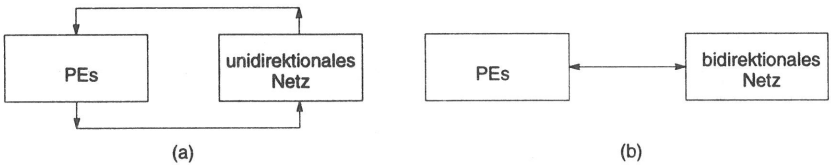


Abbildung 1.6: *Parallelrechnersystem in PE-zu-PE-Organisation mit (a) gefaltetem und (b) einseitigem Netz.*

1.4 Organisation von Verbindungsnetzen in Übermittlungssystemen

Abbildung 1.7 zeigt ein klassisches Telefonsystem mit Vermittlungsstellen und Übertragungsstrecken. In einer Vermittlungsstelle werden Gesprächsteilnehmer miteinander verbunden, und auf den Übertragungsstrecken werden viele Gespräche gebündelt und so zwischen Vermittlungsstellen übertragen. Das Telefonsystem ist als ein hierarchisches Netz angelegt, in dem möglichst effektiv Kommunikation zwischen Teilnehmern ermöglicht wird. Die einzelnen Vermittlungsstellen stellen Verbindungsnetze bereit, die zwischen einer großen Zahl von Teilnehmern gleichzeitige Verbindungen herstellen können. Diese Netze waren ursprünglich ausschließlich leitungsorientiert und – insbesondere im lokalen Bereich – auch in Analogtechnik konzipiert; sie stellten im lokalen Bereich eine glavanische Verbindung zwischen den Gesprächspartnern bereit.

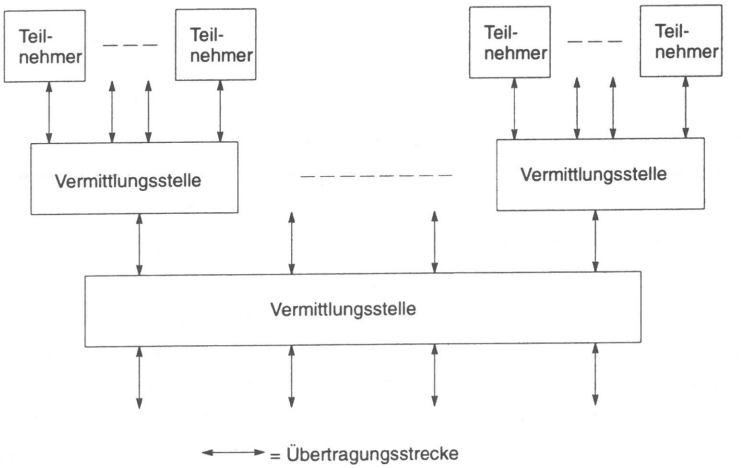


Abbildung 1.7: *Übermittlungssystem mit hierarchischen Vermittlungsstellen.*

Vermittlungsstellen sind durch die Möglichkeit der zentralen Steuerung gekennzeichnet, die bei vielen Parallelrechnern nicht akzeptabel ist. So sind bei einem Parallelrechner mit gemeinsamem Speicher die erforderlichen Verbindungen von Speicherzugriff zu Speicherzugriff verschieden, so daß eine zentrale Verwaltung des Verbindungsaufbaus zu einem Engpaß führen würde. Auch kann keine deterministische Voraussage über die Verteilung der Zugriffe von Prozessoren auf Speicher gemacht werden, so daß es zu lokalen Überlastungen des Netzes kommen kann. Bei einem Telefongespräch hingegen steht mehr Zeit für den Verbindungsaufbau zur Verfügung. Auch bleiben Verbindungen über längere Zeit bestehen. Eine zentrale Steuerung der

Vermittlungsstellen ist also möglich. Diese Steuerung kann über die Verbindungsannahme auch die Auslastung des Netzes kontrollieren und so Überlastsituationen vermeiden.

Die Sprachübertragung stellt heute nur noch einen (kleinen) Teil der verfügbaren Telekommunikationsdienste dar. Während Telefax-Übertragung und Datenübertragung per Modem oft noch über Sprachkanäle geleitet werden, ist für die Rechnerkommunikation und insbesondere für Multimedia-Anwendungen mit Bewegtbildübertragung ein Hochgeschwindigkeitszugang zum Kommunikationsnetz erforderlich. Dies wird heute durch gemeinsame dienstintegrierte, digitale Netzzugänge für Daten und Sprache zur Verfügung gestellt; ein Beispiel ist **ISDN**, das **Integrated Services Digital Network**. Die Integration verschiedener Dienste stellt höchst unterschiedliche Anforderungen an die Datenübertragung. Beispielsweise reicht die erforderliche Datenrate von 64 Kbit/s für digitale Telefonübertragung bis 100 MBit/s für Bewegtbildübertragung und darüber hinaus. Manche Dienste, wie die reine Rechnerdatenübertragung sind zeitunkritisch, während andere Dienste hohe Anforderungen an Isochronität stellen, z. B. die Echtzeit-Videoübertragung.

Den Problemen, die durch die unterschiedlichen Anforderungen an die Dienste entstehen, kann durch dienstspezifische Vermittlungen und Übertragungskanäle entgegengewirkt werden. In einem solchen System kann ein Teilnehmer zwar alle Dienste am gleichen Zugang einspeisen, jedoch werden die Dienste klassifiziert und im Kommunikationssystem je nach Typ z. B. über Video- oder Datenkanäle übertragen. Diese Trennung erleichtert die Verwaltung des Systems, da sich die Dienste mit ihren unterschiedlichen Anforderungen nicht gegenseitig beeinflussen. Der wesentliche Nachteil ist die Notwendigkeit paralleler Kanäle mit entsprechender paralleler Hardware, deren Bandbreite nicht gegenseitig nutzbar ist.

Mit fortschreitender Digitalisierung des Telekommunikationsnetzes erfolgt zur Zeit ein Übergang zu dienstintegrierter Vermittlung und Übertragung. Hierbei werden alle Nachrichten, unabhängig vom Typ, einheitlich behandelt. Von herausragender Bedeutung ist dabei die **ATM-Breitbandübermittlungstechnik** (**ATM** – **A**synchronous **T**ransfer **M**ode), bei der alle Kommunikationsdaten in Pakete genormter und konstanter Länge aufgeteilt werden, die auch Vermittlungsinformationen enthalten. Beim Zustandekommen einer Kommunikationsverbindung wird kein physikalischer Kanal wie im analogen Telefonnetz aufgebaut, sondern eine virtuelle Verbindung geschaffen, durch die der Weg der Datenpakete durch das Kommunikationsnetz vorgegeben wird.

Integrierte Dienste erfordern hohe Bandbreiten des Systems und intelligente Steuerung, die u. a. für jeden Kanal eine ausreichende Bandbreite reserviert und unter Umständen Prioritäten festlegt. Für die in Vermittlungsstellen eingesetzten Verbindungsnetze hat diese Verkehrsstruktur unmittelbare Auswirkungen auf die zu erfüllenden Anforderungen. Dies reicht von der festen Paketgröße bis zu Leistungsgrößen wie maximaler Verkehrsrate und Verlustwahrscheinlichkeit, erlaubter Durchlaufzeit und akzeptablen Durchlaufzeitschwankungen. In den meisten Fällen werden für ATM-Vermittlungen mehrstufige, indirekte Verbindungsnetze eingesetzt.

1.5 Topologien von Verbindungsnetzen

Die Struktur von Netzen läßt sich durch endliche Graphen repräsentieren, in denen die Kanten den Zwischenleitungen (**Links**) und die Knoten den Koppelelementen bzw. Prozessorelementen entsprechen. Alle aus der Graphentheorie bekannten Merkmale treffen auch auf Verbindungsnetze zu. Die Topologie des Netzes beschreibt die Graphenstruktur, und Netze können in **direkte** und **indirekte** Netze eingeteilt werden.

1.5.1 Direkte Verbindungsnetze

In einem direkten Netz besitzt jedes PE eine Anzahl fester Leitungen zu benachbarten PEs, entlang derer Nachrichten unmittelbar von PE zu PE gesandt werden können. Das Netz beschränkt sich somit auf Direktleitungen (Links); Vermittlungsfunktionen werden durch die PEs selbst ausgeübt, indem Nachrichten entlang der ausgewählten Leitung gesandt werden. Da die Verbindung zwischen Paaren von PEs ohne Möglichkeit der Rekonfiguration erfolgt, wird oft auch von **statischen Netzen** gesprochen. Abbildung 1.8 zeigt als Beispiel eine Reihe möglicher direkter Topologien.

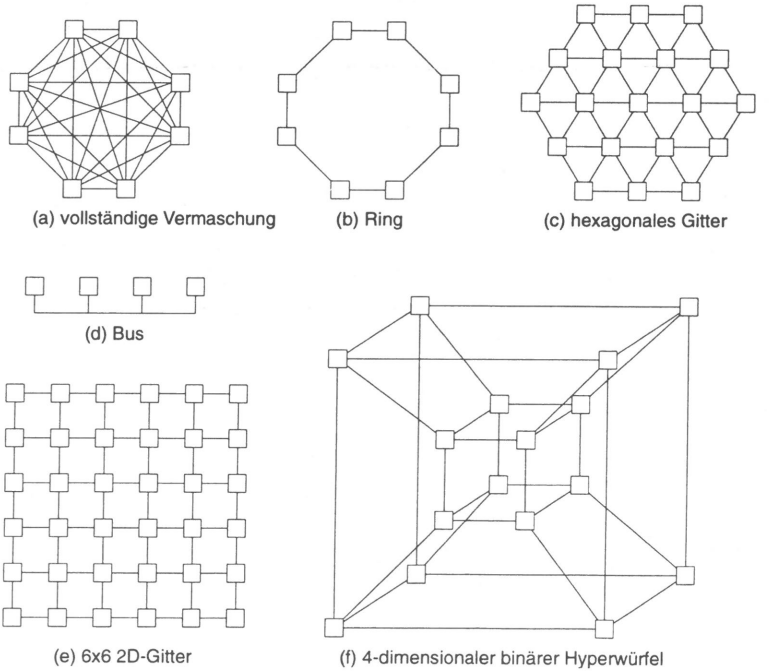


Abbildung 1.8: *Beispiele direkter Netztopologien: (a) vollständige Vermaschung; (b) Ring; (c) hexagonales Gitter; (d) Bus; (e) 2D-Gitter; (f) Hyperwürfel.*

Ein Überblick über Topologien und Architekturen direkter Netze findet sich z. B. in [Hwa93, Fen81, Sie90]. Da in direkten Netzen nur eine Stufe von Verbindungsleitungen existiert, wird in vielen Arbeiten die Bezeichnung **einstufige Netze** genutzt, zum Beispiel in [Sie90]. Diese Nomenklatur wird hier nicht benutzt, um eine Verwechselung mit einstufigen indirekten Netzen zu vermeiden.

1.5.2 Indirekte Verbindungsnetze

Ein indirektes Netz führt alle Vermittlungsaufgaben mittels aktiver Koppelelemente eigenständig durch. In vielen indirekten Netzen ist jedes PE nur über einen Ein- und einen Ausgang mit dem Netz verbunden. Die Netze bedienen sich einer oder mehrerer Stufen von Koppelelementen, in denen die Nachrichten durch das Netz geleitet werden. Abbildung 1.9 zeigt das Shuffle-Exchange-Netz als ein Beispiel eines **einstufigen indirekten Netzes**. Die Koppelelemente dieses Netzes haben im einfachsten Fall zwei Ein- und Ausgänge, besitzen keine internen Puffer und können zur Datenvermittlung die Zustände **straight** und **exchange** annehmen, wie in Abbildung 1.10 gezeigt. Durch Hinzufügen der Zustände **upper broadcast** und **lower broadcast** können Nachrichten von einem Eingang zu beiden Ausgängen vermittelt werden. Um Daten von beliebigen Eingängen zu beliebigen Ausgängen zu vermitteln, muß das Shuffle-Exchange-Netz in manchen Fällen mehrmals durchlaufen werden. Abbildung 1.11 zeigt ein **Generalized-Cube Netz** als Repräsentanten eines mehrstufigen indirekten Netzes [SiN89]. Da die Vermittlung von Nachrichten dynamisch durch die Koppelelemente erfolgt, wird oft der Begriff **dynamische Netze** genutzt, im Gegensatz zu den **direkten "statischen" Netzen**.

Indirekte Netze unterscheiden sich in der Komplexität, Größe und Funktionalität der Koppelelemente, der Anzahl der Stufen und der Struktur der Links zwischen den Stufen (ein Überblick über Architekturen indirekter Netze findet sich z. B. in [Agr83, NaS93, Sie90]). Topologien und Eigenschaften dieser Netze werden in Kapitel 2 näher behandelt.

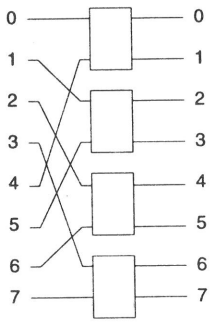


Abbildung 1.9: *Einstufiges Shuffle-Exchange-Netz mit 8 Ein- und Ausgängen.*

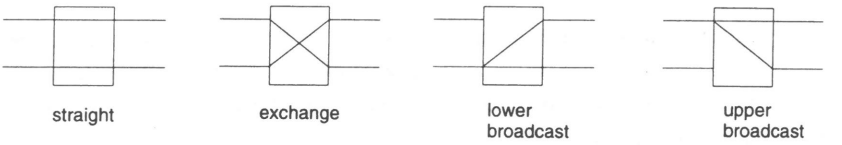


Abbildung 1.10: *Zustände eines 2 x 2 ungepufferten Koppelelements.*

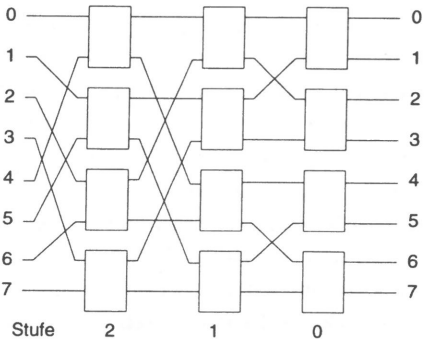


Abbildung 1.11: *Generalized Cube-Netz mit 8 Ein- und Ausgängen.*

1.6 Betriebsaspekte

1.6.1 Operationsmodus

Netze können synchron oder asynchron betrieben werden; dies bezieht sich sowohl auf den Verbindungsaufbau als auch auf die eigentliche Datenvermittlung. In einem **synchronen Netz** werden Nachrichten im festen Verhältnis zu einem zentralen Takt gesandt bzw. empfangen. Dies ist beispielsweise in einem SIMD-Rechner von hoher Bedeutung, da in den erforderlichen Permutationen alle Prozessoren gleichzeitig eine Verbindung anfordern (z. B. von PE k zu $k+1$ mod N) und Daten entlang der aufgebauten Wege synchron zum Befehlstakt weiterleiten.

Bei der **asynchronen Betriebsart** können PEs Verbindungen zu beliebigen Zeitpunkten anfordern, wie dies bei MIMD-Rechnern erforderlich ist. Beim asynchronen Datentransfer existiert

kein globaler Takt, sondern miteinander verbundene Quellen und Senken benutzen Handshake-Protokolle oder mitgeführte lokale Takte, um Daten auszutauschen.

Der Operationsmodus kann unabhängig auf unterschiedliche Kommunikationsphasen angewendet werden. So können Verbindungen synchron oder asynchron angefordert und aufgebaut werden, wobei unabhängig davon die Übertragung individueller Pakete oder auch Datenworte synchron oder asynchron erfolgen kann.

1.6.2 Kommunikationsflexibilität

Im einfachsten Fall unterstützt ein Netz lediglich **Einzelverbindungen** (**one-to-one**), bei denen eine Quelle nur mit einer Senke und umgekehrt kommuniziert. Vielfach ist jedoch eine Datenverteilung erwünscht, für die eine **Multicast-Fähigkeit** (**one-to-many**) [LiN93] des Netzes erforderlich ist, bei der eine Quelle zu einer Untermenge der Senken Daten sendet. In einem Netz mit **Broadcast-Möglichkeiten** (**one-to-all**) [YaM91] kann eine Quelle Daten zu *allen* Senken versenden; Broadcast ist somit ein Spezialfall des Multicast. Oft ist ein gleichzeitiger Zugriff vieler Quellen auf eine einzelne Senke gewünscht, zum Beispiel beim Zugriff auf eine Synchronisationsvariable in einem System mit gemeinsamem Speicher. Der dabei entstehende Engpaß kann durch **kombinierende Netze** [PfN85] (siehe Abschnitt 6.2.1) reduziert werden, die an das gleiche Ziel gerichtete Nachrichten zusammenfassen. Ein anderer Aspekt ist die Permutationsfähigkeit eines Netzes. Ein **blockierendes Netz** kann nur eine Untermenge aller möglichen Permutationen vermitteln. In einem **blockierungsfreien Netz** können alle Datenpermutationen durchgeführt werden, während dies in **rearrangierbaren Netzen** nur durch die Umkonfiguration bestehender Verbindungen erreicht werden kann.

1.6.3 Steuerungsstrategien

Ein Netz besteht in der Regel aus einer Anzahl von Koppelelementen und Zwischenleitungen, deren dynamische Konfiguration die augenblicklichen Verbindungen realisiert. Bei einem **zentral gesteuerten Netz** werden alle Elemente durch eine gemeinsame Steuerung verwaltet. Von Vorteil ist dabei die Möglichkeit, alle Ressourcen koordiniert zu nutzen, so daß eine optimale Verbindungsstruktur erreicht werden kann, allerdings unter dem Nachteil eines hohen Aufwandes zur Rekonfiguration. Zentral gesteuerte Netze können daher nur in Anwendungen eingesetzt werden, bei denen die Kommunikationsstruktur über eine längere Zeitspanne konstant bleibt. Dies ist zum Beispiel bei Schaltnetzen in Telefonvermittlungen der Fall, die über einen zentralen Vermittlungsrechner gesteuert werden, und bei denen aufgebaute Verbindungen über eine längere Zeit – die Dauer des Telefonats – bestehen bleiben. Die **individuelle Steuerung** eines jeden Koppelelements im Netz (**individual box control**) kann sehr aufwendig sein. So müssen beispielsweise in einem mehrstufigen Cube-Netz mit 256 Ein- und Ausgängen 2048 2x2-Koppelelemente gesteuert werden. Sind nicht alle möglichen Einstellungen von Bedeutung, so kann

durch eine **stufenweise Kontrolle (central stage control)** oder eine **partielle Stufenkontrolle (partial stage control)** wie im STARAN Flip-Netz [Bat76] die Komplexität der Steuerleitungen erheblich verringert werden.

Bei einem **dezentral gesteuerten System** übernehmen die individuellen Koppelelemente die Steuerung, wie zum Beispiel im Generalized-Cube Netz. In solchen **selbststeuernden (self-routing) Netzen** können Verbindungen sehr schnell auf- und abgebaut werden, jedoch werden manche Möglichkeiten des Systems wegen der fehlenden Koordination nicht optimal genutzt. Dies gilt insbesondere für die Ausführung von Permutationen, da in vielen Netzen willkürliche Permutationen nur durch die gleichzeitige Berücksichtigung aller Verbindungsanforderungen möglich sind.

1.6.4 Vermittlungsverfahren

Die zwei fundamentalen Vermittlungsverfahren sind die Durchschalte- und die Paketvermittlung. In einem Netz mit **Durchschaltevermittlung** wird vor dem Datentransfer ein vollständiger Weg zwischen Quelle und Senke aufgebaut, entlang dessen dann Daten transferiert werden. Dies kann auf unterschiedliche Weise geschehen; so kann ein Verbindungsaufbau zum Beispiel durch eine zentrale Steuerung erfolgen oder durch Versenden eines Vermittlungspakets, das durch das Netz transportiert wird und dabei eine Verbindung aufbaut. In einem **paketvermittelnden Netz** werden Vermittlungsinformation und Daten zu einem Paket zusammengefaßt, das durch das Netz vermittelt wird, ohne jedoch eine Verbindung aufzubauen. Im einfachsten Fall wird ein Paket, das über mehrere Zwischenstufen vermittelt werden muß, in jeder dieser Stufen komplett zwischengepuffert (**Speichervermittlung, store-and-forward switching**).

Für den Datenaustausch zwischen einer Quelle und einer Senke gibt es die Möglichkeiten der verbindungsorientierten und der verbindungslosen Kommunikation. Bei der **verbindungsorientierten Kommunikation** wird ein Weg zwischen Quelle und Senke etabliert, der für die gesamte Dauer der Verbindung genutzt wird. Die Verbindung kann in einem System mit Durchschaltevermittlung physikalisch über eine feste Leitung bestehen, oder sie kann in einem paketvermittelnden System lediglich virtuell existieren, indem die Pakete, aus denen die Nachricht besteht, stets über den gleichen Kommunikationsweg geleitet werden. Bei einer **verbindungslosen Kommunikation** werden Nachrichtenpakete über den jeweils günstigsten Weg geleitet (es besteht kein bevorzugter Kommunikationsweg). Dies ist nur in einem paketvermittelnden System möglich, in dem während des Datentransports auch Vermittlungsfunktionen ausgeübt werden. In Kommunikationssystemen herrscht die verbindungsorientierte Kommunikation vor, sowohl im klassischen analogen Telefonnetz als auch im modernen Breitbandnetz. Bei paketvermittelnden Parallelrechnern wird je nach Netz auch die verbindungslose Kommunikation eingesetzt, beispielsweise bei direkten Netzen mit adaptiver Wegesuche [DoR91, KaS92, Kon92, LiE93].

Das **Wormhole-Routing** [DrG94, Kec94, KiD91, NiM93] nutzt die Vorteile beider Vermittlungsverfahren in Netzen, bei denen der Datentransport über Zwischenschritte erfolgt. Hierbei werden Pakete in kleinere Komponenten, die **Flits**, aufgeteilt [Dal92]. Das erste Flit enthält die Verbindungsinformation, und in jeder Stufe wird der erforderliche Verbindungsabschnitt aufgebaut. Nach Aufbau der Verbindung erstrecken sich die Flits eines Pakets daher über alle Stufen zwischen Quelle und Senke, so daß ein vollständiger Weg verfügbar ist, entlang dessen weitere Flits schnell transportiert werden können, wie in Abbildung 1.12 von PE *A* zu PE *C* gezeigt. In den meisten Systemen sind die Router und PEs zu Rechner/Kommunikationsknoten zusammengefaßt.

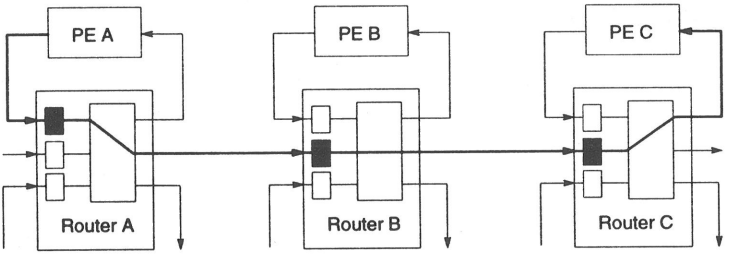


***Abbildung 1.12:** Wormhole-Routing-Weg zwischen PE A und PE C.*

Die wesentlichen Vorteile des Wormhole-Routing sind die reduzierte Latenzzeit zwischen Absenden einer Nachricht und dem Empfang an der Senke sowie die variable Paketlänge, die nicht durch Pufferplatz in einer Zwischenstufe beschränkt wird. Der Nachteil ist die Möglichkeit von Verklemmungen, durch die Nachrichten nie an ihr Ziel gelangen. Eine Erweiterung dieser Methode ist z. B. das Mad-Postman-Routing [GeJ91], bei dem Flits schon vor der Auswertung ihrer Zieladresse weitergeschickt werden, um eine schnellere Datenübertragung zu erhalten. Zur Auflösung von Verklemmungen wurde u. a. das Konzept der **virtuellen Kanäle** [DaA93, Dal92, SuS93] eingeführt.

Mit dem Wormhole-Routing verwandt ist das **Virtual-Cut-Through-Routing** [KeK79] in paketvermittelnden Netzen. Hier ist in einer Stufe des Netzes ausreichend Pufferplatz für die Pufferung des gesamten Pakets vorhanden, so daß Store-and-Forward-Routing möglich ist. Pakete werden durch das Netz transferiert und in jeder Stufe weitervermittelt. Ist ein Wegsegment zur nächsten gewünschten Stufe verfügbar, so wird jedoch nicht auf das Eintreffen des gesamten Pakets in der Stufe gewartet, sondern das Paket schnellstmöglichst weitergeleitet.

Damit ist es auch hier möglich, daß sich ein solches Paket über den gesamten Weg von Quelle zu Senke erstreckt. Ist das Wegsegment zur nächsten Stufe nicht frei, so sammeln sich die Datenworte des Pakets jedoch in der Stufe an und werden gepuffert, bis eine Weiterleitung möglich ist. Somit kann eine längere Blockierung eines Weges wie beim Wormhole-Routing nicht auftreten.

Beispiele für Leitungsvermittlung sind das analoge Telefonnetz sowie die PASM [SiS81] und GF11 Parallelrechner [BeD85, KuB93]. Integrierte digitale Kommunikationsnetze wie Breitbandnetze sowie Rechnersysteme wie RP3 [PfB85] sind Beispiele für Store-and-Forward-Paketvermittlung. Wormhole-Routing wird beispielsweise im Intel-Paragon-Parallelrechner [EsK93] genutzt, und das Verbindungsnetz des EDS-Parallelrechners [HaL90] verfügt über Virtual-Cut-Through-Routing.

1.6.5 Blockierungen

Werden Nachrichten von zwei oder mehr Quellen gleichzeitig zur selben Senke gesandt, so resultiert daraus eine **äußere Blockierung (Abnehmerblockierung)**, und nur eine der Nachrichten kann vermittelt werden (Abbildung 1.13a). In vielen Netzen ergeben sich Blockierungen auch innerhalb des Netzes **(innere Blockierung)**, wenn Nachrichten zwar an unterschiedliche Senken adressiert sind, intern streckenweise jedoch den selben Weg benötigen, wie in Abbildung 1.13b) dargestellt. Während innere Blockierungen in blockierungsfreien oder umkonfiguierbaren Netzen nicht auftreten, sind Abnehmerkonflikte verkehrsabhängig und können daher durch topologische Maßnahmen nicht umgangen werden. Manche Abnehmerblockierungen werden durch kombinierende Netze vermieden, wie in Abschnitt 6.2.1 beschrieben.

Drei prinzipielle Methoden zur Auflösung von Blockierungen sind zu unterscheiden. Bei der **Block-Methode** wartet eine Nachricht, die nicht weitervermittelt werden kann, auf eine Freigabe des benötigten Wegsegments. In der **Drop-Methode** wird der Vermittlungsversuch beendet, falls eine Blockierung auftritt. In der **modifizierten Drop-Methode** wird der Vermittlungsversuch erst nach einer festgelegten Wartezeit oder bei Überschreiten von Pufferkapazität abgebrochen. In einem System mit Verkehrsumleitung wird bei einer Blockierung versucht, die Nachricht entlang eines anderen verfügbaren Weges weiterzuvermitteln; unter Umständen werden hierbei auch Umwege in Kauf genommen. Methoden der Verkehrsumleitung sind besonders bei der Wegesuche in direkten Netzen von Bedeutung [GlN92].

In einem System mit Durchschaltevermittlung bedeutet die Block-Methode, daß der Verbindungsaufbau durch das Netz fortschreitet, bis die Senke oder ein blockiertes Wegsegment erreicht wird. Auch ein nur teilweise aufgebauter Weg bleibt erhalten, bis durch Freigabe der Blockierung der Verbindungsaufbau fortschreiten kann. Bei der Drop-Methode wird der Versuch des Verbindungsaufbaus abgebrochen, sobald eine Blockierung erfolgt.

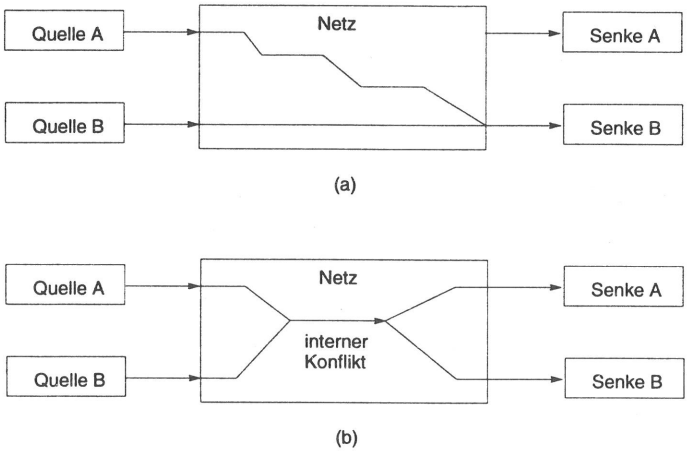


Abbildung 1.13: *(a) Abnehmerblockierung am Netzausgang; (b) Innere Blockierung.*

In paketvermittelnden Netzen werden bei der Block-Methode Pakete in Puffern zwischengepuffert, wenn ein Wegsegment blockiert ist. Ist der Puffer voll, so wird den vorhergehenden Stufen mitgeteilt, keine weiteren Pakete mehr zu senden (**Datenflußsteuerung**). Beim Drop-Verfahren werden Pakete ebenfalls zwischengepuffert, falls ein Wegsegment nicht verfügbar ist. Ist jedoch kein Pufferplatz mehr verfügbar, so wird dies der vorangegangenen Stufe nicht mitgeteilt, so daß Pakete verloren gehen können.

Die Block-Methode wurde im obigen Beispiel beim Wormhole-Routing gezeigt; auch in indirekten Verbindungsnetzen für Parallelrechner wird meist die Block-Methode eingesetzt, um jeglichen Datenverlust zu vermeiden. In der Breitbandübermittlungstechnik wird meist das Drop-Verfahren eingesetzt, da verspätete Datenpakete meist nicht mehr genutzt werden können (z. B. bei der Videoübertragung). Bei Diensten, die keine Datenverluste zulassen (z. B. File-Transfer) müssen auftretende Verluste durch Funktionen höhere Protokoll-Schichten ausgeglichen werden.

1.7 Realisierungsaspekte

1.7.1 Schnittstellenstruktur

1.7.1.1 Schnittstellen in direkten Netzen

Ein direktes Netz kann im einfachsten Fall durch PEs mit je einer Eingangs- und einer Ausgangs-Netzschnittstelle konstruiert werden, wie in Abbildung 1.14 gezeigt. Prozessoren senden Daten

zur Eingangsschnittstelle, in der in einem Demultiplexer eine von K möglichen Verbindungslinks ausgewählt wird, entlang derer Daten von Quelle zu Senke transferiert werden. Im allgemeinen Fall kann die Anzahl der Verbindungslinks sich von PE zu PE unterscheiden, ebenso wie die Anzahl von Ein- und Ausgängen pro PE. Auch sind die PEs vielfach nur mit wenigen anderen PEs direkt verbunden. Für eine Kommunikation zwischen nicht benachbarten Prozessoren sind Zwischenschritte erforderlich. In diesem einfachen Modell entspricht jeder Zwischenschritt einem Transfer der Nachricht über die Eingangsschnittstelle in ein PE und eine Weiterleitung über die Ausgangsschnittstelle.

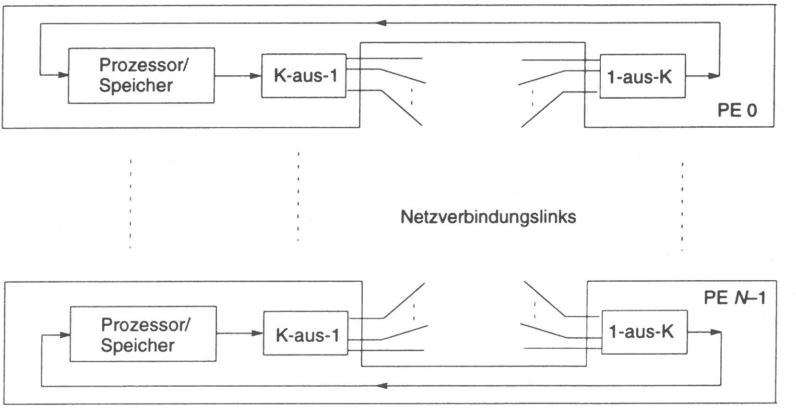


Abbildung 1.14: *Direktes Netz mit Multiplexer-Demultiplexer-Schnittstellen.*

Um den Zeitaufwand für solche Zwischenschritte zu reduzieren, wird die Netzschnittstelle im PE oft als Vermittlungsschalter *(***Router***)* entworfen, der die Eingänge direkt zu den Ausgängen vermitteln kann; der eigentliche Prozessor ist über einen oder mehrere Ein- und Ausgänge an diesen Schalter angeschlossen (siehe Abbildung 1.15). Die Kommunikationsleistung wird so erhöht, und die PEs werden von Vermittlungsaufgaben entlastet. Da in einer solchen Struktur dynamische Schalter in das Netz eingefügt sind, wird der Übergang zwischen "statischen" und "dynamischen" Netzen fließend.

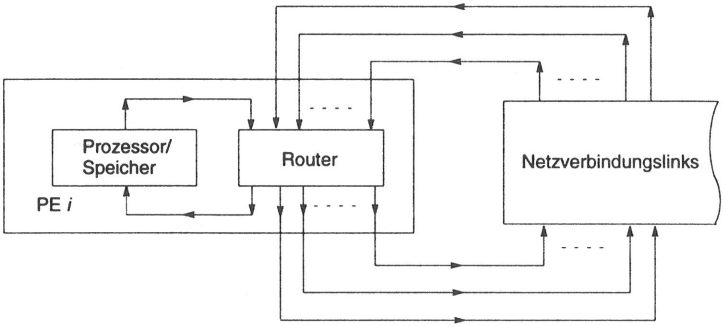


Abbildung 1.15: *Direktes Netz mit Router in der Netzschnittstelle.*

1.7.1.2 Schnittstellen in indirekten Netzen

In den meisten indirekten Netzen hat eine Quelle nur einen einzelnen, möglicherweise bidirektionalen Verbindungslink zum Netz, wobei alle Vermittlungsfunktionen vom Netz selbst ausgeführt werden. Dann sind für die Schnittstellen keine besonderen Hardwarekomponenten wie Multiplexer oder Router erforderlich. In manchen Fällen verfügt das Netz über mehrere Eingänge, wie beispielsweise beim Extra-Stage-Cube-Netz [AdS82], der fehlertoleranten Version des Multistage-Cube-Netzes. Da auch hier jedoch keine Vermittlungsfunktionen ausgeführt werden, sondern nur einer der möglichen Netzzugänge gewählt wird, entspricht die Struktur der in Abbildung 1.14 für direkte Netze gezeigten.

1.7.1.3 Kommunikationsprozessoren

Insbesondere bei Parallelrechnern belastet die Kommunikation aufgrund der zu bearbeitenden Protokolle den Prozessor erheblich. Daher werden vielfach Schnittstellen-Prozessoren eingesetzt, die den Hauptprozessor von diesen Aufgaben entlasten. Dies kann von einfacher Hardware zur Paketierung von Nachrichten bis zu komplexen Prozessoren reichen, die wesentliche Funktionen des Betriebssystems ausführen. Zum Beispiel werden in der Intel Paragon zwei i860XP Prozessoren pro PE verwendet, einer als eigentliche Recheneinheit und der andere als Kommunikationsprozessor [EsK93].

1.7.2 Router- und Koppelelement-Architektur

Router in den Schnittstellen direkter Netze und Koppelelemente indirekter Netze haben ähnliche Hardwareanforderungen. In manchen Fällen eignen sich die gleichen Komponenten für den Ein-

satz in beiden Anwendungsbereichen. In diesem Abschnitt werden einige grundlegende Eigenschaften von Koppelelementen und Routern diskutiert, die zeigen, wie die zuvor diskutierten Eigenschaften wie Kontrollstrategie, Operationsmodus, Vermittlungsverfahren usw. in starkem Maße Einfluß auf die Organisation und Realisierung dieser Bausteine haben.

1.7.2.1 Steuerungsstrategie

In Abhängigkeit von der Steuerungsstrategie des Netzes werden die Zustände eines Koppelelements zentral oder verteilt festgelegt. In einem System mit zentraler Steuerung werden Steuerleitungen zu den Koppelelementen geführt, von denen die Wege durch das Koppelelement bestimmt werden, wie in Abbildung 1.16a für ein $R \times S$-Koppelelement gezeigt. In einem System mit verteilter Steuerung werden Informationen der R Eingänge der Steuerlogik des Koppelelements zugeführt, die daraufhin den Schaltzustand festlegt (Abbildung 1.16b).

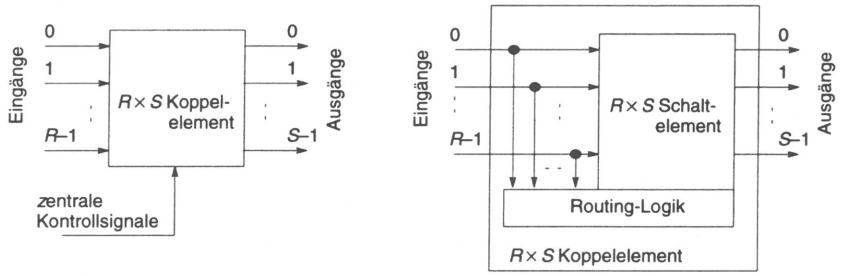


Abbildung 1.16: *Koppelelement mit (a) zentraler und (b) verteilter Steuerung.*

1.7.2.2 Verbindungsstruktur

Router und Koppelelemente können unterschiedlichste Komplexitäten und Vermittungsmöglichkeiten aufweisen, teilweise aufgrund der Topologie, für die sie eingesetzt werden, teilweise aber auch aufgrund der internen Konstruktion. Diese Vermittungsmöglichkeiten bestimmen auch die Wege, die durch das gesamte Netz aufgebaut werden können; je flexibler die individuellen Koppelelemente, desto flexibler ist auch das Netz.

In Abbildung 1.10 wurden die Zustände eines typischen 2×2 ungepufferten Koppelelements gezeigt, wie es in vielen Netzen genutzt wird. Auch bei Netzen, die Koppelelemente mit mehr als zwei Eingängen erfordern, ist die Grundstruktur in vielen Fällen ein Crossbar-Netz, das sämtliche Permutationen zwischen Ein- und Ausgängen erlaubt. Alternativ dazu kann ein solches Koppelelement auch hierarchisch aus kleineren Elementen aufgebaut werden, wie in Abbildung 1.17a für vier Ein- und Ausgänge gezeigt. Ein solches Element kann unter Umständen nicht alle

Permutationen ausführen; so ist im Beispiel eine gleichzeitige Verbindung der Ein/Ausgangspaare 0 zu 1 und 1 zu 0 nicht möglich, da intern die gleiche Zwischenleitung benutzt werden müßte.

Während bei den obigen Koppelelementen alle Eingänge gleichzeitig Daten zu Ausgängen leiten können, stellt ein **Selektor-Schalter** immer nur einen einzelnen Weg zwischen einem der Eingänge und einem der Ausgänge zur Verfügung. Solche Schaltelemente werden in Datenmanipulatoren [McS82] und Dynamic-Redundancy-Netzen [JeS88] verwendet, die zu der Klasse der fehlertoleranten Netze gehören. Eine mögliche Struktur mit einem Multiplexer und einem Verteiler ist in Abbildung 1.17b gezeigt; aufgrund des einfachen Aufbaus ist die Realisierung in der Regel einfacher als die der weiter oben erwähnten Koppelelemente.

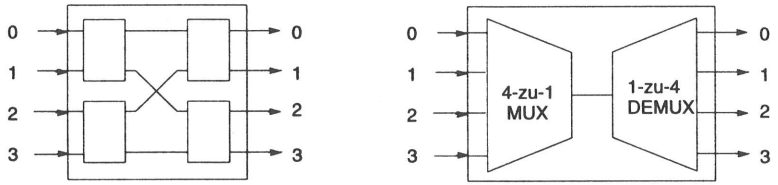


***Abbildung 1.17:** (a) hierarchisches Koppelelement; (b) Selektor-Schalter.*

1.7.3 Technologische Realisierung

Die Geschwindigkeit und Leistungsfähigkeit eines Netzes wird ganz wesentlich von der Schaltungsstruktur und der verwendeten Technologie bestimmt. Dies umfaßt einen weiten Bereich, von der physikalischen Anordnung über Zwischenleitungsführung bis zur VLSI-Technologie.

Einerseits erweitern neue Technologien die Bandbreite der Realisierungsmöglichkeiten. Ein Beispiel ist die VLSI-Technologie, die durch erheblich gesteigerte Schaltungskomplexitäten die Realisierung vieler effizienter Netz- und Koppelelement-Architekturen erlaubt, die mit diskreten Aufbauten nicht möglich wären. Dies wird in Kapitel 3 anhand der Realisierung von Koppelelementen verdeutlicht.

Auf der anderen Seite eröffnen neue Technologien zwar großes Potential zur Leistungssteigerung, führen gleichzeitig aber zu restriktiven Randbedingungen. So ist bei optischen Netzen zwar eine wesentlich höhere Übertragungsgeschwindigkeit als bei elektronischen Schaltkreisen möglich, jedoch bestehen starke Beschränkungen beispielsweise bei den möglichen Topologien und im Bereich der Zwischenpufferung [Gre92, KiM91].

1.8 Datenverkehre in Verbindungsnetzen

Die Leistung eines Netzes ist stark abhängig vom zu vermittelnden Datenverkehr. Der Datenverkehr hängt wiederum von der Applikation ab, in der das Netz betrieben wird. In einem Parallelrechner z. B. wird der Datenverkehr durch die Ausführung eines Applikationsprogramms oder durch Steuermechanismen wie Prozessorsynchronisation [Din89] oder Multithreading [Aga92, LeH94] verursacht. In Kommunikationssystemen können die Datenströme z. B. durch Sprach-, Video- oder Fileübertragung erzeugt werden. Da der zu erwartende Verkehr nur in seltenen Fällen hinreichend genau bekannt ist, ist eine exakte Beschreibung der Verkehre nur sehr selten möglich. Um die Leistung eines Netzes dennoch beurteilen zu können, müssen Verkehrsmodelle erstellt werden, die den tatsächlich auftretenden Verkehr möglichst gut modellieren.

Bei der Erzeugung eines Modellverkehrs sind zwei voneinander unabhängige Parameter zu berücksichtigen: die Verkehrsrate und die Verbindungsverteilung. Bei der **Verkehrsrate** wird bestimmt, in welchem zeitlichen Abstand bzw. in welcher statistischen Verteilung Nachrichten in das Netz gesandt werden. Die **Verbindungsverteilung** legt fest, welche Quellen und Senken miteinander kommunizieren.

Ein einfaches Modell der Verbindungsverteilung ist der **gleichverteilte Verkehr**. Kann eine Quelle N mögliche Ziele erreichen, so ist in diesem Verkehrsmodell eine Nachricht zu jedem Ziel mit gleicher Wahrscheinlichkeit gerichtet. Auch wird eine symmetrische Belastung der Netzeingänge angenommen. Somit beträgt die Wahrscheinlichkeit $1/N$, daß eine Nachricht zu einem bestimmten Ziel geleitet wird, und alle Ziele werden im statistischen Mittel gleich häufig angesprochen werden. Diese Verteilung wird insbesondere bei analytischen Methoden zur Leistungsbewertung von Netzen weit verbreitet angewendet (z. B. in [Bhu85, DiJ81, KrS88, MuY94, ThR91, YoL90]). Für Kommunikationssysteme ist dieses Modell meist eine gute Annäherung des tatsächlichen Verkehrs, bei Parallelrechnern ist es jedoch problematisch. Dort findet sich oft eine Ungleichverteilung der Zieladressen, die z. B. durch Permutationen oder Synchronisationen entstehen und zu **ungleichverteilten Verkehren** (**schiefe Verkehre**) führen [LaK88, Len78, PfN85]. In Verbindungsnetzen können unter diesen Verkehren vermehrt Blockierungen auftreten, die zu Leistungseinbrüchen in den Netzen und somit in den Parallelrechnersystemen führen [Har91, KiL93, LaK88, YeT87].

Die hier angesprochenen Modelle können beliebig kombiniert werden. Häufig wird die Kombination eines gleichverteilten Hintergrundverkehrs mit einer ungleichverteilten Verkehrsart betrachtet, um den gegenseitigen Einfluß der beiden Verkehre zu beurteilen [KiL93, KuT91, LaK88].

Die wichtigsten Verkehre bzw. Verkehrsmixe und deren Modellierung sind in Kapitel 4 näher beschrieben.

1.9 Übersicht über die Arbeit

Diese Arbeit beschäftigt sich mit Methoden zur Modellierung, Klassifizierung und Leistungscharakterisierung von Verbindungsnetzen aus der Klasse der indirekten Netze. Hierbei wird insbesondere das Verhalten der Netze unter ungleichverteilten Datenverkehren untersucht.

Kapitel 1 gibt eine Übersicht über den Stand der Technik im Bereich der Verbindungsnetze für Parallelrechner und Breitbandkommunikationssysteme, während in Kapitel 2 topologische und architekturelle Aspekte einer Klasse von Verbindungsnetzen, den indirekten Netzen, vertiefend diskutiert werden. Architekturen und Realisierungsaspekte von Koppelelementen, aus denen indirekte Netze aufgebaut sind, werden im Kapitel 3 vorgestellt. Bei einer Leistungsbewertung von Netzen müssen auch die zu vermittelnden Datenströme berücksichtigt werden. Deshalb stellt Kapitel 4 wichtige Datenverkehre und deren Abstraktion durch Datenverkehrsmodelle vor.

In Kapitel 5 werden Leistungsparameter von Verbindungsnetzen und Konzepte zu deren Bestimmung vorgestellt. Neben analytischen Verfahren werden, insbesondere bei komplexeren Systemen, Simulationstechniken zur Leistungsbestimmung herangezogen, wobei die parallele Simulation durch ihre gesteigerte Leistungsfähigkeit hierzu besonders gut geeignet ist. Allerdings hat die Modellierung der zu simulierenden Netze und die Abbildung der Simulationsalgorithmen auf den Parallelrechner einen entscheidenden Einfluß auf die zu erreichende Simulatorleistung. Deshalb werden verschiedene Abbildungen von Netzsimulationsmodellen auf SIMD-Parallelrechner zur Implementierung paralleler zeitgesteuerter Netzsimulatoren untersucht. Dies erlaubt eine Klassifizierung und Optimierung der verschiedenen Abbildungsstrategien basierend auf Schnelligkeit, Speicherverbrauch und Kommunikationsanforderungen.

Unter Zuhilfenahme des parallelen Simulators werden in Kapitel 6.1 Effekte von statisch und temporär ungleichverteilten Datenverkehren in mehrstufigen store-and-forward-paketvermittelnden Delta-Netzen untersucht. Hieraus werden Kriterien zur Leistungsbewertung von Netzen, die auch den transienten Charakter dieser Verkehre berücksichtigen, abgeleitet. In Kapitel 6.2 werden Methoden zur Verringerung von Leistungseinbrüchen in Delta-Netzen unter ungleichverteilten Verkehren diskutiert. Aus den Untersuchungen wird in Kapitel 6.3 ein Konzept zur Klassifizierung von Netzen und Koppelelement-Architekturen unter statisch und temporär ungleichverteilten Datenverkehren abgeleitet. Dieses Konzept der Head-of-Line-Blockierungen (HOL-Blockierungen) höherer Ordnung klassifiziert Netze nach der minimalen Länge von Wegen durch das Netz, die keine gemeinsamen Puffer besitzen. Das HOL-Blockierungskonzept kann für die Klassifizierung und analytische Leistungsbewertung, sowie für die Leistungsoptimierung von bekannten wie auch neuen Netzarchitekturen verwendet werden.

Die Untersuchungen von Verbindungsnetzen unter ungleichverteilten Datenverkehren haben gezeigt, daß Leistungseinbrüche durch die gemeinsame Pufferung von "heißen" und "kalten" Datenpaketen in den Koppelelementpuffern zustande kommen. Um dies zu vermeiden, ist die Architektur eines Prioritätskoppelelements entwickelt worden, die in Kapitel 6.4 vorgestellt

wird. Sie zeichnet sich durch eine getrennte Pufferung von gleichverteilten und ungleichverteilten Datenpaketen aus. Mit dieser Architektur ist es möglich, Leistungseinbrüche in Verbindungsnetzen unter temporär ungleichverteilten Datenverkehren zu minimieren.

Videodienste in Kommunikationssystemen stellen besondere Anforderungen an das ATM-Verbindungsnetz. Die vor der Verbindungsannahme ausgehandelten QoS-Parameter (QoS = Quality of Service, Dienstgüte) müssen in engen Grenzen eingehalten werden. Hierbei müssen unter Umständen Verbindungsanfragen anderer Datenströme abgewiesen werden, da ansonsten die QoS-Parameter der Videoströme nicht mehr garantiert werden können. Eine Möglichkeit, die Qualitätsparameter einer Videoübertragung garantiert einhalten zu können, ohne andere Verbindungen abzuweisen, ist die gesonderte Übertragung des Videostromes durch das ATM-Netz, wobei der Videostrom nicht durch die anderen Datenströme beeinflußt wird. Dies kann effizient mit dem in dieser Arbeit entwickelten Prioritätskoppelelement durchgeführt werden. Die erreichbare Leistungssteigerung wird anhand von MPEG-codierten Videoströmen in ATM-Netzen in Abschnitt 6.4.3 nachgewiesen.

In Kapitel 7 werden schließlich die Ergebnisse der Arbeit zusammengefaßt und ein Ausblick auf weiterführende Untersuchungen gegeben.

2 Indirekte Verbindungsnetze

Wie in Abschnitt 1.5.2 diskutiert, verfügen indirekte Verbindungsnetze über eine oder mehrere Stufen von Koppelelementen, durch die Verbindungen zwischen Quellen und Senken hergestellt werden. Im Gegensatz zu direkten Netzen, bei denen die Vermittlungsfunktion ausschließlich in den Knoten der Quellen bzw. Senken durch Auswahl der geeigneten Anschlußleitung ausgeführt wird, werden im indirekten Netz die Wege durch Koppelelemente innerhalb des Netzes festgelegt. Daher haben Quellen und Senken bei vielen indirekten Netzen nur eine Anschlußleitung in das Netz.

2.1 Modell indirekter Netze

Indirekte Netze bestehen aus einer oder mehreren Stufen von Koppelelementen; Abbildung 2.1 zeigt ein allgemeines Modell eines indirekten Verbindungsnetzes. In einem Netz für Parallelrechner entspricht ein Eingang bzw. Ausgang des Netzes einer Anschlußleitung zu den PEs, Prozessoren oder Speichern. Oft sind solche Leitungen wegen erhöhter Leistungsfähigkeit oder Fehlertoleranz mehrfach ausgeführt [AdS84a, AdS84b, SzH87].

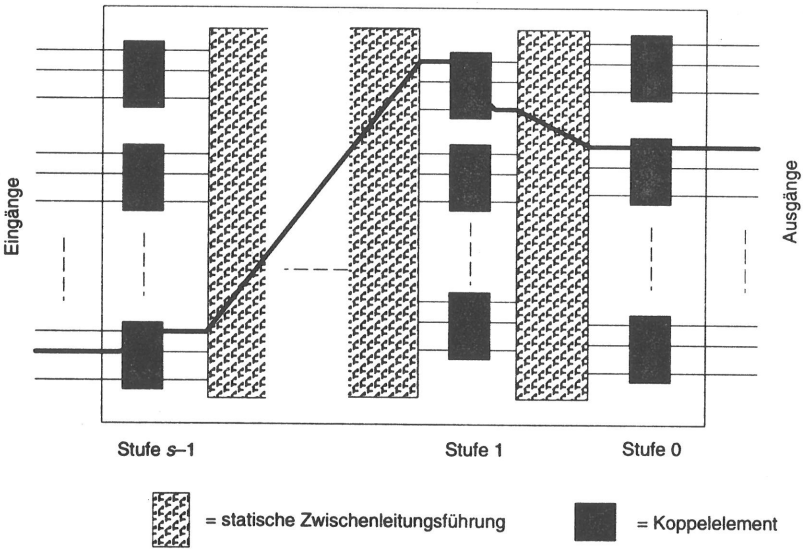


Abbildung 2.1: *Allgemeines Modell eines mehrstufigen indirekten Verbindungsnetzes.*

Das Netz besteht aus s Stufen von Koppelelementen; Stufe j ($0 \leq j < s$) enthält s_j Koppelelemente. Bei einem einstufigen indirekten Netz (z. B. beim einstufigen Shuffle-Exchange-Netz, siehe Abschnitt 1.5.2) ist $s = 1$, bei mehrstufigen indirekten Netzen (**Multistage Interconnection Network, MIN**) ist $s \geq 2$. Die Stufen sind über statische Zwischenleitungen miteinander verbunden, die zwischen den Stufen gleich oder unterschiedlich geführt sein können. Sind alle Koppelelemente und die Zwischenleitungsführung zwischen den Stufen identisch, spricht man von einem **homogenen Multistage-Netz**. Ein Koppelelement der Stufe j hat E_j Eingänge und A_j Ausgänge. Im allgemeinen Fall unterscheiden sich die Koppelelemente von Stufe zu Stufe oder sogar innerhalb jeder Stufe; in vielen Netzen sind jedoch alle Koppelelemente identisch. Die Klasse der indirekten Netze kann anhand der Anzahl der disjunkten Wege von einer Quelle zu einer Senke weiter unterteilt werden: in die **indirekten Einpfadnetze** (Netze, bei denen genau ein Weg zwischen einer Quelle und einer Senke besteht) und die **indirekten Mehrpfadnetze** (Netze, bei denen mehrere disjunkte Wege zwischen einer Quelle und einer Senke bestehen).

2.2 Indirekte Einpfadnetze

Indirekte Einpfadnetze werden allgemein durch die Klasse der Banyan-Netze repräsentiert. Netze aus einer Unterklasse der Banyan-Netze, der SW-Banyan Netze, in denen das Routing der Daten verteilt durch Bitstellen der Paketadressen gesteuert wird, gehören zur Klasse der Delta-Netze. Symmetrische Delta-Netze (Delta-Netze, die aus symmetrischen $B \times B$ Koppelelementen aufgebaut sind), werden bevorzugt in Parallelrechnern und in Kommunikationssystemen eingesetzt. Die Klasse der symmetrischen Delta-Netze mit B=2 umfaßt mehrere topologisch äquivalente Netze, die unter den Namen Generalized-Cube-Netz, Omega-Netz, Indirektes Binary-n-Cube-Netz und Baseline-Netz bekannt geworden sind. Diese Netzklassen und Netze werden in den nächsten Abschnitten kurz vorgestellt.

2.2.1 Banyan-Netze

Banyan-Netze, eingeführt von Goke und Lipovski [GoL73], sind durch deren Graphenrepräsentation definiert. Der Graph eines Banyan-Netzes besteht aus drei verschiedenen Knoten: dem **Basisknoten** (einem Knoten ohne Eingangskanten), dem **Zwischenknoten** und dem **Apex-Knoten** (einem Knoten ohne Ausgangskanten). Die fundamentale Eigenschaft des Banyan-Graphen ist, daß exakt ein Pfad von jedem Basisknoten zu jedem Apex-Knoten existiert. Ein Beispiel eines Banyan-Graphen ist in Abbildung 2.2a gezeigt. Aus einem Banyan-Graphen kann ein Banyan-Netz konstruiert werden, indem jeder Graphenknoten durch ein Koppelelement und jede Graphenkante durch eine Zwischenleitung ersetzt wird (siehe Abbildung 2.2b). Jeder Knoten eines Banyan-Graphen kann unterschiedlich viele Eingangs- bzw. Ausgangskanten besitzen.

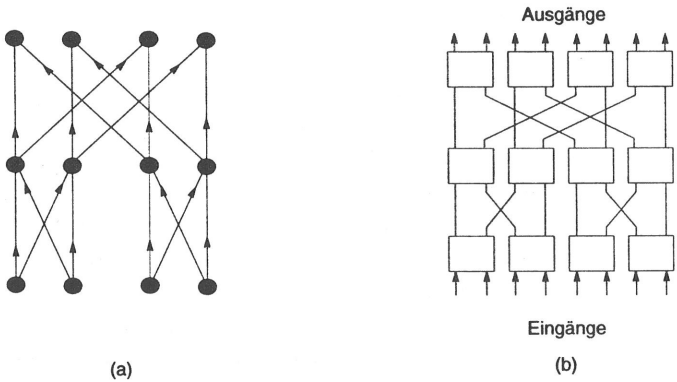


Abbildung 2.2: (a) Banyan-Graph; (b) korrespondierendes Banyan-Netz.

Eine Unterklasse der Banyan-Netze, die Klasse der **SW-Banyan-Netze**, zeichnet sich durch die ausschließliche Benutzung von $a \times b$ Koppelelementen aus.

2.2.2 Delta-Netze

Delta-Netze [Pat81] sind eine allgemeine Klasse von SW-Banyan-Netzen, in denen das Routing in den Koppelelementen durch ein oder mehrere Bits der Paketadressen gesteuert wird (**digit-controlled routing**). Ein Delta-Netz besteht allgemein aus s Stufen und benutzt $a \times b$ Koppelelemente und verbindet a^s Eingänge mit b^s Ausgängen. Die Leitungsführung zwischen den Stufen kann beliebig gewählt werden, solange zwei Bedingungen erfüllt sind: 1) es existiert genau ein Pfad von jedem Netzeingang zu jedem Netzausgang, und 2) die Koppelelemente in Stufe k können durch die k-te Stelle der Paketadresse, ausgedrückt in Radix-b Arithmetik, gesteuert werden. In Abbildung 2.3 ist ein Delta-Netz mit a=3, b=4 und s=2 gezeigt, das 9 Eingänge mit 16 Ausgängen verbinden kann.

Delta-Netze berücksichtigen Koppelelemente beliebiger Größe, die als Crossbar-Netze aufgebaut werden. Um eine einfache Hardware-Realisierung zu erhalten, wurden in vielen Fällen Delta-Netze mit 2×2 Koppelelementen benutzt. Ein wichtiger Vertreter dieser Netzklasse ist das **Generalized Cube-Netz (GC-Netz)**. Diese Netze und ihre Modifikationen werden in vielen bestehenden und vorgeschlagenen Parallelrechnern eingesetzt [Bat82, Sie90, SiN89]. Deshalb werden in den nächsten Abschnitten die Topologie und das Routing in GC-Netzen näher erläutert. Wie in Abschnitt 2.2.2.4 gezeigt wird, sind viele Netze, die unter anderem Namen bekannt sind, topologisch mit den Generalized Cube-Netzen äquivalent.

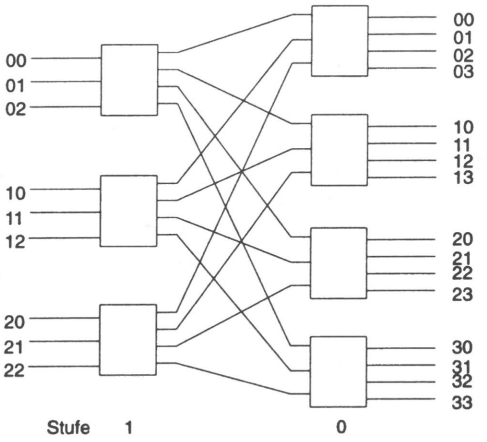


***Abbildung 2.3:** Delta-Netz mit a=3, b=4 und s=2.*

2.2.2.1 Topologie

Das mehrstufige GC-Netz mit N Ein- und Ausgängen besteht aus $s = \log_2 N$ Stufen mit jeweils $N/2$ Koppelelementen. Vom Eingang zum Ausgang sind die Stufen in absteigender Reihenfolge numeriert. Eingänge und Ausgänge sowie die Leitungen zwischen den Stufen sind mit den Zahlen von 0 bis N–1 indiziert. Abbildung 2.4 zeigt ein mehrstufiges Cube-Netz für N=8. Die Koppelelemente entsprechen denen des einstufigen indirekten Shuffle-Exchange-Netzes und können Daten damit geradeaus (*straight*) oder kreuzweise (*exchange*) transportieren. Im allgemeinen Fall ist auch eine Broadcast-Funktion möglich, mit der Daten von einem Eingang des Koppelelements zu beiden Ausgängen vermittelt werden können. Von der schaltungstechnischen Realisierung des Netzes hängt es ab, ob Broadcast-Funktionen unterstützt werden.

Die Koppelelemente und damit der Datentransport können durchschalte- oder paketvermittelnd ausgeführt sein und sich dabei der Wormhole-Routing-Methode oder des Virtual Cut-Through bedienen (siehe Abschnitt 1.6.4). Diese Eigenschaften haben auf die hier diskutierten grundsätzlichen Eigenschaften wie Topologie und Wegesuche keinen unmittelbaren Einfluß.

Die Indizes der Zwischenleitungen an einem Koppelelement unterscheiden sich immer in genau einem Bit, und zwar in Stufe k in Bit k. Der obere Ein- bzw. Ausgang eines Koppelelements in der Stufe k besitzt also den Index $P_O = p_{s-1}\ p_{s-2}\ \cdots\ p_k\ \cdots\ p_1\ p_0$, während der untere Ein- bzw. Ausgang den Index $P_U = p_{s-1}\ p_{s-2}\ \cdots\ \bar{p}_k\ \cdots\ p_1\ p_0$ hat. Ein Ausgang der Stufe k mit dem Index P ist immer mit dem Eingang P der darauffolgenden Stufe k–1 verbunden. Bei einer *straight*-Operation bleibt der Index also erhalten, und die *exchange*-Operation entspricht der **cube$_k$-Funktion** (der Invertierung des k-ten Bits der Adresse). Da jeder Bitposition der Paketadresse eine Stufe entspricht, können alle s Cube-Funktionen in einem Durchlauf durchgeführt

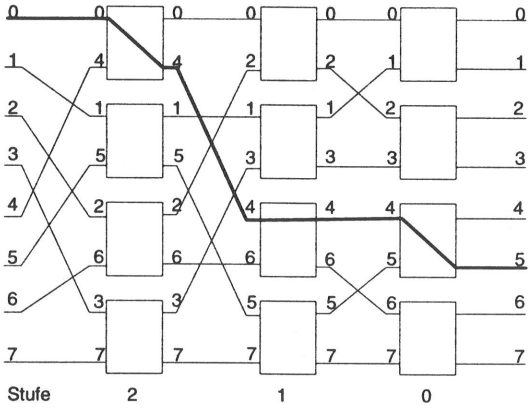


Abbildung 2.4: *Topologie des Generalized-Cube-Netzes für N=8 .*

werden, so daß ein Transport von jeder Quelle zu jeder Senke möglich ist. Aus dem gleichen Grund existiert nur ein Weg zwischen einem Netzein- und -ausgang, da jede Cube-Funktion bei einem Durchlauf nur in genau einer Stufe ausgeführt wird.

Abbildung 2.4 zeigt ein Beispiel eines Datentransports von Quelle $Q = q_{s-1}\ q_{s-2}\ \dots\ q_1\ q_0 = 0$ zu Senke $S = s_{s-1}\ s_{s-2}\ \dots\ s_1\ s_0 = 5$. In Stufe k, und nur dort, kann Bit k modifiziert werden. Daher muß in dieser Stufe das Bit q_k auf das Bit s_k abgebildet werden. Aufgrund der Reihenfolge der Stufen wird zunächst das höchstwertige Bit angepaßt, so daß die Nachricht das Koppelelement der Stufe s–1 am Ausgang $s_{s-1}\ q_{s-2}\ \dots\ q_1\ q_0$ verläßt; dieses Verfahren setzt sich über alle Stufen fort, so daß allgemein eine Nachricht den Eingang $s_{s-1}\ s_{s-2}\ \dots\ s_{k+1}\ q_k\ q_{k-1}\ \dots\ q_1\ q_0$ und den Ausgang $s_{s-1}\ s_{s-2} \dots\ s_{k+1}\ s_k\ q_{k-1}\ \dots\ q_1\ q_0$ eines Koppelelements in Stufe k belegt.

2.2.2.2 Routing

Das mehrstufige Cube-Netz benötigt keine zentrale Kontrolle, sondern kann durch die zu vermittelnden Nachrichten verteilt gesteuert werden. Hierbei wird das **XOR-Routing** und das **zielgerichtete Routing (destination tag routing)** unterschieden. Im folgenden wird auf das Routing für 1-zu-1 Verbindungen eingegangen; Routingalgorithmen für Broadcast-Verbindungen können z. B. in [Sie90] nachgelesen werden.

Die Quelle einer 1-zu-1 Verbindung sei $Q = q_{s-1}\ q_{s-2}\ \dots\ q_1\ q_0$, und die Senke sei $S = s_{s-1}\ s_{s-2}\ \dots\ s_1\ s_0$. Um die Verbindung aufzubauen, wird eine Nachricht (**Routing Tag**) in das Netz gesandt, durch die ein Pfad aufgebaut wird. Bei einem durchschaltevermittelnden Netz werden Daten erst gesandt, nachdem der gesamte Pfad aufgebaut wurde. Bei paketorientierten Netzen ist das Routing Tag der erste Teil des Pakets, das von Stufe zu Stufe durch das Netz transportiert wird.

XOR-Routing

Beim XOR-Routing wird das s-Bit Routing Tag durch die Exklusiv-Oder-Verknüpfung von Q und S gebildet: $T = Q \oplus S = t_{s-1}\ t_{s-2} \cdots t_1\ t_0$.

Ein Bit t_k des Routing Tag ist also 0 wenn $q_k = s_k$, und gleich 1, falls $q_k \neq s_k$. Ein Koppelelement in Stufe k untersucht das Bit t_k. Ist $t_k = 1$, so wird das Koppelelement auf exchange gestellt; damit wird die Nachricht vom Eingang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ q_k\ q_{k-1} \cdots q_1\ q_0$ zum Ausgang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ \bar{q}_k\ q_{k-1} \cdots q_1\ q_0$ transportiert. Da wegen $t_k=1$ auch $q_k \neq s_k$ ist, ist dies der gewünschte Ausgang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ s_k\ q_{k-1} \cdots q_1\ q_0$. In gleicher Weise wird bei $t_k = 0$ das Koppelelement auf straight gestellt, und die Nachricht wird ebenfalls zum Ausgang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ s_k\ q_{k-1} \cdots q_1\ q_0$ geleitet.

Als Beispiel soll eine Verbindung vom Eingang 0 zum Ausgang 5 dienen, wie in Abbildung 2.4 gezeigt. Damit ist das Routing Tag $T = Q \oplus S = 000_2 \oplus 101_2 = 101_2$. Stufe 2 wird aufgrund von T auf exchange gestellt, so daß der Ausgang 100_2 erreicht wird. Stufe 1 ist straight, so daß auch hier der Ausgang 100_2 benutzt wird. Stufe 0 ist wiederum exchange, und die korrekte Senke 101_2 ist erreicht.

Ein wesentlicher Vorteil des XOR-Routing ist die Möglichkeit, in der Senke die Quelle mit Hilfe des Routing Tags zu bestimmen. Da definitionsgemäß $T = Q \oplus S$ ist, gilt auch $Q = S \oplus T$. Insbesondere bei Parallelrechnern mit gemeinsamem Speicher ist dies von Vorteil, da bei jedem Lese-Zugriff der Inhalt einer Speicherzelle an die Quelle zurückgesandt werden muß, und die Quelladresse nicht als Teil der Nachricht mitgeschickt werden muß.

Aufgrund des XOR-Tags kann nicht festgestellt werden, ob eine Nachricht, die an einer Senke empfangen wurde, tatsächlich an diese Senke adressiert war. Falls eine Nachricht innerhalb des Netzes fehlgeleitet wurde, wird dies nicht im Tag widergespiegelt, und bestimmte Hardware-Probleme können so nicht unmittelbar erkannt werden. Sollen fehlgeleitete Pakete sofort erkannt werden, muß der Nachricht die Adresse der Quelle oder der Senke mitgegeben werden; aufgrund dieser Adresse und des XOR-Tags kann die gewünschte Überprüfung erfolgen.

Destination-Tag-Routing

Beim Destination-Routing wird die Zieladresse als s-Bit Routing Tag genutzt. Wie im XOR-Routing untersucht ein Koppelelement in Stufe k das Bit t_k. Ist $t_k = d_k = 0$, so wird die Nachricht zum oberen Ausgang vermittelt, dessen Index in Bit k eine 0 hat. Dies bedeutet, daß das Koppelelement entweder auf *straight* oder *exchange* gestellt wird, je nachdem, ob die Nachricht am oberen oder unteren Eingang des Koppelelements eintrifft. Ist $t_k = s_k = 1$, so wird die Nachricht zum unteren Ausgang vermittelt. Damit wird in beiden Fällen die Nachricht vom Eingang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ q_k\ q_{k-1} \cdots q_1\ q_0$ zum korrekten Ausgang $s_{s-1}\ s_{s-2} \cdots s_{k+1}\ s_k\ q_{k-1} \cdots q_1\ q_0$ transportiert.

Im Beispiel von Abbildung 2.4 mit einer Verbindung vom Eingang 0 zum Ausgang 5 ist das Destination Routing Tag $T = S = 101_2$. In Stufe 2 trifft die Nachricht am oberen Eingang ein. Da

t_2 =1, wird die Nachricht zum unteren Ausgang vermittelt und das Koppelelement damit auf exchange gesetzt. In Stufe 1 trifft die Nachricht am oberen Eingang ein; da $t_1 = 0$, muß der obere Ausgang genutzt werden, und eine straight-Stellung resultiert. In Stufe 0 wird der obere Eingang des Koppelelements genutzt, und wegen $t_0 = 1$ folgt die Exchange-Stellung, so daß die korrekte Senke 101_2 erreicht wird.

Beim Destination Tag Routing kann zwar anhand des Tags unmittelbar erkannt werden, ob eine Nachricht innerhalb des Netzes fehlgeleitet wurde, jedoch ist keine Aussage über die Adresse der Quelle möglich. Somit muß in der Regel die Quelladresse als Teil der Nachricht vermittelt werden.

2.2.2.3 Permutationen

Ein mehrstufiges Cube-Netz kann durch Setzen aller Koppelelemente Permutationen ausführen. Abbildung 2.5 zeigt eine Permutation, bei der Eingang K mit Ausgang $K + 2$ modulo N verbunden ist. Nicht alle Permutationen sind möglich; bei einer Permutation, in der z. B. Daten zwischen 0 und 2 sowie 4 und 0 ausgetauscht werden, müssen Daten von beiden Eingängen 0 und 4 zum gleichen Ausgang des obersten Koppelelements in Stufe 2 geleitet werden. Dies verursacht einen Verbindungskonflikt, so daß keine Permutation mit diesen beiden Quellen/Senkenpaaren möglich ist.

Die Anzahl der möglichen Permutationen in einem mehrstufigen Cube-Netz kann einfach abgezählt werden. Das Netz besteht aus $n = \log_2 N$ Stufen mit jeweils $N/2$ Koppelelementen. Jedes Koppelelement hat zwei Permutationsstellungen (straight und exchange); da zwischen einer Quelle und einer Senke nur ein Weg besteht, resultiert jede Einstellung in einer unterschiedlichen Permutation. Dies ergibt insgesamt $2^{nN/2}$ mögliche Permutationen, also wesentlich weniger als die $N!$ Permutationen, die zwischen N Eingängen und Ausgängen möglich sind.

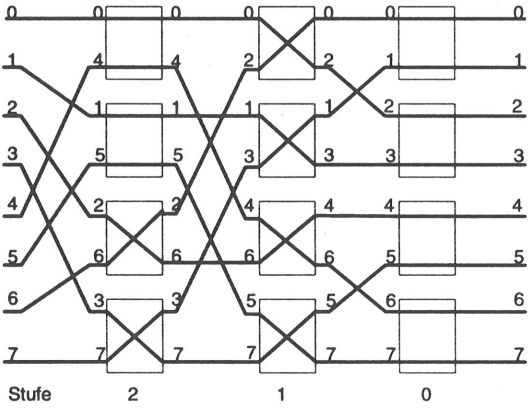


Abbildung 2.5: *Permutation von Eingang K zu Ausgang (K+2)mod 8 im GC-Netz mit N=8.*

Insbesondere bei Ausnutzung der VLSI-Integration ist ein Delta-Netz mit 4×4 bis 32×32 Koppelelementen einem mehrstufigen Cube-Netz mit 2×2 Elementen überlegen, da es mehr Permutationen als das GC-Netz ausführen kann. Der Unterschied kann an der Konzeption der Koppelelemente erläutert werden. In Abbildung 2.6 ist ein Ausschnitt zweier Stufen eines GC-Netzes gezeigt. Dieser Block besteht aus vier Koppelelementen und hat vier Eingänge und vier Ausgänge. Eine gleichzeitige Verbindung zwischen Eingang 0 und Ausgang 2 sowie Eingang 2 und Ausgang 3 ist nicht möglich, da bei beiden Wegen der Ausgang 2 der ersten Stufe benötigt wird. In einem Delta-Netz sind die vier Ein- und Ausgänge über einen Crossbar verbunden, bei dem grundsätzlich alle Permutationen möglich sind. Daher ist die Menge der Permutationen des Delta-Netzes eine Übermenge der erlaubten GC-Permutationen.

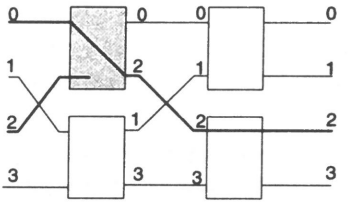


***Abbildung 2.6:** Konflikt in einem Generalized Cube-Netz mit N=4, der durch ein Crossbar-Netz mit N=4 ausgeführt werden kann.*

2.2.2.4 Topologische Äquivalenz von Netzen

Das Generalized-Cube-Netz ist ein Vertreter einer Reihe von topologisch äquivalenten Netzen, die aus $\log_2 N$ Stufen von 2×2-Koppelelementen aufgebaut sind. Beispiele äquivalenter Systeme sind die Omega, Indirect Binary n-Cube und die Baseline-Netze, die im folgenden kurz beschrieben werden.

Das **Omega-Netz** [Law75] (Abbildung 2.7) ist topologisch äquivalent zum Generalized Cube Netz, wie in [SiS78, Sie79] gezeigt. Dieses Netz besteht aus $s = \log_2 N$ Stufen von 2×2-Koppelelementen, wobei die Leitungsführung zwischen zwei Stufen der Shuffle-Exchange-Funktion entspricht. Die Indizierung der Eingänge von Koppelelementen ist in allen Stufen gleich; die beiden Eingänge eines Koppelelements unterscheiden sich immer in Bit 0. Durch Vertauschung der Koppelelemente A und B und nachfolgender Umnumerierung der Links kann das Omega-Netz in das Generalized-Cube-Netz überführt werden [Sie79].

Das **Indirect Binary *n*-Cube Netz** [Pea77] (Abbildung 2.8) entspricht dem Generalized Cube Netz, bis auf eine vertauschte Reihenfolge der Stufen, und somit kann auch dieses Netz alle 1-zu-1 Verbindungen in einem Durchlauf durch das Netz ausführen. Die Netze unterscheiden sich jedoch in den zulässigen Permutationsverbindungen. So kann das GC-Netz zum Beispiel

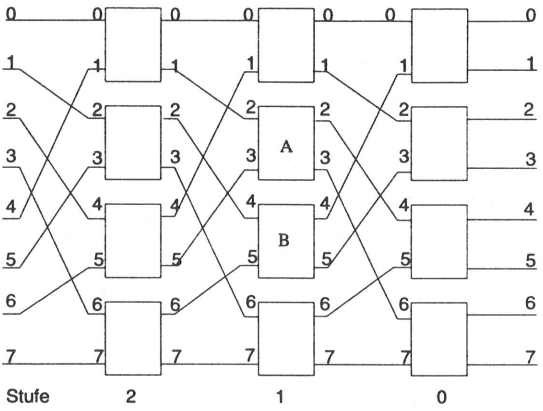


Abbildung 2.7: Omega-Netz mit N=8.

eine Permutation zwischen Eingang 2 und Ausgang 3 sowie 3 und 1 ausführen; im Indirect-Binary-*n*-Cube-Netz führt dies zu einem Konflikt, so daß diese Permutation nicht zulässig ist (Abbildungen 2.9a und b). Kann das Generalized-Cube-Netz die Permutation f ausführen, so kann aufgrund der vertauschten Reihenfolge der Stufen das IBNC-Netz die Permutation f^{-1} ausführen, bei der die Rollen von Eingängen und Ausgängen vertauscht sind. Durch eine Umnumerierung der Ein- und Ausgänge des IBNC-Netzes kann die topologische Äquivalenz zum GC-Netz gezeigt werden [Sie79].

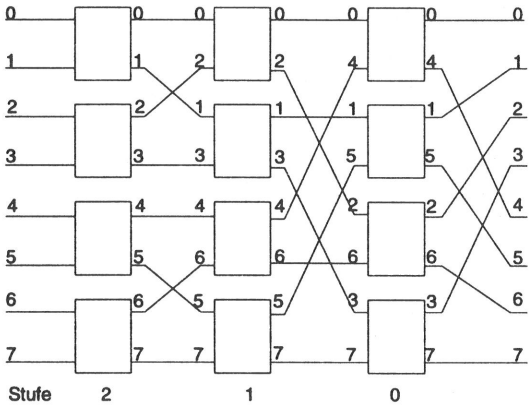


Abbildung 2.8: Indirect-Binary-n-Cube-Netz mit N=8.

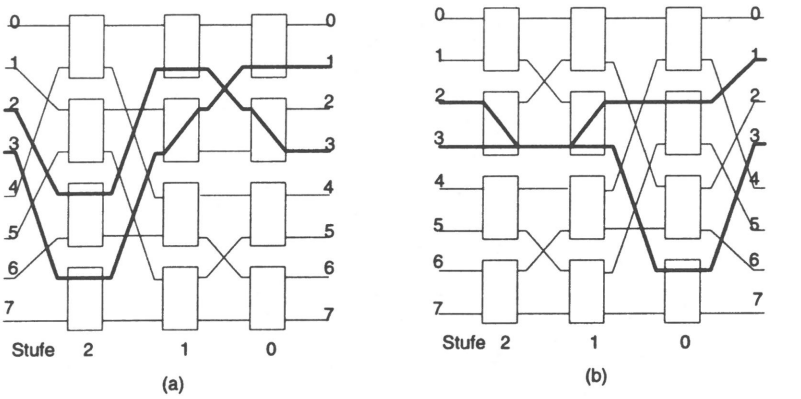


Abbildung 2.9: *Permutation in einem (a) GC-Netz und (b) Indirect-Binary-n-Cube Netz.*

Das **Baseline-Netz** [WuF80a] ist durch seinen rekursiven Aufbau aus kleineren Baseline-Netzen definiert, wie in Abbildung 2.10 gezeigt. Ein 2×2 Baseline-Netz ist ein einfaches 2×2 Koppelelement, während ein $N \times N$ Netz aus zwei $(N/2) \times (N/2)$ Netzen aufgebaut ist, mit einer zusätzlichen Netzstufe vor den beiden Netzen. In Abbildung 2.11 ist ein 8×8 Baseline-Netz gezeigt, das auch **Reverse-Exchange-Netz** genannt wird [WuF80b].

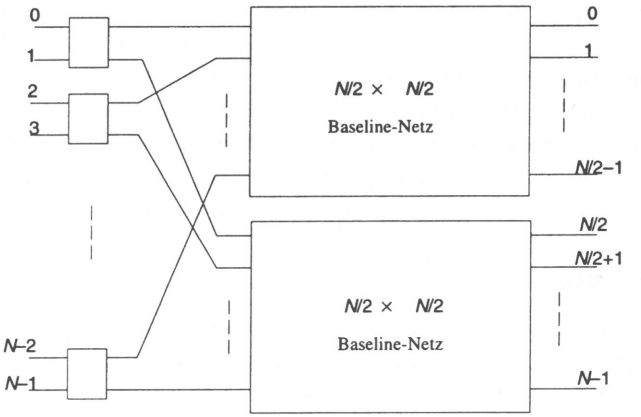


Abbildung 2.10: *Rekursive Konstruktion eines Baseline-Netzes.*

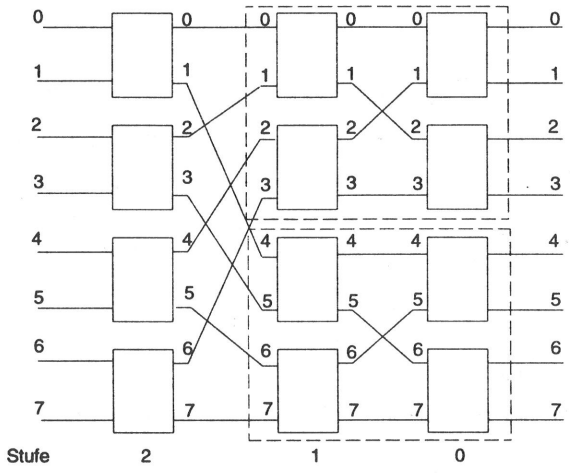


Abbildung 2.11: Baseline-Netz mit N=8.

2.3 Indirekte Mehrpfadnetze

In indirekten Banyan-Netzen existiert genau ein Weg zwischen jeder Quelle und jeder Senke. Dies vereinfacht die Wegesuche in solchen Netzen, kann aber zu Leistungseinschränkungen führen, da Wege sich Zwischenleitungen und Koppelelemente teilen müssen. So können innere Blockierungen im Netz entstehen (siehe Abschnitt 1.6.5). Auch können, wie in den vorherigen Abschnitten gezeigt, Einpfadnetze nicht jede beliebige Permutation durchführen. Hier schaffen Mehrpfadnetze Abhilfe. Repräsentative Vertreter dieser Netzklasse, die Clos und die Beneš -Netze, werden in den nächsten Abschnitten kurz beschrieben.

2.3.1 Clos-Netze

Von Clos wurde gezeigt [Clo53], daß ein dreistufiges Netz wie in Abbildung 2.12 gezeigt unter bestimmten Voraussetzungen blockierungsfrei bzw. umkonfigurierbar ist. In einem blockierungsfreien Netz können alle Datenpermutationen durchgeführt werden. In einem dreistufigen Clos-Netz ist in Stufe k die Anzahl der Koppelelemente durch den Parameter r_k, die Anzahl der Eingänge pro Koppelelement m_k und die der Ausgänge durch n_k gegeben. Aufgrund der Topologie gilt $m_2=r_1$, $m_3=r_2$, $n_1=r_2$ und $n_2=r_3$. Clos-Netze werden häufig für die ATM-Kommunikation eingesetzt [KaS88, LiL89, SaI90, SeT90].

Das Clos-Theorem legt die Bedingungen fest, unter denen ein dreistufiges Netz im strengen Sinne für Einfachverbindungen blockierungsfrei ist. Ein Clos-Netz ist dann und nur dann streng

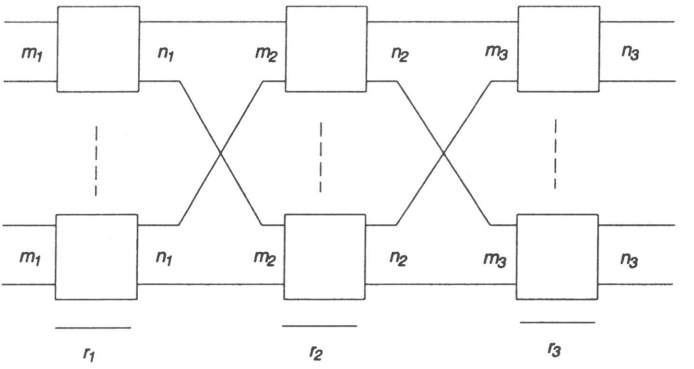


***Abbildung 2.12:** Struktur eines dreistufigen Clos-Netzes.*

nicht blockierend für 1-zu-1 Verbindungen, wenn $r_2 \geq m_1 + n_3 - 1$. Ein symmetrisches Netz mit $m_1 = n_3 = n$ ist demnach genau dann streng nicht-blockierend, wenn $r_2 \geq 2n - 1$.

Ein umkonfigurierbares Netz kann alle möglichen Permutationen durchführen, jedoch sind unter Umständen bestehende Verbindungen zu modifizieren. Daher hat ein solches Netz geringere Hardwareanforderungen als das nichtblockierende Netz. Ein dreistufiges Clos-Netz ist genau dann umkonfigurierbar, wenn $r_2 \geq \max(m_1, n_3)$ ist. Ein symmetrisches Netz mit $m_1 = n_3 = n$ ist demnach genau dann umkonfiguierbar, wenn $r_2 \geq n$ ist.

2.3.2 Beneš-Netze

Da ein Clos-Netz nur über drei Stufen verfügt, werden die individuellen Koppelelemente bei großen Netzen sehr komplex. Die Konstruktion solch großer Koppelelemente stellt eine wesentliche Schwierigkeit dar, so daß eine Verringerung der Komplexität der Koppelelemente wünschenswert ist, die durch die rekursive Konstruktion erreicht werden kann. Das Beneš-Netz ist ein Beispiel für ein Netz, das aus 2×2 Koppelelementen aufgebaut ist.

Grundlage des Konstruktionsprinzips eines **Beneš-Netzes** [Ben65] ist ein dreistufiges symmetrisches Clos-Netz mit N Ein- und Ausgängen, wie in Abbildung 2.13 für N=8 gezeigt. Eingangs- und Ausgangsstufen sind aus 2×2 Koppelelementen aufgebaut (n=m=2), woraus unmittelbar folgt, daß die mittlere Stufe aus zwei $N/2 \times N/2$ Koppelelementen bestehen muß. Um ein insgesamt umkonfigurierbares Netz zu erhalten, ist es ausreichend, daß die Koppelelemente der mittleren Stufe lediglich umkonfigurierbar sind; die nichtblockierende Eigenschaft eines Crossbars ist nicht erforderlich. Daher kann jedes der Koppelelemente der mittleren Stufe wiederum als ein dreistufiges Clos-Netz mit n=m=r=2 aufgebaut werden, wie in Abbildung 2.14 gezeigt. Dieses rekursive Prinzip kann auf beliebig große Netze angewendet werden.

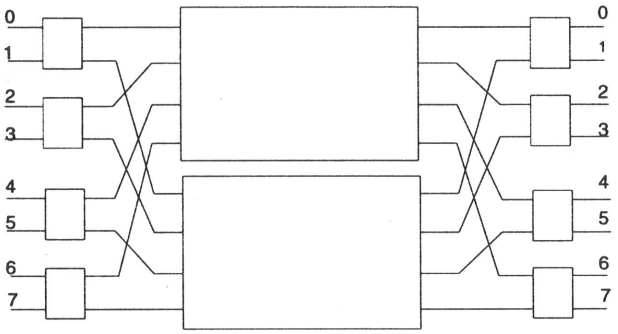


Abbildung 2.13: *Dreistufiges Clos-Netz mit 2 × 2 Koppelelementen in erster und dritte Stufe.*

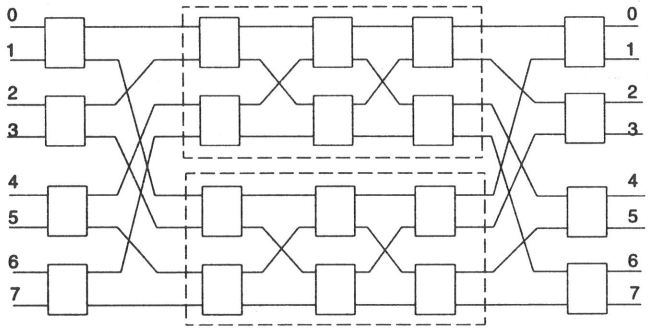


Abbildung 2.14: *Beneš-Netz mit N=8, aufgebaut aus 2 × 2 Koppelelementen.*

3 Koppelelementarchitekturen und VLSI-Realisierungsaspekte

In paketvermittelnden Netzen müssen die Koppelelemente neben der Vermittlungsfunktion bei Blockierungen an Ausgängen Pakete zwischenpuffern. Die Organisation von Puffern ist dabei für die Leistungsfähigkeit der Elemente ausschlaggebend. Die wichtigsten grundlegenden Organisationen sind Eingangspuffer, Ausgangspuffer, Zentralpuffer und verteilte Pufferung. Diese Architekturen werden in den folgenden Abschnitten aufgeführt. Varianten dieser Prinzipien können genutzt werden, um eine weitere Steigerung der Leistungsfähigkeit zu erzielen.

3.1 Eingangspuffer

Vom Standpunkt der schaltungstechnischen Realisierung bieten Koppelelemente mit Eingangspuffer wesentliche Vorteile. Abbildung 3.1 zeigt die Organisation eines $B \times B$ eingangsgepufferten Koppelelements. Jeder Eingang ist mit einem Puffer verbunden, der einen $B \times B$ Ausgangsschalter speist.

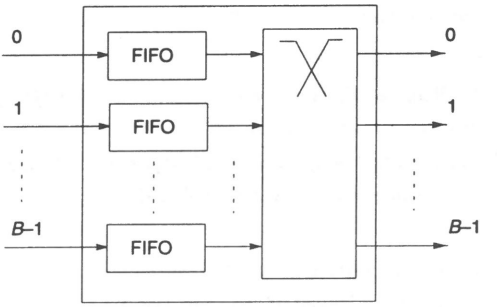


Abbildung 3.1: *$B \times B$ Koppelelement mit Eingangspufferarchitektur.*

Im einfachsten Fall folgen die Puffer der **"First In First Out"-Disziplin (FIFO)** (d. h. Pakete verlassen die Puffer in der Reihenfolge ihrer Ankunft) und nur ein Paket wird zu einer Zeit aus dem Puffer ausgelesen. Vom schaltungstechnischen Standpunkt her ist diese Struktur besonders einfach zu realisieren, da sowohl die Puffer als auch die Schaltmatrix mit der Datenübertragungsrate operieren.

Enthalten zwei oder mehr Ausgänge der Puffer Pakete, die zum selben Koppelelementausgang gerichtet sind, so wird nur eines dieser Pakete durchgeschaltet, während die anderen Pakete in ihrem jeweiligen Puffer warten, bis der Ausgang frei wird. Die Auswahl des zu lesenden Puffers

wird durch einen Arbitrierungsalgorithmus bestimmt. Diese Arbitrierungsmechanismen lassen sich in zwei Klassen einteilen: die **zustandsunabhängigen** und die **zustandsabhängigen Arbitrierungsstrategien**.

Zustandsunabhängige Arbitrierungsstrategien

Bei der **zufälligen Auswahl** [Ili91, KaH87, RaT89] wird ein Paket aus einem zufällig ausgewählten Puffer herausgenommen. Dies führt zu einer fairen Auswahl, ist jedoch schwierig zu implementieren.

Bei der **festen Priorisierung** [BaM91, KaH87] werden die Puffer in einer festen Reihenfolge ausgewählt. Dieser Mechanismus ist einfach zu implementieren, führt jedoch zu einer unfairen Behandlung der Pakete.

Bei der **zyklischen Priorisierung** [Dai87, Hah91] werden die Puffer ebenfalls sequentiell ausgewählt. Allerdings wird der Startpunkt bei jedem Zyklus um einen Puffer verschoben. Dies führt zu einer fairen Auswahl der Pakete und ist einfach zu implementieren.

Zustandsabhängige Arbitrierungsstrategien

Bei der **wartezeitabhängigen Auswahl** [Ili91, RaT89] wird der Puffer mit dem ältesten Paket ausgewählt. Hierbei können die Wartezeitschwankungen minimiert werden [The94]. Jedoch ist ein erhöhter Hardware-Aufwand erforderlich, da die Wartezeiten einzelner Pakete bestimmt werden müssen.

Die **Auswahl des längsten Puffers (Longest Queue First; LQF)** [GaG91, KaH87, RaT89] führt zu gleichmäßigen mittleren Längen der einzelnen Puffer. Die Verlustwahrscheinlichkeit wird so minimiert, allerdings steigt die Schwankung der Durchlaufzeit [The94]. Diese Strategie führt nur zu einem geringen Hardware-Mehraufwand.

Ein Nachteil der Eingangspufferstruktur in Koppelelementen ist die geringe Leistungsfähigkeit dieser Organisation, die im **"Head-of-the-line" (HOL)**-Flaschenhals begründet ist. Hierbei können wartende Pakete an den Ausgängen der Puffer Pakete, die hinter ihnen gepuffert sind, blockieren. Dieser Mechanismus ist in Abschnitt 6.3.1 näher beschrieben und führt zu einer Reduzierung des maximalen Durchsatzes auf rund 58% bei einem 16×16 Koppelelement [KaH87].

Ein zweites Problem ist in Abbildung 3.2 anhand eines 2×2 Koppelelements illustriert. Nur einer der Puffer enthält Pakete, von denen je eines zu Ausgang A bzw. B gerichtet ist. Aufgrund der sequentiellen Abarbeitung eines FIFO-Puffers kann in dieser Situation pro Zyklus nur ein Paket ausgelesen werden, obwohl Nachrichten für beide Ausgänge verfügbar sind. Auch durch diese Situation wird der Durchsatz solcher Koppelelemente fundamental begrenzt.

Zwei Modifikationen dienen dazu, die beiden Probleme zu reduzieren. So kann durch Abkehr vom FIFO-Prinzip, also durch wahlfreien Zugriff zu beliebigen Elementen eines Puffers, die

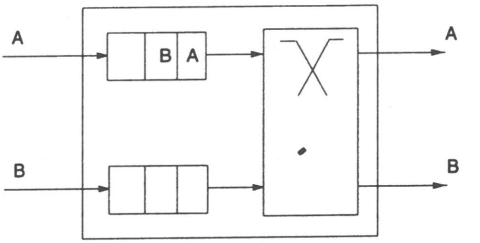


***Abbildung 3.2:** Durchsatzbegrenzung durch sequentiellen Pufferzugriff.*

HOL-Blockierung umgangen werden [ArG89, BuT89, HlK88, ShM90]. Dies ist für ein 2×2-Koppelelement in Abbildung 3.3a dargestellt. Durch gleichzeitiges Auslesen mehrerer Nachrichten aus einem Puffer ist eine wesentliche Leistungssteigerung möglich (siehe Abbildung 3.3b) [TaF88]. In [RoS88] wurde gezeigt, daß das gleichzeitige Lesen von zwei Pufferplätzen auch bei größeren Koppelelementen ausreicht. Können mehr als zwei Nachrichten gelesen werden, so resultiert nur eine unwesentliche Leistungssteigerung. Höchste Anforderungen werden durch eine Kombination beider Methoden in Verbindung mit einer effektiven Auswahlstrategie des nächsten zu lesenden Puffers erfüllt [SiM90]. Diese Struktur wird mit **verallgemeinerten Eingangspuffern** beschrieben und ist in Abbildung 3.4 dargestellt.

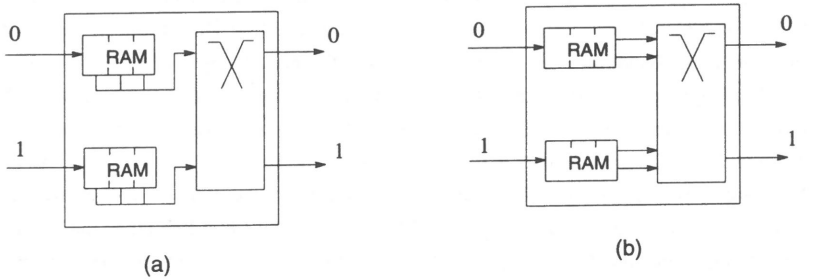


Abbildung 3.3: 2×2-Koppelelement mit (a) wahlfreiem Zugriff auf die Puffer; (b) gleichzeitigem Auslesen zweier Elemente aus jedem Puffer

Wie in Abschnitt 1.6.5 beschrieben, dürfen Pakete in Block-Netzen von einem Quellkoppelelement in einer Netzstufe zu einem Zielkoppelelement in der nächsten Stufe nur dann weitergeleitet werden, wenn in dem Zielpuffer des Zielelements Platz für das Paket vorhanden ist. Hierzu

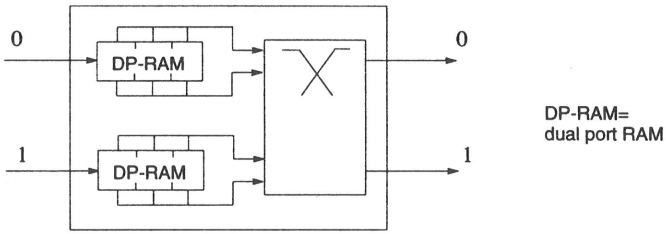


Abbildung 3.4: 2×2-Koppelelement mit verallgemeinerten Eingangspuffern.

existieren Steuerleitungen an jedem Koppelelementport, die den Zustand der Elementpuffer den angeschlossenen Koppelelementen mitteilen. Bei eingangsgepufferten Koppelelementen reicht hierfür eine Steuerleitung pro Elementport aus. Diese Leitung signalisiert dem Quellelement, ob der Eingangspuffer des Eingangsports am Zielelement, an dem das Quellelement angeschlossen ist, noch Pufferplätze frei hat. Solange Pufferplätze vorhanden sind, kann das Quellelement Pakete senden.

3.2 Ausgangspuffer

Werden die Puffer nicht den Eingängen, sondern den Ausgängen zugeordnet, wie in Abbildung 3.5 gezeigt, so wird eine höhere Leistungsfähigkeit erreicht. Ankommende Pakete werden vom Verteiler zum korrekten Ausgangspuffer geleitet und dort gepuffert. Dies stellt besondere Anforderungen an die Bandbreite des Verteilers sowie der Puffers. Im ungünstigsten Fall treffen an allen Eingängen gleichzeitig Nachrichten ein, die zum gleichen Ausgang gerichtet sind. Dann müssen bei einem $B \times B$-Koppelelement B Nachrichten durch den Verteiler vermittelt werden und in jedem der Ausgangspuffer B Pakete geschrieben und eines gelesen werden. Der interne Pufferzugang muß also stark parallelisiert werden, um diesen Anforderungen gerecht zu werden. Diese Parallelisierung kann z. B. unter Zuhilfenahme des **Bit-Slice-Prinzips** [BaG91, Ito91, SuN89] oder durch die Aufteilung der Puffer in mehrere kleinere Puffer, die zyklisch gelesen und beschrieben werden [BaF91], verwirklicht werden.

Da in einem Ausgangspuffer immer nur Pakete gepuffert werden, die zum gleichen Ausgang geleitet werden, kann eine HOL-Blockierung, wie sie im vorangegangenen Abschnitt diskutiert wurde, nicht auftreten (siehe Abschnitt 6.3.1). So verhält sich jeder Ausgangspuffer wie ein idealer, statischer und von den anderen Puffern unabhängiger Multiplexer. Bei einer unbegrenzten Puffergröße kann deshalb ein maximaler Durchsatz von 100% erreicht werden. Werden Puffer mit der FIFO-Strategie verwendet, erhält man die kleinstmöglichen Durchlaufzeitschwankungen der Pakete.

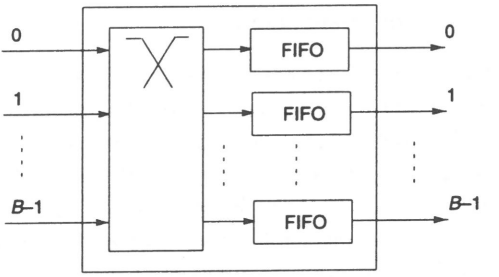


Abbildung 3.5: *$B \times B$ Koppelelement mit Ausgangspufferarchitektur.*

3.2.1 Interne Paketvermittlung

Die Vermittlung der Datenpakete zwischen den Ein- und Ausgängen eines ausgangsgepufferten Koppelelements kann unterschiedlich realisiert werden [AhD89, GaR90, PrS87, SuN89, YeH87]. Aus Geschwindigkeitsgründen wird hierbei keine zentrale, sondern eine verteilte Steuerung, die die Zieladressen der Pakete auswertet, verwendet. Häufig wird eine interne Busstruktur benutzt, um die ankommenden Pakete auf die Ausgänge zu verteilen. Diese Koppelelementstruktur ist in Abbildung 3.6 gezeigt. Nach einer Seriell/Parallel-Wandlung der empfangenen Daten werden diese auf ein Zeitmultiplex-Bussystem gegeben. Ein Adreßfilter an jedem Ausgangspuffer selektiert die Daten, so daß die Pakete in den richtigen Puffern gepuffert werden können. Hierbei muß die Buskapazität mindestens so groß sein wie die Bitraten aller Eingangsleitungen, da anderenfalls nicht immer alle Pakete vermittelt werden können und Verluste entstehen.

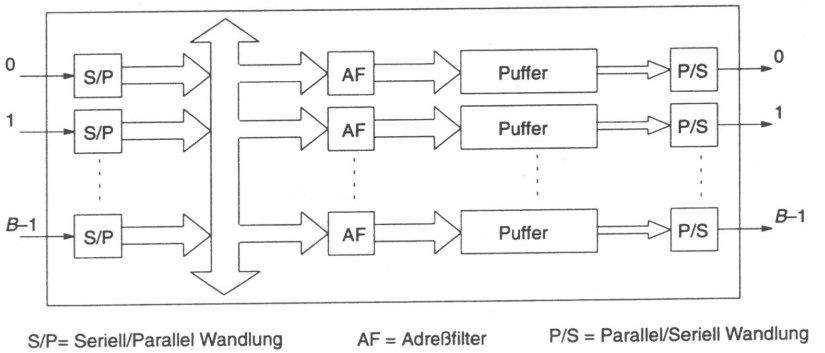


Abbildung 3.6: *Struktur eines $B \times B$ ausgangsgepufferten Koppelelements auf der Basis eines Zeitmultiplex-Bussystems.*

3.2.2 Externe Paketvermittlung

Die Paketflußsteuerung zwischen Koppelelementen in Block-Netzen mit ausgangsgepufferten Schaltelementen gestaltet sich schwieriger als bei eingangsgepufferten Schaltern. Eine Möglichkeit ist, B Steuerleitungen pro Koppelelementport zu verwenden, die dem angeschlossenen Koppelelement signalisieren, welche der B Ausgangspuffer noch Pakete aufnehmen kann. Da es jedoch vorkommen kann, daß bis zu B Pakete an den B Eingangsports eines Koppelelements gleichzeitig zu einem Elementausgangsport geroutet werden müssen, muß die Flußsteuerung Pakete bereits abweisen, wenn ein Puffer noch B–1 freie Pufferplätze besitzt, damit zu keinem Zeitpunkt Pakete verloren gehen können. Diese Flußkontrolle resultiert also einerseits in einem großen Systemhardware-Aufwand (insbesondere für größere Koppelelemente, da mehrfache Steuerleitungen pro Elementport verwendet werden müssen) und andererseits in einer suboptimalen Ausnutzung der Ausgangspuffer, da es vorkommen kann, daß Pakete abgewiesen werden, obwohl Pufferplatz vorhanden gewesen wäre.

Eine effektivere Flußsteuerung kann implementiert werden, wenn pro Koppelelementeingangsport ein zusätzliches Eingangslatch benutzt wird, wie in Abbildung 3.7 gezeigt [PfN85]. Bei einem Pakettransfer zwischen zwei Koppelelementen wird das Paket zuerst in dem Eingangslatch zwischengepuffert und dann in einem zweiten Schritt durch das eigentliche Koppelelement geroutet. Ist der Zielausgangspuffer voll, wartet das Paket im Eingangslatch. So muß einem angeschlossenen Koppelelement nur mitgeteilt werden, ob das Latch des Zieleingangsports voll ist, so daß nur eine Steuerleitung pro Elementport benötigt wird. Da in jedem Zyklus ein Element selbst erkennen kann, wieviele Pakete zu einem bestimmten Ausgangspuffer geroutet werden müssen, können durch einen Arbitrierungsmechanismus, z. B durch eine zyklische Auswahl der Latches (siehe Abschnitt 3.1), die Pakete auf die einzelnen Ausgangspuffer verteilt werden. So kann jeder Ausgangspuffer vollständig ausgenutzt werden. Dies führt zu einer höheren Leistung unter gleichverteiltem Verkehr [PfN85]. In Abschnitt 6.3 wird jedoch gezeigt, daß diese Eingangslatchkonfiguration zu einer Netzleistungsminderung unter ungleichverteilten Verkehren führen kann.

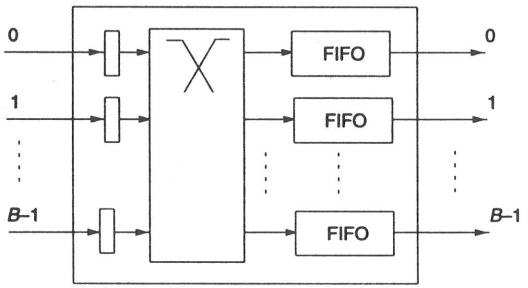


Abbildung 3.7: *$B \times B$ Koppelelement mit Ausgangspufferarchitektur und Eingangslatches.*

3.3 Verteilte Pufferung

Die Probleme der hohen Bandbreiteanforderungen an die Puffer und des HOL-Blockierens können mittels verteilter Pufferanordnungen vermieden werden. So ist in Abbildung 3.8 eine Crossbar-Anordnung gezeigt, bei der jedem Koppelpunkt ein eigener Puffer zugeordnet ist. Das wesentliche Problem solcher verteilten Strukturen ist der sehr hohe Pufferbedarf.

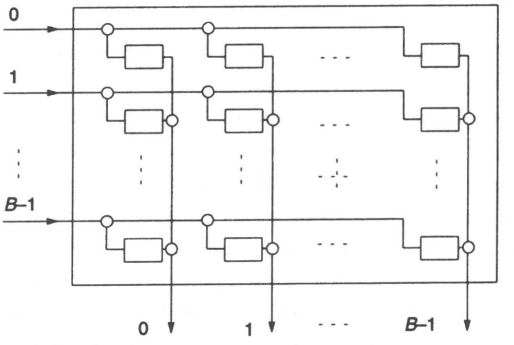


Abbildung 3.8: *$B \times B$ Koppelelement mit verteilter Pufferarchitektur.*

3.4 Zentralpufferung

Um bei der Ausgangspufferorganisation eine hohe Leistungsfähigkeit zu erreichen, ist ein erheblicher Pufferbedarf erforderlich. Dieser Nachteil kann durch eine gemeinsame Nutzung der Pufferplätze vermieden werden. Die Konfiguration der resultierenden Zentralpufferorganisation ist in Abbildung 3.9 dargestellt.

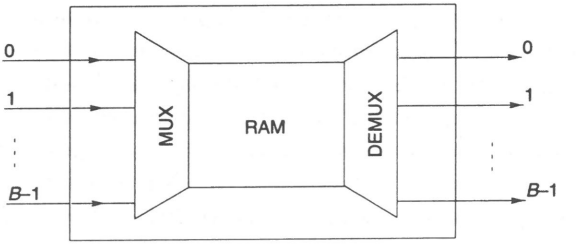


Abbildung 3.9: *$B \times B$ Koppelelement mit Zentralpuffer.*

Pakete an den Eingängen werden über einen Multiplexer in einen Puffer geleitet, und aus dem Puffer durch einen Demultiplexer zum korrekten Ausgang geschickt. Der Zentralpuffer kann bis zu Z Pakete puffern. Intern ist jedem Ausgang ein logischer Puffer zugeordnet, aus dem die Zellen in der Reihenfolge ihres Eintreffens entnommen werden. Die Anforderungen an die Pufferbandbreite sind noch größer als bei der Ausgangspufferorganisation, da im ungünstigsten Fall Pakete von allen Eingängen in den Puffer geschrieben und Pakete für alle Ausgänge gelesen werden müssen. Aufgrund dieser Anforderungen ist ein Aufbau aus diskreten Komponenten schwierig; bei VLSI-Schaltungen ist eine Realisierung einfacher möglich (siehe nächsten Abschnitt).

Beim **Complete Sharing** der Pufferplätze [KaK80] wird der Puffer gemeinsam von allen Ausgängen benutzt, wobei sich die einzelnen Puffer gegenseitig mit Pufferplätzen aushelfen können. Der Pufferplatz kann so sehr effizient ausgenutzt werden, so daß wesentlich weniger Gesamtpufferplatz benötigt wird, als bei physikalisch getrennten Ausgangspuffern. In [BoB87] wurde gezeigt, daß ausgehend von einer Paketverlustwahrscheinlichkeit von 10^{-12} der Pufferplatz in einem 16×16 Koppelelement um den Faktor 7.5 geringer ist als bei getrennten Ausgangspuffern. Diese Pufferplatzersparnis ist besonders bedeutsam, wenn eine Realisierung des Koppelelements in Form eines einzelnen Schaltkreises angestrebt wird. Da in diesem Fall ein großer Teil der Chipfläche vom Puffer eingenommen wird, wird ein möglichst geringer Pufferbedarf angestrebt. Deshalb basieren viele Ein-Chip Lösungen von Koppelelementen auf dem Zentralpufferprinzip [FiF91, HeS90].

Das Prinzip des Complete Sharing wirkt sich jedoch nachteilig aus, wenn ein oder mehrere Ausgänge des Koppelelements häufiger angesprochen werden als andere. In diesem Fall kann sich der Zentralpuffer mit Paketen füllen, die für die häufiger angesprochenen Ausgänge bestimmt sind, so daß für die restlichen Pakete kein Platz mehr vorhanden ist. Um diesen Effekt zu vermeiden, muß die Belegung des Puffers durch einzelne Ausgänge vermieden werden. Dies kann z. B. durch die Begrenzung der maximalen Länge der einzelnen Ausgangspuffer geschehen (**address queue length limitation**, in der Vermittlungstechnik sind diese Verfahren unter den Begriffen "partial buffer sharing" oder "class limitation" bekannt) [ChH91, LeK90, ScS90b]. Hierbei kann ein einzelner Ausgang nur einen Teil des Zentralpuffers belegen, so daß bei einer Verkehrsunsymmetrie zu diesem Ausgang noch genügend Pufferplatz für die restlichen Ausgänge zur Verfügung steht. Wird angenommen, daß Z (Zentralpuffergröße) ein Vielfaches von B (Koppelelementgröße) ist, so führt die Begrenzung der Pufferlängen auf jeweils Z/B Plätze zu einem ausgangsgepufferten Koppelelement. Falls in Zentralpufferelementen mit Pufferlängenbegrenzung jedoch Verkehrsunsymmetrien bestehen, die mehrere Ausgänge stärker belasten, so kann sich der Zentralpuffer dennoch füllen und den restlichen Verkehr behindern. Dies kann vermieden werden, indem für jeden Ausgangspuffer ein kleiner Teil des Zentralpuffers reserviert wird, auf den kein anderer Puffer zugreifen kann [ChH91, KaK80, Tze91b]. Bei einer Verkehrsunsymmetrie steht dann jedem Ausgangspuffer ein Mindestpuffer zur Verfügung, so daß Paketverluste verringert werden können.

3.5 Prioritätsmechanismen in ATM-Koppelelementen

Es sind verschiedene Mechanismen vorgeschlagen worden, um gleichzeitige ATM-Verbindungen mit zwei unterschiedlichen Zellverlustraten zu erhalten. Hierbei werden die ATM-Verkehrsströme in zwei Prioritätsklassen eingeteilt. Zwei Prioritätsmethoden, die sich durch eine entsprechende Pufferverwaltung in den einzelnen Koppelelementen auszeichnen, werden in den folgenden Abschnitten näher beschrieben.

3.5.1 Verdrängungssystem

Bei dieser Methode verdrängt eine eintreffende höher priorisierte Zelle in einem Koppelelementpuffer eine gepufferte Zelle mit niedrigerer Priorität, wobei die verdrängte Zelle verloren geht. Meist wird hierbei die zuletzt gepufferte (**Last In First Drop, LIFD**) oder die zuerst gepufferte Zelle niedrigerer Priorität (**First In First Drop, FIFD**) verworfen. So kann eine Zelle höherer Priorität nur dann verloren gehen, wenn der Puffer nur mit Zellen dieser Prioritätsklasse vollständig gefüllt ist. Damit die Zellreihenfolge erhalten bleibt, müssen nach dem Verdrängen einer Zelle aus einem FIFO-Puffer die hinter dieser Zelle gepufferten Zellen um einen Platz im Puffer vorrücken. Die eingetroffene Zelle höherer Priorität wird dann hinter allen anderen Zellen im Puffer gepuffert.

Der Nachteil dieser Methode ist der relativ hohe Hardware-Mehraufwand. Es müssen FIFO-Puffer durchsucht werden, Zellen entnommen und andere Zellen verschoben werden. Diese Puffersteuerung kann z. B. durch die Benutzung von mehreren verketteten Listen erfolgen (dann ist *keine* Zellverschiebung nötig).

3.5.2 Schwellwertverfahren

Bei diesem Verfahren wird ein Koppelelementpuffer in zwei Bereiche eingeteilt. In jedem Puffer der Länge D dürfen nur solange Zellen einer Prioritätsklasse i gepuffert werden, bis ein Grenzwert D_i überschritten wird. Dieser Grenzwert kann sich entweder auf die Gesamtzahl der im Puffer abgelegten Zellen, oder auf die Gesamtzahl der Zellen der Prioritätsklasse i im Puffer beziehen. Hierbei werden keine gepufferten Zellen verdrängt.

Beim **Partial Buffer Sharing**, einem Schwellwertverfahren, welches in [KrH91] eingeführt worden ist, ist der Grenzwert für die höher priorisierten Zellen D_1=D. Diese Zellklasse darf also den gesamten Puffer benutzen. Zellen mit niedrigerer Priorität dürfen jedoch nur einen Bereich von D_2 Pufferplätzen benutzen. Dies ist in Abbildung 3.10 an einem 2×2 Koppelelement dargestellt. Ein Teilbereich von D–D_2 Pufferplätzen ist demnach ausschließlich für Zellen mit höherer Priorität reserviert, während der Bereich von D_2 Pufferplätzen von beiden Klassen genutzt werden kann.

Im Vergleich zum Verdrängungssystem erfordert das Schwellwertverfahren erheblich weniger Hardware-Aufwand, da pro Prioritätsklasse zusätzlich nur ein Register für den Grenzwert, ein

Zähler für den aktuellen Füllstand und Vergleichslogik benötigt wird. Da die Leistungscharakteristiken des Schwellwertverfahrens vergleichbar mit denen eines Verdrängungssystems sind, wird es in vielen Implementierungen von Verlustprioritäten in ATM-Netzen verwendet [BaF91, KoE91]. In Abschnitt 6.4.3 wird die Leistungsfähigkeit von partial buffer sharing Koppelelementen unter MPEG-codierten Videoströmen mit der des in dieser Arbeit entwickelten Prioritätskoppelelements verglichen.

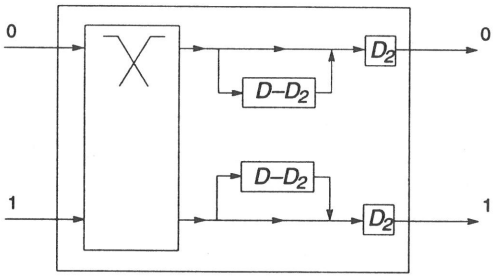


Abbildung 3.10: Schwellwertverfahren in einem 2 × 2 Koppelelement.

3.6 VLSI-Realisierungsaspekte

Wie im vorherigen Abschnitt dargestellt, bieten Koppelelemente mit Zentralpuffer die Vorteile hoher Leistungsfähigkeit bei reduzierten Anforderungen an die Puffergröße. Da der Flächenbedarf der Puffer direkt in die Kosten eines entsprechenden VLSI-Chips eingeht, sind solche Architekturen besonders attraktiv, wenn die Probleme der hohen Pufferbandbreite gelöst werden können. In diesem Abschnitt sollen zwei Architekturen vorgestellt werden (die Zentalpufferorganisation mit paralleler Pufferstruktur und mit Schieberegisterstruktur), mit denen dies erreicht wird.

3.6.1 Zentralpufferorganisation mit paralleler Pufferstruktur

Diese Pufferstruktur ist in Abbildung 3.11 anhand eines 2 × 2 Koppelelements zu sehen und besteht aus jeweils einem Schieberegister an jedem Elementport und einem RAM. Die an einem Eingangsport seriell ankommenden Bits eines Datenpakets werden in das Schieberegister des Ports eingeschoben (Abbildung 3.12a). Ist das gesamte Paket eingetroffen und im Schieberegister gepuffert, wird es in einem Schritt in das RAM eingeschrieben (Abbildung 3.12b), wobei die RAM-Zelle durch die extrahierte Paketadresse ausgewählt wird. Beim Auslesen des Pakets wird es in einem Schritt von der RAM-Zeile in das Ausgangsschieberegister eingelesen und dann

seriell auf den Ausgangsport geschoben (Abbildung 3.12c). Die minimale Verweilzeit eines Pakets in einem Koppelelement dauert also drei Zelltakte. Das Element wird durch einen zentralen Controller im Kontrollpfad gesteuert. Die globalen Signalleitungen im RAM, verbunden mit der Auswahl der zu bearbeitenden RAM-Zeile, sind in dieser Struktur allerdings die leistungsbegrenzenden Komponenten.

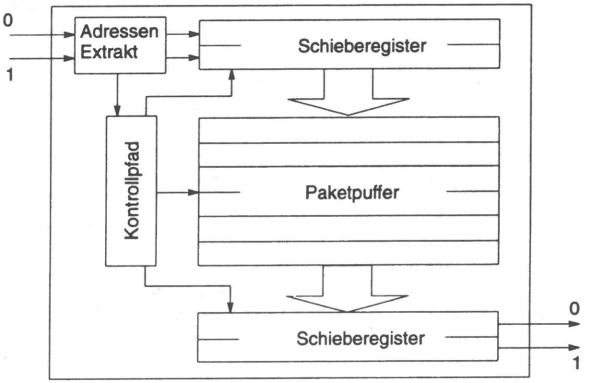


Abbildung 3.11: Zentralpufferorganisation mit paralleler Pufferstruktur eines 2×2 Koppelelements.

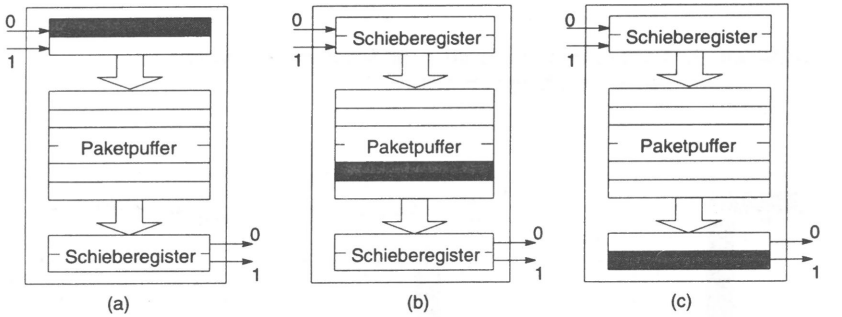


Abbildung 3.12: 2×2 Koppelelement mit paralleler Pufferstruktur. (a) Eingangsoperation; (b) Pufferung; (c) Ausgabeoperation.

3.6.2 Zentralpufferorganisation mit Schieberegisterstruktur

Bei dieser Organisation besteht, wie in Abbildung 3.13 dargestellt, der Paketpuffer aus einzelnen Schieberegistern. Ein an einem Eingangsport seriell eintreffendes Paket wird über einen Ein-

gangsschalter in eins der Schieberegister geschoben (Abbildung 3.14a). Hierbei wird die Stellung des Eingangsschalters durch den Inhalt des Eingangsselektionsregisters gesteuert. Ist das letzte Bit des Pakets eingeschoben worden, so befindet sich das gesamte Paket bereits im Puffer (Abbildung 3.14b). Beim Auslesen wird das Paket aus dem Schieberegister über den Ausgangsschalter auf den richtigen Ausgangsport gegeben (Abbildung 3.14c), wobei die Stellung des Ausgangsschalters durch den Inhalt des Ausgangsselektionsregisters gesteuert wird.

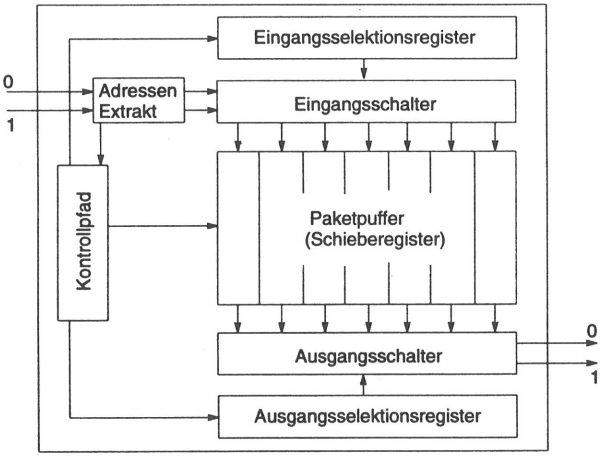


Abbildung 3.13: Zentralpufferorganisation mit Schieberegisterstruktur eines 2×2 Koppelelements.

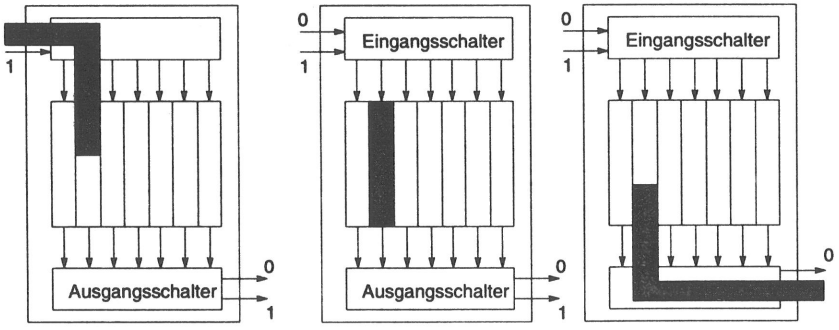


Abbildung 3.14: 2×2 Koppelelement mit Schieberegisterstruktur. (a) Eingangsoperation; (b) Pufferung; (c) Ausgabeoperation.

Diese Struktur führt zu einer schnellen Pufferung der Pakete, da nur Schieberegister verwendet werden. Eine effiziente VLSI-Struktur einer Schieberegisterzelle ist in Abbildung 3.15 gezeigt. Sie kann aus minimal 7 Transistoren aufgebaut werden. Durch die lokalen Verbindungen zwischen den Schieberegisterzellen ist es möglich, eine Schieberate von 5ns (200 Mbit/s) zu erreichen [BeS91].

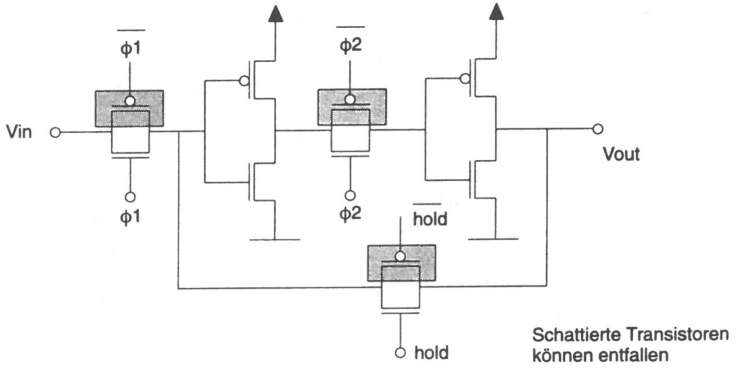


Abbildung 3.15: *Dynamische Schieberegisterzelle [BeS91].*

Um die Schnelligkeit der Schieberegister auszunutzen, muß eine effiziente Steuerung und Auswahl der freien Pufferplätze gewährleistet sein. Eine Möglichkeit des Kontrollpfades ist in Abbildung 3.16 gezeigt. Die Schaltersteuerung muß empfangenen Paketen leere Puffer zuweisen, die Paket- und Pufferadressen puffern, und den Abgang von Paketen koordinieren. Hierzu wird für jeden Schalterausgang eine FIFO-Adreßqueue (OQB) benutzt, in dem die Adressen der Schieberegisterzellen gepuffert werden, die Pakete für diesen Ausgang gepuffert haben. Ebenso existiert ein FIFO-Adreßpuffer (EQB) für die Schieberegister, die leer sind (siehe Abbildung 3.16). Zu Anfang ist der Zentralpuffer leer; die EQB ist gefüllt mit allen Schieberegisteradressen. Wird nun an einem Eingangsport des Koppelelements ein Paket empfangen, so wird eine Adresse einer leeren Schieberegisterzelle aus dem EQB genommen und in der OQB gepuffert, für dessen Ausgangsport das Paket bestimmt ist. Gleichzeitig wird das Paket in das ausgewählte Schieberegister eingeschoben. Soll ein Paket auf einem Ausgangsport des Koppelelements ausgegeben werden, so wird die Schieberegisteradresse aus der zu dem Ausgangsport gehörenden OQB ausgelesen und in den EQB eingeschrieben. Gleichzeitig wird über diese Adresse das dazugehörige Schieberegister ausgewählt und das gepufferte Paket aus dem Koppelelement herausgeschoben. Um eine hohe Verarbeitungsgeschwindigkeit zu erzielen, existieren Register für den nächsten Eingang und den nächsten Ausgang, die die Schieberegisteradressen für den nächsten Paketzyklus puffern.

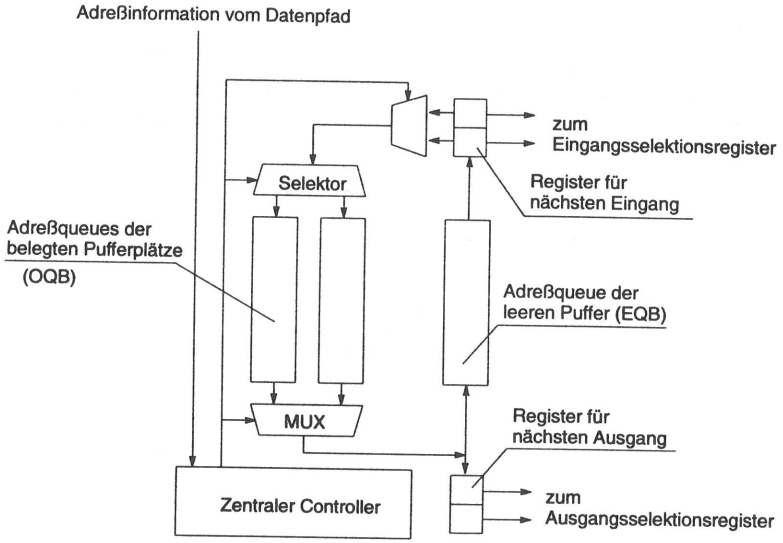


Abbildung 3.16: Kontrollpfad eines 2×2 Zentralpufferkoppelelements mit Schieberegister-Organisation.

Es existieren verschiedene Möglichkeiten, FIFO-Adreßqueues zu implementieren. In [Sch92] wurden mehrere Varianten untersucht und das RAM mit Zeigeradressierung als effiziente Struktur für die Implementierung der Adreßqueues eines Zentralpufferkoppelelements ausgewählt. Das Blockschaltbild solch einer Adreßqueue, welche n Adressen puffern kann, ist in Abbildung 3.17 dargestellt. Es wird eine $n \times m$ Puffermatrix verwendet, so daß n Zentralpufferadressen, die jeweils m bit lang sind, gepuffert werden können. Zur Steuerung der Puffermatrix werden keine Adreßdekoder benutzt. Der zu schreibende oder lesende Pufferplatz wird durch einen n-bit Zeiger direkt angesteuert. Da pro Pufferzugriff jeweils genau ein Pufferwort im RAM angesprochen wird, bestehen die Binärwerte der Schreib- bzw. Lesezeiger aus einer Eins und n-1 Nullen (durch die Eins wird die entsprechende Pufferzeile im RAM ausgewählt). Zur Erzeugung der Schreib- und Lesezeiger werden zwei jeweils n-bit lange und 1-bit breite Schieberegisterringe verwendet, in denen bei jedem Schreib- bzw. Lesetakt eine "1" um eine Position im Ring weitergeschoben wird. Dies entspricht dem Prinzip einer verketteten Liste. Wird bei einer Leseoperation nur das Leseschieberegister getaktet, und bei einer Schreiboperation nur das Schreibschieberegister, so erhält die Adreßqueue FIFO-Charakter. Befinden sich die beiden Einsen in den Ringen nach einer Schreiboperation in der gleichen Position, so ist der Puffer voll. Geschieht dies nach einer Leseoperation, zeigt dies einen leeren Puffer an.

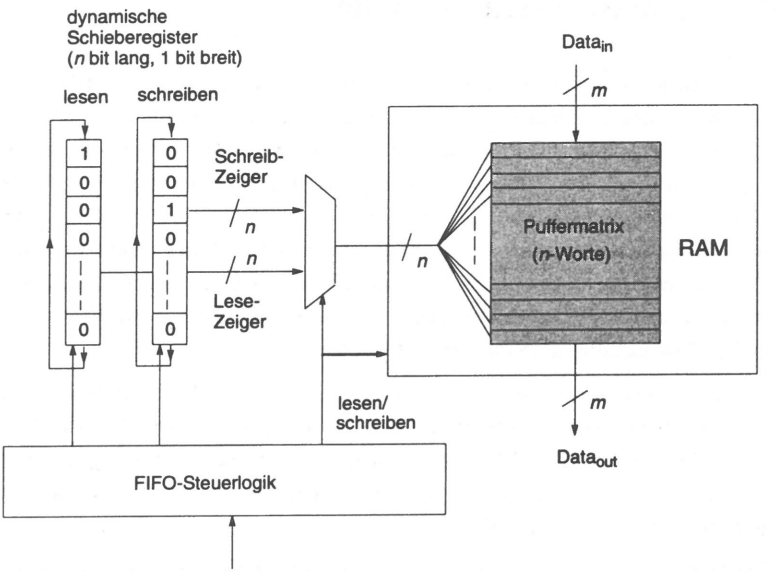


Abbildung 3.17: *Blockschaltbild einer FIFO-Queue mit n Worten, aufgebaut aus einem RAM mit Lese- und Schreibzeigeradressierung.*

In [BeS91] wird die Implementierung eines Ein-Chip 4×4 Zentralpufferkoppelelements mit Schieberegister-Organisation beschrieben. Die Schaltung wurde auf dem IMS Gate-Forest [BeC89, BeH88, BeK88, Hoe90], einer für gemischt statisch/dynamischen CMOS-Logik optimierten 1.2µm CMOS Semicustom Sea-of-Gates Technologie [KeS90], realisiert. Der Zentralpuffer kann 32 10-Byte Pakete bei einer 25 MHz Byte-Taktrate puffern. Die Schaltung benötigt etwa 56000 Transistoren auf dem 85 mm^2 großen Chip. Der Datenpfad mit der Eingangs- und Ausgangslogik und dem Zentralpuffer belegt etwa 2/3 des Chips, während die EQB und OQBs den meisten Platz im Kontrollpfad beanspruchen (etwa 75%). Dieses Koppelelement kann eine Verkehrsrate von bis zu 70% bewältigen.

Durch eine Begrenzung der Längen der OQBs kann ein Zentralpuffer mit address queue length limitation (siehe Abschnitt 3.4) realisiert werden. Wird angenommen, daß Z (Zentralpuffergröße) ein Vielfaches von B (Schaltergröße) ist, so kann durch eine Begrenzung der OQB-Längen auf jeweils Z/B Pufferplätze ein ausgangsgepuffertes Koppelelement realisiert werden.

4 Datenverkehrsmodelle

Die Leistungsfähigkeit eines Verbindungsnetzes kann nur im Kontext der Belastung gesehen werden, also dem Kommunikationsverkehr. Soll die Leistungsbewertung zur Beurteilung und Auswahl eines geeigneten Netzes vor der Realisierung genutzt werden, so ist der zu erwartende Verkehr nur in seltenen Fällen hinreichend genau bekannt. Insbesondere in Parallelrechnern schwankt die Verkehrsrate und die Verkehrscharakteristik von Programm zu Programm, so daß auf vereinfachende Modelle zurückgegriffen werden muß. In Kommunikationssystemen kann für Sprachverbindungen durch die Verbindungsannahme eine recht gute Aussage über die Art der zu erwartenden Verkehrsrate gemacht werden. Für neue Dienste wie Videoverbindungen sind die Eigenschaften noch nicht endgültig bekannt.

Nur in Ausnahmefällen ist die Messung von Netzen unter realen Verkehrsbedingungen möglich, so daß in der Regel eine Modellierung des Verkehrs vorgenommen werden muß. Unter einem solchen Verkehrsmodell sind dann die Leistungsparameter eines Verbindungsnetzes analytisch oder zumindest durch Simulationen faßbar. Oft werden unterschiedliche Verkehrsarten überlagert, um Problematiken aufzuzeigen.

Die Modellbildung soll die realen Bedingungen zwar vereinfachen, gleichzeitig jedoch eine möglichst sinnvolle und angemessene Abstraktion sein. Aus diesem Grund wurden zahlreiche Verkehrsmodelle entwickelt. Einige wichtige Modelle werden hier vorgestellt, die in Kapitel 6 zur Leistungsuntersuchung und Bewertung von Parallelrechner- und ATM-Verbindungsnetzen herangezogen werden.

4.1 Verkehrsmodelle für Parallelrechner

4.1.1 Gleichverteilter Verkehr

In diesem Verkehrsmodell ist eine Nachricht zu jedem Ziel mit gleicher Wahrscheinlichkeit gerichtet. Somit beträgt die Wahrscheinlichkeit $1/N$, daß eine Nachricht zu einem bestimmten von N Zielen geleitet wird; alle Ziele werden im statistischen Mittel gleich häufig angesprochen. Zur Modellierung dieses Verkehrs wird häufig die Bernoulli-Verteilung benutzt, die von einem Zufallsexperiment mit nur zwei möglichen Ergebnissen, X=0 und X=1, ausgeht. Hat das Experiment mit Wahrscheinlichkeit λ das Resultat $X = 0$, so gilt dann für die Auftrittswahrscheinlichkeiten P der Ereignisse X=0 und X=1:

$$P\{X=0\} = \lambda \qquad P\{X=1\} = 1 - \lambda.$$

In synchronen paketvermittelnden Netzen ist diese Verteilung sehr wichtig, um die Nachrichtenerzeugung am Netzeingang zu modellieren. An jedem Eingang kann ein Paket in einem Zyklus (Zeitschlitz für die Übertragung eines Pakets) eingespeist werden, oder der Eingang

bleibt während des Zyklus ungenutzt. Dann ist λ ($0 \leq \lambda \leq 1$) die auf eine Zykluszeit bezogene **Verkehrsrate** an einem Netzeingang. Beim Modell des gleichverteilten Verkehrs wird weiterhin angenommen, daß alle dem Netz angebotenen Verkehrsströme die gleiche Verkehrsrate λ besitzen. Dieses Verkehrsmodell wird insbesondere bei analytischen Methoden zur Leistungsbewertung von Netzen weit verbreitet angewendet (z. B. in [DiB94, Jen83, KuJ86, Pat81, Tur93, YoL90]).

4.1.2 Hot-Spot-Verkehr

Das Modell des gleichverteilten Verkehrs ist einfach und sehr gut für die analytische Leistungsanalyse von Verbindungsnetzen geeignet. Für viele Situationen beim Betrieb von Parallelrechnern ist dieses Modell jedoch nicht ausreichend. So existieren Verkehrsszenarien, bei denen bestimmte Senken (die **Hot-Spots**) häufiger angesprochen werden als andere. Diese Verkehre können durch statische und temporäre **Hot-Spot-Verkehrsmodelle** beschrieben werden.

4.1.2.1 Statischer Hot-Spot-Verkehr

Bei diesem Verkehrsmodell wird ein gleichverteilter Hintergrundverkehr mit der Rate λ angenommen. Ein Bruchteil h ($0 \leq h \leq 1$) dieses Verkehrs ist zu einem Hot-Spot gerichtet. Pakete, die zu dem Teil h des Verkehrs gehören, werden auch **heiße Pakete** genannt, während die restlichen Pakete **kalte Pakete** sind. So sendet jeder Prozessor mit einer Rate von $h\lambda$ heiße Nachrichten zum Hot-Spot, während die übrigen Nachrichten mit gleicher Wahrscheinlichkeit zu allen N Senken gerichtet sind. Demnach ist die Gesamtverkehrsrate am Hot-Spot gleich

$$\lambda_h = \lambda * (hN + 1 - h). \qquad (1)$$

Eine Senke kann jedoch nur maximal einen Zugriff pro Netzzyklus bedienen, so daß bei genügend großem h eine Überlast entsteht. Die asymptotisch maximale Verkehrsrate pro Prozessor λ_{h0}, bei der noch keine Überlast entsteht, ist dadurch charakterisiert, daß die Last am Hot-Spot maximal ist (λ_h=1). Dann gilt

$$\lambda_{h0} = \frac{1}{hN + 1 - h}. \qquad (2)$$

Bei dieser Last kann der Hot-Spot also im Mittel alle eingehenden Anfragen bedienen, jedoch führt die hohe Last zu einer starken Pufferauslastung, insbesondere in den Stufen in der Nähe des Netzausgangs. Übersteigt λ_h den Wert 1, so kann der Hot-Spot nur noch einen Teil der eingehenden Nachrichten schnell genug bearbeiten. In einem indirekten mehrstufigen Block-Netz mit

begrenzter Pufferkapazität pro Koppelelement werden dann zunächst die Pufferplätze im Koppelelement gefüllt, das mit dem Hot-Spot verbunden ist. Sobald diese Plätze belegt sind, füllen sich die Puffer aller Koppelelemente, die mit dem Koppelelement am Ausgang verbunden sind, usw. Da gemäß dem Hot-Spot-Verkehrsmodell alle Prozessoren Nachrichten zum Hot-Spot senden, füllen sich die Puffer auch in der Eingangsstufe, so daß schließlich alle Nachrichten verzögert werden, ob sie zum Hot-Spot gerichtet sind oder nicht. Somit bildet sich ein **Überlastbaum** (**Sättigungsbaum**) im Netz, der sich von dem Hot-Spot bis zu allen Netzeingängen erstreckt. Die Puffer in allen Koppelelementen des Baums sind gefüllt.

Die **Tiefe *R* des Sättigungsbaums** in einem s-stufigen indirekten Netz ($0 < R \leq s$) kann allgemein als die Anzahl der Netzstufen, über die sich der Baum erstreckt, definiert werden. So beginnt ein Sättigungsbaum der Tiefe R in einem s-stufigen indirekten Netz an einem Ausgangslink der Stufe s–R und erstreckt sich bis zu den Netzeingängen. In einem indirekten Netz mit s Stufen kann sich demnach ein Sättigungsbaum der Tiefe R_h mit

$$R_h = s \tag{3}$$

unter statischem Hot-Spot-Verkehr ausbilden.

Das Modell des statischen Hot-Spot-Verkehrs geht von einer kontinuierlichen Last am Hot-Spot aus. Für reale Systeme ist diese Annahme nicht gerechtfertigt, da nach Entstehen des Sättigungsbaums weitere Anfragen verlangsamt werden, was auch die Rechengeschwindigkeit verlangsamt. Insbesondere für das konzeptionelle Verständnis ist dieses Modell jedoch gut geeignet. In realen Systemen ist die Lebensdauer eines bestimmten Hot-Spots in der Regel jedoch begrenzt. So können beispielsweise alle Prozessoren an einer Stelle eines parallelen Programms eine Anzahl von Zugriffen auf den Hot-Spot durchführen. Zwar gilt das obige Verkehrsmodell noch, jedoch nur für einen begrenzten Zeitraum, so daß sich dynamische Effekte des Auf- und Abbaus von Hot-Spots ergeben [KuP86]. Eine Möglichkeit, dieses dynamische Verkehrsverhalten mit zu berücksichtigen, ist die Benutzung des Modells des normalverteilten Hot-Spot-Verkehrs [AbP89, WaS95], welches im nächsten Abschnitt beschrieben wird.

4.1.2.2 Normalverteilter Hot-Spot-Verkehr

Dieses Modell berücksichtigt das dynamische Verhalten von temporären Hot-Spots, die z. B. bei der Synchronisation von Prozessoren auftreten können. Die Synchronisation an einer Stelle in einem parallelen Programm auf einer Maschine mit gemeinsamem Speicher ist zum Beispiel dadurch möglich, daß alle Prozessoren auf eine Synchronisationsvariable zugreifen, den Wert um eins erhöhen und anhand des Wertes überprüfen, ob alle Prozessoren den Zugriff bereits durchgeführt haben [Hwa84, Per87, Sto87]. Nach dem Zugriff auf die Synchronisationsvariable führen sie andere Programme aus, so daß während der Wartezeit also von allen Prozessoren wei-

terhin Nachrichten generiert werden, die zum Beispiel dem gleichverteilten Verkehrsmodell folgen können. Der letzte synchronisierte Prozessor verschickt dann eine Broadcast-Nachricht an alle anderen Prozessoren, so daß der Synchronisationsprozeß abgeschlossen ist und alle Prozessoren synchronisiert mit der Programmausführung fortfahren. Ein ähnliches Verhalten kann auch für die Zuteilung von Verarbeitungsaufgaben genutzt werden. Hierbei kann z. B. durch eine gemeinsame Variable ein Wartepuffer mit neuen Aufgaben verwaltet werden, so daß auch hier die Prozessoren diese Variable abfragen und verändern müssen.

In beiden Szenarien wird also pro Prozessor nur eine Nachricht an den Hot-Spot gesandt. Da die Prozessoren den Synchronisationspunkt zu unterschiedlichen Zeiten erreichen, kann dieses Hot-Spot-Verhalten nährungsweise durch eine Normalverteilung der Zugriffe beschrieben werden. In diesem Modell erreichen die Prozessoren den Synchronisationspunkt im Mittel zum Zeitpunkt μ, mit einer Streuung σ. Die Wahrscheinlichkeitsfunktion des Zugriffs ist in Abbildung 4.1 gezeigt. Auch bei einem solchen Modell ergibt sich ein Hot-Spot-Verhalten, das zeitlich begrenzt ist und einem realistischen Anwendungsszenario näher kommt. Die Erfassung der Konsequenzen dieses Verkehrsmodells ist jedoch ungleich schwieriger als die des statischen Modells, insbesondere wegen der dynamischen Effekte.

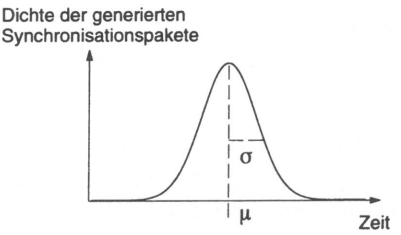


Abbildung 4.1: *Verteilung der Synchronisationspakete bei einem normalverteilten Hot-Spot-Verkehr.*

4.1.3 Permutationsverkehr

Bei vielen Algorithmen müssen Daten zwischen spezifischen Prozessoren ausgetauscht werden. Insbesondere bei Programmen, die auf SIMD-Rechnern laufen, können hierbei eine größere Anzahl von Prozessoren beteiligt sein, die gleichzeitig kommunizieren. Ist M_Q die Menge aller Quelladressen und M_S die Menge aller Senkenadressen in einem Parallelrechner, so ist im allgemeinen Fall eine **Verbindungsfunktion** eine **Abbildung** einer Teilmenge $D(M_Q) \subset M_Q$ auf eine Teilmenge $D(M_S) \subset M_S$ anhand der Vorschrift f: $M_Q \rightarrow M_S$. Dabei weist f jedem Element x_1 aus der **Definitionsmenge** $D(M_Q)$ eindeutig ein Element x_2 aus der **Bildmenge** $B(M_S)$ zu.

Dabei gilt:

$$B(M_S) = \{ x_2 : x_2 = f(x_1) \text{ für ein } x_1 \in D(M_Q) \}.$$

D und B können echte Teilmengen von M_Q bzw. M_s sein, so daß also nur ein Teil der Quelladressen durch die Verbindungsfunktion auf einen Teil der Senkenadressen abgebildet wird. Physikalisch bedeutet dies, daß beim Ausführen der Verbindungsfunktion f nur ein Teil der Quellen mit einem Teil der Senken kommuniziert.

Bei einer Abbildung können zwei oder mehr Elemente in D auf das gleiche Element in B abgebildet werden, d. h. physikalisch senden zwei oder mehr Quellen Daten zur selben Senke. Dieser Fall wird durch die **eineindeutige Abbildung** eingeschränkt. Hierbei folgt für alle Elemente von D aus der Beziehung $f(x_1) = f(x_1')$, daß $x_1 = x_1'$ gilt. Dies bedeutet, daß eine Quelle mit einer Adresse aus der Menge D mit genau einer Senke mit einer Adresse aus B kommuniziert. Ein spezieller Fall der eineindeutigen Abbildung ist die **Permutation**, bei der Definitions- und Bildmenge identisch sind ($M_Q = M_S$); es wird also eine geordnete Liste von Elementen in eine neue Ordnung überführt. Physikalisch kommuniziert bei der Ausführung einer Permutation also jeder Knoten einer Teilmenge D der verfügbaren Knoten mit genau einem Knoten derselben Teilmenge, wobei nie mehrere Quellenknoten mit einem gleichen Senkenknoten kommunizieren. Beispiele von Permutationsverkehren in Parallelrechnern sind die Bit-Reverse-Permutationen (eine Unterklasse der Bit-Permute-Complement Permutationen [NaS82]) und die Perfect Shuffle Permutationen [Len78], die z. B. bei der Berechnung von Fast Fourier–Transformationen auftreten [Sto71].

Die **Bit-Reverse-Permutation** zeichnet sich dadurch aus, daß eine Quelle mit der Adresse $Q=q_{s-1},q_{s-2},\ldots,q_1,q_0$ mit der Senke $S=q_0,q_1,\ldots,q_{s-2},q_{s-1}$ kommuniziert. Da die Bit-Reverse-Permutation zu starken Konflikten in Delta-Netzen führt [JuS93, KiL90, LaK88], wird dieser Typ von Permutation (zusammen mit den Hot-Spot-Verkehren) zur Leistungsbestimmung von Delta-Netzen in Kapitel 6 verwendet und in den nächsten Abschnitten näher beschrieben.

4.1.3.1 *Statischer Bit-Reverse-Permutationsverkehr*

Bei diesem Verkehrsmodell wird wiederum ein gleichverteilter Hintergrundverkehr mit der Rate λ angenommen. Ein Bruchteil h ($0 \le h \le 1$) dieses Verkehrs führt hierbei eine Bit-Reverse-Permutation aus. Pakete, die zu dem Teil h des Verkehrs gehören, werden wie beim Hot-Spot-Verkehr heiße Pakete genannt, während die restlichen Pakete kalte Pakete sind. Dieser Verkehr führt in Delta-Netzen zu **netzinternen Konfliktpunkten (nonuniform traffic spots – NUTS)** [LaK88]; die Konfliktpunkte sind in Abbildung 4.2 in einem 8×8 Delta-Netz unter Bit-Reverse-Permutationsverkehr gezeigt.

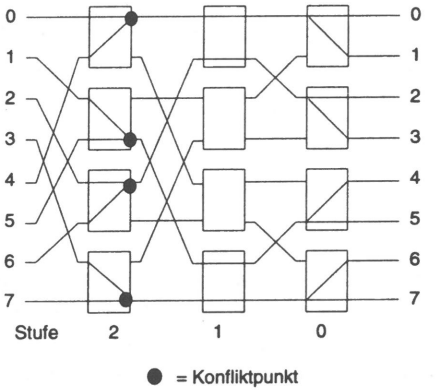


Abbildung 4.2: *Delta-Netz mit 8 Ein- und Ausgängen unter Bit-Reverse-Permutationsverkehr.*

In Delta-Netzen unter Bit-Reverse-Permutationsverkehren werden allgemein die Permutationsnachrichten, die ein Koppelelement in der vorderen Netzhälfte betreten, von allen Elementeingängen auf einen Elementausgang konzentriert. In der Mittelstufe bei Netzen mit einer ungeraden Anzahl von Stufen werden die Daten permutiert, während sie in der hinteren Netzhälfte auf die einzelnen Koppelelementausgänge wieder verteilt werden. Die konvergierenden und divergierenden Wege der Permutationsnachrichten in einem fünfstufigen Delta-Netz, aufgebaut aus 4×4 Koppelelementen, sind in Abbildung 4.3 gezeigt.

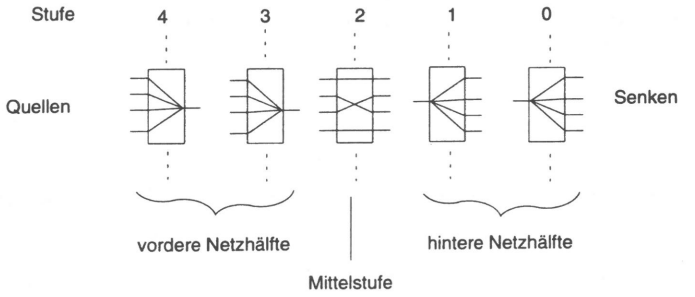


Abbildung 4.3: *Routing in einem Delta-Netz unter Bit-Reverse-Permutationsverkehr.*

So werden in einem s-stufigen Delta-Netz, aufgebaut aus $B \times B$ Koppelelementen, die Permutationsnachrichten von $B^{\lfloor s/2 \rfloor}$ verschiedenen Quellen auf einem heißen Ausgangslink der letzten Stufe in der vorderen Netzhälfte konzentriert. Das statische Bit-Reverse-Permutationsverkehrsmodell ist demnach durch die **Bit-Reverse-Verkehrslast** λ_b auf diesen heißen Ausgangslinks charakterisiert:

$$\lambda_b = \lambda * (h * B^{\left\lfloor \frac{s}{2} \right\rfloor} + 1 - h). \tag{4}$$

Ein Ausgangslink eines Koppelelements kann jedoch nur maximal eine Nachricht pro Netzzyklus weiterleiten, so daß bei genügend großem h, ähnlich wie beim statischen Hot-Spot-Verkehr, eine Überlast entsteht. Die asymptotisch maximale Verkehrsrate pro Prozessor λ_{b0}, bei der noch keine Überlast entsteht, ist dadurch charakterisiert, daß die Last an den Ausgangslinks der letzten Stufe in der vorderen Netzhälfte maximal ist (λ_b=1). Dann gilt

$$\lambda_{b0} = \frac{1}{h * B^{\left\lfloor \frac{s}{2} \right\rfloor} + 1 - h}. \tag{5}$$

Bei dieser Rate können die Ausgangslinks also im Mittel alle eingehenden Anfragen weiterleiten, jedoch führt die hohe Last zu einer starken Pufferauslastung in der vorderen Hälfte des Netzes. Übersteigt λ_b den Wert 1, so kann die letzte Stufe der vorderen Netzhälfte nur noch einen Teil der eingehenden Nachrichten schnell genug bearbeiten. In einem indirekten mehrstufigen Block-Netz mit begrenzter Pufferkapazität pro Koppelelement bauen sich dann, ähnlich wie unter Hot-Spot-Verkehr, Sättigungsbäume gefüllter Puffer auf. Diese **partiellen Sättigungsbäume** beginnen an den heißen Ausgängen der vorderen Netzhälfte und erstrecken sich bis zu den Netzeingängen. Die Tiefe dieser Bäume kann demnach zu R_b mit

$$R_b = \left\lfloor \frac{s}{2} \right\rfloor \tag{6}$$

berechnet werden. Sind alle N Prozessoren an der Permutation beteiligt, so können insgesamt

$$N_{ps} = \frac{N}{B^{R_b}} \tag{7}$$

partielle Sättigungsbäume im Netz entstehen.

4.1.3.2 *Temporärer Bit-Reverse-Permutationsverkehr*

Auch das Modell des statischen Bit-Reverse-Permutationsverkehrs geht von einer kontinuierlichen Erzeugung der heißen Permutationsnachrichten aus. Für reale Parallelprogramme ist diese Annahme, wie auch beim statischen Hot-Spot-Verkehr, nicht gerechtfertigt. Dies wird beispielhaft durch die parallele **Fast-Fourier-Transformation** (**FFT**) [JaM86, MuS80, Pea68] verdeutlicht. Die FFT Algorithmen spielen eine bedeutende Rolle in der mathematischen und numerischen Analyse. Applikationen der FFT beinhalten z. B. Partikelsimulation, Löser von partiellen Differentialgleichungen, Löser von Poisson-Gleichungen und Digitalfilter [Cve87]. Die parallele FFT benötigt zwei Typen von Interprozessorkommunikation, die in einer bestimmten Reihenfolge mehrfach benutzt werden: die **Butterfly-Operation**, in dem Prozessorpaare Daten austauschen, und die **Reverse-Binary-Transformation**, in dem die Prozessoren über eine Bit-Reverse-Permutation Daten austauschen [Sto87]. So gilt zwar das Verkehrsmodell der statischen Bit-Reverse-Permutation in diesem Fall noch, jedoch nur für begrenzte Zeiträume.

Eine Möglichkeit, diesen temporären Bit-Reverse-Permutationsverkehr zu modellieren, ist die Korrelation des Bit-Reverse-Verkehrsanteils h am Gesamtverkehr mit der Zeit. So wird in dieser Arbeit ein temporärer Bit-Reverse-Verkehr mit konstanter Rate λ angenommen, bei dem der Verkehrsparameter h nur in einem Zeitintervall $T_{begin} \leq t \leq T_{end}$ von Null verschieden ist:

$$h = \begin{cases} 0, & 0 \leq t < T_{begin} \\ h_0, & T_{begin} \leq t \leq T_{end} \\ 0, & t > T_{end} \end{cases} \quad . \tag{8}$$

Ein normalverteilter Bit-Reverse-Permutationsverkehr kann zur Modellierung dieser Effekte ebenfalls herangezogen werden.

4.2 MPEG-Codierte Videoströme in ATM-Kommunikationssystemen

Durch die Einführung von Videodiensten und Multimedia-Anwendungen in Breitbandkommunikationssystemen wird ein großer Teil des Datenverkehrs in ATM-Netzen durch die Übertragung von Videoströmen bestimmt. Da diese Ströme in vielen Fällen eine variable Bitrate aufweisen, stellen sie erhöhte Anforderungen an das ATM-Netz, so daß zur Leistungsbestimmmung von ATM-Netzen unter anderem dieser Datenverkehr herangezogen wird [Ens94, NoF89, PaZ93, RaS90, SkS93].

Die Charakteristiken dieser Videoströme hängen wesentlich von der verwendeten Datencodierung ab. Ein weit verbreiteter Video-Codierungsalgorithmus ist der MPEG-1-Algorithmus [Iso92]. Diese Methode benutzt Intra- und Interframecodierungen, um einen wahlfreien Zugriff innerhalb der Videosequenz zu ermöglichen. Hierbei werden drei verschiedene Bildtypen verwendet: das I-Bild, das P-Bild und das B-Bild. **I-Bilder (intracoded)** werden isoliert (ohne Referenzen zu anderen Bildern) codiert. Bewegungsschätzungen mit Bezug auf vorhergehende I- oder P-Bilder werden in den **P-Bildern (predictive coded)** codiert, während Bewegungsschätzungen mit Bezug auf vorhergehende und nachfolgende Bilder in den **B-Bildern (bidirectionally-predictive coded)** codiert werden. Eine Videosequenz besteht aus der Aneinanderreihung einer Bildergruppe (**group of pictures, GOP**), die aus einer festen Anzahl von I-, B- und P-Bildern besteht. Im weitern wird eine GOP mit der Bilderfolge IBBPBBPBB [Ens95] angenommen.

Zur Herleitung von mathematischen Verkehrsmodellen ist unter anderem das Starwars-Video herangezogen worden [Ens95, GaV91], da es eine Variabilität seines Datenvolumens aufweist. Die Statistik der Bildgrößen dieses Videos wurden in [Ens95] durch Messungen an einem MPEG-Codierer bestimmt und sind in Abbildung 4.4 zusammengefaßt.

Bildtyp	Minimum [*bit*]	Maximum [*bit*]	Mittel [*bit*]	Variationskoeffizient
I	44520	387656	$1.82*10^5$	0.24
B	14352	394920	$1.03*10^5$	0.33
P	2720	144048	$4.41*10^4$	0.32
gesamt	2720	394920	$7.25*10^4$	0.72

***Abbildung 4.4:** Statistik der Bildgrößen des Starwars-Videos [Ens95].*

Die mathematische Modellierung des Starwars-MPEG-Videostromes auf Szenen-, Bild- und Paketebene ist in [Ens95] beschrieben. Das im weitern verwendete Videoquellenmodell ist in Abbildung 4.5 dargestellt und stellt eine Vereinfachung des Modells aus [Ens95] dar. Es ist zweistufig aufgebaut und besteht aus einem Bildgrößengenerator und einer Paketiereinheit. In [Ens95] wurde gezeigt, daß sich in ATM-Netzen der Einfluß der Videoszenenaktivitäten auf die Netzleistung erst bei sehr großen Pufferlängen (größer 200 Pakete pro Puffer) in den ATM-Koppelelementen bemerkbar macht. Da in den Untersuchungen in Kapitel 6 nur Koppelelemente mit geringeren Pufferlängen betrachtet werden, kann der Einfluß der Szenenaktivitäten vernachläßigt werden, so daß diese im Modell nicht mitberücksichtigt werden müssen.

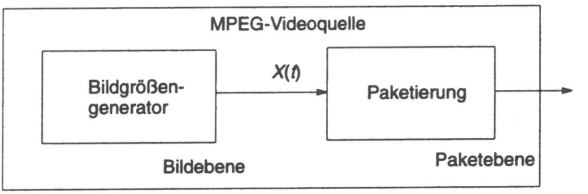


Abbildung 4.5: *Vereinfachtes MPEG-Videoquellenmodell [Ens95].*

Sind die statistischen Größen der einzelnen Bildtypen einer Videosequenz bekannt, so können die log-normalverteilten Bildgrößen zu

$$X(t) = \exp\left[\sqrt{\ln\left(1 + \frac{\mathrm{Var}[X_i]}{\mathrm{E}[X_i]^2}\right)}\; N(0,1) + \ln \mathrm{E}[X_i] - 0.5 \ln\left(1 + \frac{\mathrm{Var}[X_i]}{\mathrm{E}[X_i]^2}\right)\right] \quad (9)$$

berechnet werden [Ens95] (mit $i \in \{ I, B, P \}$ und $N(0,1)$ einer Normalverteilung mit dem Mittel 0 und der Varianz 1), durch die der Bildgrößengenerator beschrieben wird. Der Paketierer berechnet schließlich aus den vom Bildgrößengenerator kommenden Bildgrößen die Generierungszeiten der Datenpakete, wobei angenommen wird, daß die Zwischenankunftsabstände der Datenpakete eines Bildes gleich groß sind.

Dieses Videoquellenmodell wird in Abschnitt 6.4.3 zusammen mit den Starwars-Videodaten aus Abbildung 4.4 benutzt, um die Leistung von ATM-Netzen unter Videoströmen zu bestimmen und zu verbessern.

5 Leistungsbestimmung von Netzen

Die Gesamtleistung eines Parallelrechners wird entscheidend durch die Leistung des verwendeten Verbindungsnetzes beeinflußt. Entwurfsparameter, wie z. B. Netztopologie und Architektur, die leistungsbestimmend für das Netz sind, müssen bereits in der Netzentwurfsphase festgelegt werden. Leistungsparameter des Netzes, die vom Datenverkehr im Parallelrechner abhängen können, müssen deshalb in der Entwurfsphase bestimmt werden, um zu entscheiden, ob das gewählte Netz den Leistungsanforderungen des Parallelrechners genügt. Die wichtigsten Leistungsparameter und deren Bestimmung sind in den folgenden Abschnitten beschrieben.

5.1 Leistungsparameter von paketvermittelnden Verbindungsnetzen

Um die Leistungsfähigkeit von Verbindungsnetzen beurteilen und vergleichen zu können, müssen Leistungsparameter definiert und bestimmt werden, die repräsentative Leistungsmerkmale von Netzen und Leistungsanforderungen von Applikationen und Algorithmen realistisch wiedergeben. Die in dieser Arbeit verwendeten Leistungsparameter für paketvermittelnde Netze werden nun kurz vorgestellt.

Ein wichtiges Leistungsmerkmal von paketvermittelnden Netzen ist die **mittlere Durchlaufzeit**, die Pakete im Quellknoten und im Netz erfahren. Diese Zeit setzt sich zusammen aus den Wartezeiten eines Pakets in den Quellknoten- und Koppelelementpuffern, aus den Vermittlungszeiten vom Quellknoten zum Netz und durch die Koppelelemente, und aus den Laufzeiten auf den Zwischenleitungen. Diese mittlere Paketdurchlaufzeit bezieht sich auf Paketklassen (z. B. alle Pakete, heiße Pakete, kalte Pakete) und kann simulationstechnisch bestimmt werden zu:

$$\text{mittlere Paketdurchlaufzeit} = \frac{\text{Summe aller Bedien- und Wartezeiten der insgesamt an den Netzausgängen empfangenen Pakete einer Klasse}}{\text{Anzahl der insgesamt an den Netzausgängen empfangenen Pakete der Klasse}}$$

Weiterhin kann die Summe aller Bedien- und Wartezeiten eines Pakets in der Quelle und im Netz als die zeitliche Differenz vom Generierungszeitpunkt zum Empfangszeitpunkt des Pakets am Netzausgang ausgedrückt werden.

In manchen Parallelrechnerarchitekturen, wie auch in Kommunikationssystemen, können Paketverluste im Verbindungsnetz und/oder in den Quellknoten auftreten. Blockierungen in den Koppelelementen werden in vielen Fällen durch Zwischenpufferung der Pakete in endlich langen Puffern aufgelöst. Wenn es zu einem Pufferüberlauf kommt, müssen Pakete entweder warten, oder sie werden verworfen. Verluste in den Quellen treten auf, wenn durch eine Netzüberlast (das Netz kann den angebotenen Verkehr nicht mehr vollständig weiterleiten) Pakete im Quell-

knoten in einem endlich langen Puffer zwischengepuffert werden müssen. Bei Pufferüberläufen müssen Pakete dann in den Quellknoten verworfen werden. Die so entstehenden Paketverluste in den Quellen und/oder im Netz können durch die **mittlere Paketverlustwahrscheinlichkeit** und durch den **mittleren relativen Datendurchsatz** quantifiziert werden. Der Datendurchsatz bezieht sich auf Paketklassen (z. B. alle Pakete, heiße Pakete, kalte Pakete) und kann simulationstechnisch bestimmt werden durch:

$$\text{mittlerer Datendurchsatz} = \frac{\text{Anzahl der insgesamt an den Netzausgängen empfangenen Pakete einer Klasse}}{\text{Anzahl der insgesamt generierten Pakete einer Klasse}}$$

Ein weiterer Leistungsaspekt ist die **mittlere Auslastung des Netzes**, welche als das Verhältnis von Abgangsrate der Pakete an den Netzausgängen zu der Kapazität des Netzes definiert ist. Simulationstechnisch kann die Auslastung folgendermaßen bestimmt werden:

$$\text{mittlere Netzauslastung} = \frac{\text{Anzahl der insgesamt an den Netzausgängen empfangenen Pakete}}{\text{Anzahl der Simulationszyklen * Netzgröße}}$$

Die Leistungsparameter sind vom Verkehrsaufkommen (Verkehrslast, Verkehrscharakteristik, etc.), ferner von der Netztopologie, der Architektur und der Realisierung abhängig. Sie können durch Messungen an Netzprototypen bestimmt werden, wobei der Kosten- wie auch der Zeitaufwand hierfür jedoch meist zu hoch ist. Eine Alternative bieten hier analytische und simulative Verfahren, die in den folgenden Abschnitten beschrieben werden.

5.2 Analyseverfahren zur Leistungsbestimmung

Bei der mathematischen Analyse zur Leistungsbestimmung von Netzen werden die Netze und die zu betrachtenden Datenverkehre abstrahiert und in mathematische Modelle überführt. Hierbei wird meist ein großer Zustandsraum aufgespannt, der in vielen Fällen mathematisch nicht mehr handhabbar ist. Um die Komplexität der Leistungsberechnung zu verkleinern, kann das Netzmodell deshalb in kleinere Einheiten, wie z. B. Bedieneinheiten und Puffer unterteilt werden. Wenn vereinfachende Annahmen über die Unabhängigkeit der Elemente getroffen werden, können diese Einheiten dann getrennt betrachtet werden. Grundelemente der Analyse sind einstufige Bediensysteme mit einer Bedieneinheit und einer meist endlich langen Warteschlange.

Die Leistungsprofile dieser Bediensysteme und des gesamten Netzes können mittels der **Warteschlangentheorie** (siehe z. B. [GrH81, Kle75, Lav83]) mathematisch bestimmt werden. Zur Klassifikation derartiger Modelle wird überwiegend die Kendallsche Notation benutzt, welche den Typ von Ankunfts- und Bedienprozeß, die Anzahl der Bedieneinheiten sowie die Pufferlänge angibt. In dieser Notation wird ein Modell durch die Kurzbeschreibung '$G^{[X]}/G/1$-S' charakterisiert, wobei 'G' für einen allgemeinen (general) Prozeß steht, das hochgestellte '[X]'

Gruppenankünfte spezifiziert und 'S' die Länge des Puffers (ohne Bedieneinheit) ist. Sofern keine anderen Angaben gemacht werden, wird eine FIFO-Warteschlange angenommen.

Mit dieser Theorie ist es möglich, Leistungscharakteristiken von Verbindungsnetzen zu bestimmen und seltene Ereignisse im Netz mathematisch zu erfassen. Allerdings kann dies nur durch die Benutzung von vereinfachten mathematischen Modellen der Netzelemente erreicht werden. So ist es meist nur möglich, komplexe Abläufe wie Arbitrierungsmechanismen und Auswahlverfahren als zufallsabhängige Entscheidungen, und Prozesse als gedächtnisfrei oder mit einer gedächtnisfreien Verteilung der Zustandsdauer zu modellieren. Auch können Bediensysteme mit allgemeinen Ankunfts- und Bedienzeiten nicht exakt analytisch berechnet werden, welches zu weiteren Einschränkungen der Bediensystemmodelle führt. Diese Annahmen und Vereinfachungen treffen in der Realität jedoch nur teilweise zu, so daß es nötig ist, analytische Ergebnisse, vorwiegend durch den Vergleich mit Simulationsergebnissen, zu validieren.

5.3 Simulationstechniken zur Leistungsbestimmung

Bei der Simulation von Netzen zur Leistungsbestimmung werden die Netze und die zu betrachtenden Datenverkehre abstrahiert, in Simulationsmodelle überführt und in einen Simulator eingebettet. Die zwei bedeutendsten Simulationskonzepte sind hierbei die zeitgesteuerte und die ereignisgesteuerte Simulation [Rat91, WiJ89]. In der **zeitgesteuerten Simulation** wird die fortschreitende Zeit in gleichlange Zeitschritte eingeteilt, wobei Ereignisse bei einem Wechsel von einem Zeitschritt zum nächsten abgearbeitet werden. Bei dieser Methode werden also Ereignisse synchron zu einem Zeittakt abgearbeitet. Bei der **ereignisgesteuerten Simulation** wird die Zeit zwischen zwei aufeinanderfolgenden Ereignissen, unabhängig von der Länge dieses Zeitraumes, übersprungen. Ereignisse werden, geordnet nach ihrem Ereigniszeitpunkt, in einer Ereignisliste (auch Kalender genannt) [Bro88] gespeichert und in der chronologischen Reihenfolge abgearbeitet.

Der Vorteil der Simulation gegenüber einer mathematischen Analyse von Verbindungsnetzen ist, daß komplexerer und realitätsnähere Netzmodelle verwendet werden können, welches in vielen Fällen zu realistischeren Ergebnissen führt. Nachteilig sind allerdings die manchmal sehr langen Simulationszeiten, wenn z. B. selten auftretende Ereignisse, oder, speziell in ATM-Netzen, Verlustwahrscheinlichkeiten in der Größenordnung von 10^{-9} erfaßt werden sollen. In solchen Fällen sind bis zu 10^{12} Ereignisse zu simulieren, so daß Laufzeiten von Simulatoren auf Workstations von mehreren Tagen bis Wochen keine Seltenheit sind. In den letzten Jahren sind verschiedene Konzepte entwickelt worden, um die Simulationszeiten zu verringern. Der Einsatz von spezieller Hardware zur Leistungssteigerung von Simulationen [FuT92, ReP93] ist meist unflexibel und kann nur bedingt für andere Applikationen verwendet werden. Spezielle Verfahren zur Simulation seltener Ereignisse [ViV91] können zwar Simulationszeiten verringern, sind jedoch weder für komplexere Netze noch für komplexere Datenverkehrsmodelle geeignet. Simulato-

ren, die auf Tansputernetzen [Des91] und auf Parallelrechnern implementiert sind [AyB93, GaG93, JuS96], bieten eine vielversprechende Alternative zu den restlichen Konzepten. Bei diesen Konzepten treten allerdings zusätzliche Probleme auf, die die Implementierung paralleler Simulatoren erschweren. Da in dieser Arbeit die parallele Simulation als Grundlage zur Leistungsbestimmung von Verbindungsnetzen herangezogen worden ist, wird im nächsten Abschnitt ausführlicher auf Anforderungen und Leistungsfähigkeit der parallelen Simulation eingegangen.

5.3.1 Aspekte der parallelen Simulation

Werden Simulatoren auf Parallelrechnern implementiert, so treten spezifische Probleme der Parallelisierung auf, die bei der sequentiellen Implementation nicht beachtet werden müssen. Diese Probleme und Leistungsbetrachtungen werden in den nächsten Abschnitten anhand der parallelen ereignisgesteuerten und der parallelen zeitgesteuerten Simulation vorgestellt.

5.3.1.1 Parallele ereignisgesteuerte Simulation

Bei dieser Simulationsmethode wird das modellierte System in Prozesse unterteilt, die auf unterschiedlichen Prozessoren des Parallelrechners laufen und untereinander Informationen ausschließlich in Form von Nachrichten austauschen können. Innere Zustände oder Variablen eines Prozesses sind anderen Prozessen nicht zugänglich. Die parallele ereignisgesteuerte Simulation erzielt den Leistungsgewinn dadurch, daß mehrere Ereignisse gleichzeitig abgearbeitet werden. Dabei kann es zu Kausalitätsverletzungen kommen, da nicht immer nur das Ereignis mit der kleinsten Zeitmarke im System abgearbeitet wird. Um diese Verletzungen zu verhindern, müssen geeignete Maßnahmen ergriffen werden. Hierzu stehen **konservative** und **optimistische Verfahren** zur Verfügung.

Konservative Verfahren

Konservative Verfahren vermeiden Kausalitätsverletzungen, indem Bedingungen bezüglich der Zeitmonotonie an den Simulator gestellt werden [Fuj90], die erfüllt sein müssen. Der Simulator muß feststellen, wann ein Ereignis sicher abgearbeitet werden kann, wobei nirgends eine Ereignis-Nachricht für den betreffenden Prozeß mit früherer Zeit-Marke vorhanden sein oder erzeugt werden darf. Um dies zu gewährleisten, müssen Prozesse auf andere warten. Da das Warten auch reihum durch wechselseitige Anforderungen auf Betriebsmittel vorkommen kann, können Verklemmungen auftreten. Chandy und Misra führen in ihrer Arbeit [ChM79] ein Verfahren ein, das Verklemmungen durch **Null-Nachrichten** vermeidet, während in [ChM81] eine alternative Methode vorgestellt wird, die Verklemmungen erkennt und behebt. [Mis86] gibt einen Überblick über existierende Algorithmen zur Verklemmungsvermeidung. Die Methode von Chandy und Misra wurde von verschiedenen Autoren untersucht, modifiziert und verbessert. In [LiL90, Lin92] werden Auswirkungen der Nullnachrichten auf die Simulatorleistung untersucht, wäh-

rend in [Vri90, Yu92] Methoden zur Verringerung der Anzahl der Nullnachrichten vorgestellt werden. Anwendungen und Leistungsbewertungen des konservativen Simulationsprotokolls sind in [HeN93, LiA93, Nic92, Kec94, RaF92, Raj92] beschrieben.

Optimistische Verfahren

Bei optimistischen Methoden wird jeweils in jedem Prozeß das Ereignis mit der kleinsten Zeitmarke abgearbeitet. Dabei wird *nicht* darauf geachtet, ob möglicherweise noch ein Ereignis mit kleinerer Zeitmarke, ein **Nachzügler**, empfangen werden kann. Alle nach der Zeitmarke des Nachzüglers durchgeführten Aktionen eines Prozesses sind möglicherweise falsch, da der Nachzügler die Ausgangssituation geändert hat. Bei der Methode des **Time Warp** [Jef85] wird dann der Zustand wiederhergestellt, den das System vor der Zeitmarke des Nachzüglers hatte. Dieser **Rollback-Vorgang** macht eine periodische Zustandsspeicherung der einzelnen Prozesse erforderlich. Auch wurde in [Jef85] der Begriff der **globalen virtuellen Zeit** eingeführt, um zu entscheiden, bis zu welchem Zeitpunkt der Vergangenheit die Prozeßzustände gespeichert werden müssen. Algorithmen zur Berechnung der virtuellen Zeit und der Anzahl der zu speichernden Zustände pro Prozeß werden z. B. in [BaS93, LiP93, Mat93, ToG93] vorgestellt, während Methoden zur Reduzierung der Anzahl von Rollbacks in [SoS93, Ste93] beschrieben sind. Anwendungen, Leistungsbewertungen und Leistungsoptimierungen des Time-Warp-Protokolls sind in [AkC93, KrT92, NiH93, GlT93, GuA91, PaA93, RoA93] beschrieben.

Leistungsvergleich der Verfahren

Leistungsvergleiche zwischen konservativen und optimistischen Methoden zeigen, daß die Wahl der optimalen Methode von der Applikation abhängig ist [Fuj90, Luk93, Pre90]. Das *Time-Warp*-Verfahren bietet ein größeres Potential für allgemeine Anwendungen. Konservative Verfahren sind jedoch für viele spezielle Anwendungen ebenso gut geeignet [Fuj90]. *Time Warp* erfordert jedoch einen größeren Implementierungsaufwand. Systemteile wie die Speicherverwaltung, und die Zustandsspeicherung und Buchführung über verschickte Nachrichten sind jeweils für sich schon komplexe Subsysteme. Da es sich jedoch um ein sehr universelles Verfahren handelt, müssen diese Komponenten eines *Time-Warp*-Systemkernes nur einmal entwickelt werden. Der Aufwand für konservative Methoden ist deutlich geringer, da alle Buchführungsfunktionen wegfallen, weil keine Operationen rückgängig gemacht werden müssen. Sehr nachteilig ist jedoch die enge Verknüpfung des Simulators mit dem Systemmodell, so daß oft kleine Änderungen im Systemmodell größere Programmodifikationen erfordern und deutliche Auswirkungen auf die Leistung des Simulators haben.

Die asynchrone Abarbeitung der Ereignisse in der parallelen ereignisgesteuerten Simulation hat große Auswirkung auf die Wahl des zu verwendenden Parallelrechners. MIMD-Rechner mit deren asynchronen Prozessoren sind sehr gut geeignet, die asynchronen Ereignisse zu verarbeiten. In SIMD-Maschinen, in denen alle Prozessoren synchron zu einem Systemtakt arbeiten,

müssen spezielle Algorithmen implementiert werden, die die asynchrone Verarbeitung der Ereignisse unterstützen [AyB93]. Deshalb sind SIMD-Rechner nur bedingt für die parallele ereignisgesteuerte Simulation geeignet.

5.3.1.2 Parallele zeitgesteuerte Simulation

Das Hauptmerkmal der zeitgesteuerten Simulation ist die synchrone Abarbeitung der Simulationsereignisse. Da dies in der parallelen Simulation ebenfalls geschehen muß, müssen auch die einzelnen Prozesse, die auf unterschiedlichen Prozessoren des Parallelrechners laufen, synchron sein. Auf MIMD-Rechnern ist es deshalb erforderlich, die Prozessoren bei jedem Simulationsschritt zu synchronisieren, wodurch Leistungsverluste entstehen. In SIMD-Rechnern arbeiten die Prozessoren synchron zueinander. Da die parallele zeitgesteuerte Simulation diese implizite Synchronisation ausnutzen kann, sind SIMD-Rechner für die zeitgesteuerte Simulation sehr gut geeignet. Bei der zeitgesteuerten Simulation auf SIMD-Rechnern brauchen keine komplexen Verwaltungsfunktionen implementiert zu werden. Hier sind vielmehr die Methoden zur Abbildung von Netztopologien auf die Parallelrechnerknoten von größerer Bedeutung, da die Wahl der Abbildung die Simulatorleistung im großen Maße beeinflussen kann. Da in dieser Arbeit vorwiegend die Leistungscharakteristiken von synchronen Netzen mittels Simulationen auf einem SIMD-Rechner bestimmt worden sind, werden im nächsten Abschnitt Implementierungs- und Leistungsaspekte von parallelen, zeitgesteuerten Simulatoren auf SIMD-Rechnern vorgestellt.

5.3.2 Zeitgesteuerte Simulation von synchronen Delta-Netzen auf SIMD-Parallelrechnern

Es existieren viele Möglichkeiten, ein mehrstufiges Delta-Netz mit seinen Koppelelementen auf einen massiv-parallelen SIMD-Parallelrechner abzubilden, um einen parallelen Netzsimulator zu implementieren. Ein Unterscheidungsmerkmal der verschiedenen Abbildungsmöglichkeiten ist hierbei die **Abbildungsgranularität** [JuS96], welche als Grad der Parallelität einer Abbildung definiert ist. Zur Verdeutlichung sind drei Abbildungsmöglichkeiten mit unterschiedlicher Granularität in Abbildung 5.1 gezeigt. Hierbei wird von einer Koppelelementarchitektur ausgegangen, die an jedem Ein- und Ausgangsport einen Paketpuffer besitzt. Bei der **feingranularen Abbildung** (**Linkabbildung**, Abbildung 5.1a) sind die Zwischenleitungen inklusive der Eingangs- und Ausgangspuffer der Koppelelemente auf die Prozessoren abgebildet. Bei der **mittelgranularen Abbildung** (**Schalterabbildung**, Abbildung 5.1b) befindet sich ein Koppelelement des Netzes in jedem Prozessor, während bei der **grobgranularen Abbildung** (**Teilnetzabbildung**, Abbildung 5.1c) mehrere Koppelelemente in jedem Prozessor simuliert werden. Die Parallelität der Abbildungen nimmt hierbei von der feingranularen zur grobgranularen Abbildung ab. Leistungsfähigkeit und Anforderungen dieser Abbildungsstrategien werden nun diskutiert.

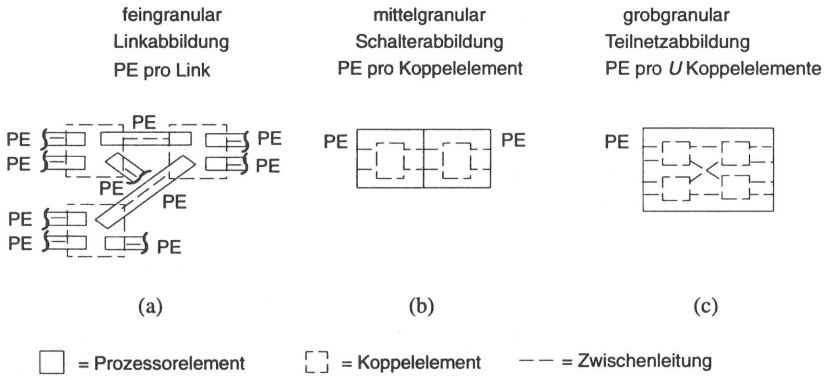


Abbildung 5.1: *Netz-Abbildungsmöglichkeiten: (a) feingranulare, (b) mittelgranulare und (c) grobgranulare Abbildung*

Um die Leistung des vorhandenen Parallelrechners vollständig auszunützen, wird angenommen, daß mehrere unabhängige Simulationen des gleichen Netzes parallel auf der Maschine laufen und deren Ergebnisse gemittelt werden. Um Konflikte unter diesen Subsimulationen zu vermeiden, benutzt jeder Subsimulator andere PEs. Wenn eine Netzabbildung P Prozessorelemente benötigt, während im Parallelrechner N_S Prozessorelemente zur Verfügung stehen ($N_S \geq P$), so können $\lfloor N_S/P \rfloor$ Netzsubsimulatoren parallel auf der Maschine laufen.

5.3.2.1 Abbildungsgranularität

Je feiner die Granularität der verwendeten Abbildung ist, desto größer ist die Anzahl der pro Subsimulator benutzten PEs. Dies bedingt eine relativ kleine Anzahl von parallelen Subsimulatoren bei der feingranularen Abbildung. Bei einer Abbildung mit gröberer Granularität werden weniger PEs pro Subsimulator benutzt, welches in einer höheren Ausführzeit für einen Simulationslauf resultiert. Allerdings können mehr Subsimulatoren auf dem Parallelrechner implementiert werden als bei einer Abbildung mit feinerer Granularität. Demnach können also mit der feingranularen Abbildungsstrategie wenige, aber schnelle Subsimulatoren implementiert werden, während man mit den mittel- und grobgranularen Abbildungen viele, aber langsame Subsimulatoren erhält. Durch dieses gegenläufige Verhalten läßt sich nicht feststellen, welche Abbildungsstrategie zu dem schnellsten Simulator führt. Deshalb müssen die Ausführzeiten der verschiedenen Simulatoren gemessen werden, um Aussagen über die Schnelligkeit der Abbildungsstrategien treffen zu können.

5.3.2.2 Kommunikationsaufwand

Die im Parallelrechner vorhandenen Verbindungsnetze können die Wahl der optimalen Abbildungsstrategie beeinflussen, wenn die Transferzeiten durch das Netz von der Transferdistanz abhängig sind. Wird ein lokales Netz (z. B. ein Gitternetz) für die Kommunikation der Simulator-PEs verwendet, so kann die Transferzeit von der Transferdistanz abhängig sein, wenn kein Wormhole Routing [DrG94, GrO93, LiE93, NiM93] im Gitternetz implementiert ist. Je mehr PEs für einen Subsimulator verwendet werden, desto größer wird die mittlere Kommunikationsdistanz zwischen den PEs. In diesem Fall ist eine grobgranulare Abbildung, die wenige PEs pro Subsimulator verwendet, einer feingranularen Abbildung bei Betrachtung des Kommunikationsaufwandes vorzuziehen. Ist ein mehrstufiges, globales Netz im Parallelrechner vorhanden, in dem die Transferzeiten unabhängig von den Transferdistanzen sind, verhalten sich die verschiedenen Abbildungen bei Betrachtung des Kommunikationsaufwandes gleich.

5.3.2.3 Speicherbedarf

Da in den mittel- und grobgranularen Abbildungen alle Puffer eines bzw. mehrerer Koppelelemente auf einem PE implementiert werden müssen, während bei der feingranularen Abbildung nur jeweils zwei Puffer auf ein PE abgebildet werden (siehe Abbildung 5.1), benötigen die mittel- und grobgranularen Abbildungen einen viel höheren Speicherplatzbedarf pro PE als die feingranulare Abbildung. Dies kann ein großer Nachteil der mittel- und grobgranularen Abbildungen sein, da in vielen SIMD-Rechnern nur ein begrenzter Speicherplatz pro PE zur Verfügung steht [JuS95b]. So können manche Netzkonfigurationen mit den mittel- und grobgranularen Abbildungen nicht mehr auf einem SIMD-Rechner implementiert werden.

5.3.2.4 Zusatzaufwand für Koppelelementoperationen

Wenn Netzfunktionen, die ganze Koppelelemente und nicht nur einzelne Zwischenleitungen und/oder Puffer berücksichtigen (z. B. die Berechnung der Anzahl der leeren Pufferplätze bei Koppelelementen mit Zentralpuffer), angewendet werden, führen mittel- und grobgranulare Netzabbildungen zu leistungsfähigeren Simulatoren. Bei diesen Abbildungen befinden sich Daten und Informationen eines gesamten Koppelelements in einem PE, so daß Koppelelementoperationen effizient durchgeführt werden können. Dagegen sind Koppelelementinformationen bei der feingranularen Abbildung auf mehrere PEs verteilt und müssen durch Kommunikation zwischen einzelnen PEs zusammengefaßt werden. Diese zusätzliche Kommunikation führt zu einer Leistungsminderung der feingranularen Abbildung.

Die Diskussion der Vor- und Nachteile der einzelnen Abbildungsstrategien ist in Abbildung 5.2 zusammengefaßt.

$N \times N$ Netz bestehend aus s Stufen mit $B \times B$ Koppelelementen, abgebildet auf einen SIMD-Rechner mit N_S Prozessoren (N ist ein Vielfaches von B)

	feingranular PE pro Link	Vor/ Nach	mittelgranular PE pro Koppel- element	Vor/ Nach	grobgranular PE pro U Kop- pel- elemente	Vor/ Nach
Anzahl der PEs pro Subsimulator	$N*(s+1)$ † groß		$\frac{N*s}{B}$ mittel		$\left\lceil \frac{N*s}{U*B} \right\rceil$ klein	
Ausführzeit einer Subsimulation	klein	+	mittel	±	groß	–
Anzahl der parallel arbeitenden Subsimulatoren	$\left\lfloor \frac{N_S}{N*(s+1)} \right\rfloor$ klein	–	$\left\lfloor \frac{N_S*B}{N*s} \right\rfloor$ mittel	±	$\left\lfloor \frac{N_S*B*U}{N*s} \right\rfloor$ groß	+
Kommunikationsaufwand	groß	–	mittel	–	klein	+
Speicherbedarf pro PE	klein	+	mittel	±	groß	–
Zusatzaufwand bei Koppelelement-operationen (z. B. bei Koppelelementen mit Zentralpuffer)	groß	–	klein	+	klein	+

† Die Netzein- und ausgänge müssen auch abgebildet werden–> $s+1$

Vor/Nach = Vorteil/Nachteil

Abbildung 5.2: *Vor- und Nachteile verschiedener Abbildungsgranularitäten.*

5.3.2.5 *Leistung von parallelen zeitgesteuerten Simulatoren*

Durch die gegenläufigen Verhalten der einzelnen Abbildungsstrategien (siehe Abbildung 5.2) läßt sich nicht feststellen, welche Granularität zu dem schnellsten Simulator führt. Deshalb wurden beispielhaft zwei Simulatoren implementiert und Leistungsmessungen vorgenommen [JuS96], um quantitative Vergleiche zu erhalten. Als Netztypen sind hierbei synchrone store-and-forward paketvermittelnde Delta-Netze gewählt worden, welche bis zu 1024 Quellen mit 1024 Senken verbinden. Diese Netze wurden auf den MasPar-Rechner MP-1 [Bla90] abgebildet, welcher 16K Prozessoren in SIMD-Architektur besitzt. Es sind die Link- und die Schalterabbildung implementiert worden; die Teilnetzabbildung konnte, bedingt durch den begrenzten Speicherplatz von 16Kbyte pro PE, nicht auf der MasPar implementiert werden.

Der Leistungsvergleich wurde anhand der Simulation eines fünfstufigen 1024×1024 Delta-Netzes, bestehend aus 256 4×4 Koppelelementen pro Stufe, durchgeführt. Messungen zeigen, daß ein Subsimulator der Linkabbildung pro Simulationsschritt 9.5 msec für eingangs- und ausgangsgepufferte Koppelelemente und 12 msec für Koppelelemente mit Zentralpuffer verbraucht. Ein Subsimulator der Schalterabbildung braucht 31 msec pro Simulationsschritt für alle Pufferstrategien. Hierbei benötigt ein Subsimulator der Linkabbildung 6144 PEs (zwei Subsimulatoren können parallel arbeiten), während 1280 PEs für einen Subsimulator der Schalterabbildung gebraucht werden (12 parallele Subsimulatoren können implementiert werden). So mitteln sich die Ausführungszeiten pro Simulationsschritt zu 4.75 msec für eingangs- und ausgangsgepufferte Koppelelemente und 6 msec für Koppelelemente mit Zentralpuffer bei der Linkabbildung, und 2.58 msec für alle Pufferstrategien bei der Schalterabbildung. Dieser Vergleich zeigt deutlich den Leistungsvorteil der Schalterabbildung, vor allem bei der Simulation von Koppelelementen mit Zentralpuffer. Die Schalterabbildung stößt allerdings an ihre Grenzen, wenn z. B. komplexere Koppelelemente mit mehreren parallelen Puffern pro Koppelelementein- und/oder Ausgang [JuS93, JuS94] simuliert werden sollen, da diese, abhängig von der Koppelelementgröße, durch den beschränkten Speicherplatz pro PE nicht mehr implementiert werden können. Messungen an einem äquivalenten seriellen Simulator auf einer SUN SPARC-10 Workstation zeigen eine Ausführungszeit pro Simulationsschritt von ca. 400 msec für alle Pufferstrategien. So kann in diesem Fall durch die parallele Simulation eine Leistungssteigerung um einen Faktor von bis zu 150 erreicht werden.

6 Leistungsverhalten, Modellierung und Optimierung von Delta-Netzen unter ungleichverteilten Datenverkehren

Ungleichverteilte Datenverkehre können in Verbindungsnetzen, die unter gleichverteilten Verkehren gute Leistungseigenschaften aufweisen, zu signifikanten Leistungseinbrüchen führen, die die Leistungsfähigkeit des Parallelrechnersystems erheblich herabsetzen kann. Da das Auftreten von ungleichverteilten Verkehren im Parallelrechner in vielen Fällen nicht vermieden werden kann, müssen die Auswirkungen und Effekte dieser Verkehre in Netzen untersucht und deren negativer Einfluß begrenzt werden, um Leistungseinbußen in Parallelrechnern zu vermeiden.

In diesem Kapitel werden zunächst beispielhaft die Auswirkungen von statischen und temporären Hot-Spot-Verkehren auf die Leistung von synchronen store-and-forward paketvermittelnden Delta-Netzen simulativ untersucht. Es werden hierbei Block-Netze betrachtet, bei denen also keine Daten innerhalb des Netzes verloren gehen können. Diese Untersuchungen resultieren dann in der Herleitung des Klassifizierungskonzepts der HOL-Blockierungen höherer Ordnung in Delta-Netzen in Abschnitt 6.3. Es wird gezeigt, daß innerhalb jeder Klasse die Netze gleicher Topologie eine vergleichbare Leistung bei vergleichbarem Hardwareaufwand unter ungleichverteilten Datenverkehren aufweisen, während die Leistung der Netze mit steigender HOL-Blockierungsordnung zunimmt.

In Abschnitt 6.4 wird dann eine Prioritätskoppelelementarchitektur mit verschiedenen Prioritätsmechanismen vorgestellt, mit der es möglich ist, die Sättigungsbaumeffekte unter ungleichverteilten Verkehren zu steuern. Hierbei kann durch die entsprechende Wahl eines Prioritätsparameters die Bildung von Sättigungsbäumen vollständig unterdrückt werden, so daß der negative Einfluß von ungleichverteilten Verkehren in diesen Netzen eliminiert werden kann.

Die vorgestellte Prioritätskoppelelementarchitektur kann auch in ATM-Netzen zur Verkehrssteuerung und Leistungsverbesserung eingesetzt werden. Dies wird im letzten Abschnitt dieses Kapitels anhand von ATM-Netzen unter MPEG-codierten Videoströmen gezeigt.

6.1 Effekte von ungleichverteilten Verkehren in Delta-Netzen

Wie in Kapitel 4 diskutiert, existieren ungleichverteilte Datenverkehre, unter denen sich Sättigungsbäume, bestehend aus gefüllten Koppelelementpuffern, in mehrstufigen paketvermittelnden indirekten Netzen aufbauen können. Diese Sättigungsbäume blockieren "heiße" und "kalte" Pakete und führen somit zu einer Leistungsminderung in den Netzen. Die Effekte dieser Bäume auf die Paketklassen werden in den nächsten Abschnitten beispielhaft anhand von statischen und temporären Hot-Spot-Verkehren in Delta-Netzen aufgezeigt. Hierbei wird ein 1024×1024 Netz

aufgebaut aus 2×2 ausgangsgepufferten Koppelelementen (siehe Abbildung 3.5) mit einer **Gesamtpufferkapazität** C von C=40 Paketen pro Koppelelement (FIFO-Pufferlänge von jeweils 20 Paketen) angenommen. Um die Effekte besser erklären zu können, werden im folgenden verschiedene Begriffe definiert, die in den Diskussionen der nächsten Abschnitte verwendet werden. Ein Link des Verbindungsnetzes ist ein **kalter Link**, wenn über ihn nur kalte Pakete transferiert werden. Über einen **heißen Link** werden heiße und kalte Pakete transferiert. Auch werden die kalten Pakete weiter in zwei Unterklassen eingeteilt: in die Tree- und die Nontree-Pakete. Hierbei gehören kalte Pakete zu der Klasse der **Tree-Pakete**, wenn sie die erste Netzstufe (die Stufe, an der die Quellen angeschlossen sind) auf einem heißen Ausgangslink verlassen. **Nontree-Pakete** verlassen die erste Netzstufe auf kalten Ausgangslinks. Dies ist in Abbildung 6.1 verdeutlicht, in dem ein $B \times B$ Koppelelement der ersten Netzstufe unter ungleichverteiltem Verkehr gezeigt ist. Es wird angenommen, daß die heißen Pakete des Verkehrsgemisches zu dem Ausgangslink A_0 der ersten Netzstufe geroutet werden. Kalte Pakete, die ebenfalls für den Ausgangslink A_0 bestimmt sind, sind somit Tree-Pakete, während kalte Pakete, die zu einem Ausgangslink $A_i \neq A_0$ geroutet werden, zur Klasse der Nontree-Pakete gehören.

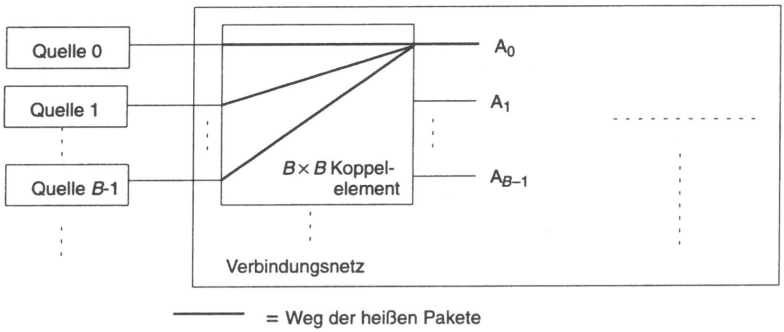


Abbildung 6.1: *$B \times B$ Koppelelement der ersten Netzstufe unter ungleichverteiltem Verkehr.*

6.1.1 Statisch ungleichverteilte Verkehre

Unter diesen Verkehrsszenarien können sich statische Sättigungsbäume in den Netzen aufbauen, die bis zu den Netzeingängen reichen und Pakete, die in das Netz gelangen wollen, behindern. Die Pakete, die nicht in das Netz gelangen können, müssen dann in den Quellen entweder verworfen oder in Quellpuffern zwischengespeichert werden. Da bei statischen Sättigungsbäumen die Anzahl der zu puffernden Pakete bei Quellen mit Quellpuffern mit der Zeit stetig wächst, können in diesem Fall keine aussagefähigen Netzleistungsparameter extrahiert werden. Deshalb

wird im folgenden angenommen, daß Pakete, die nicht in das Netz eingebracht werden können, in den Quellen verworfen werden.

Zur Verdeutlichung der Effekte von statisch ungleichverteilten Verkehren in Delta-Netzen wird nun die Leistung von einem 1024 × 1024 ausgangsgepufferten Netz unter statischem Hot-Spot-Verkehr untersucht. Ein wichtiger Leistungsparameter ist hierbei die zu erreichende Netzauslastung (siehe Abschnitt 5.1), die in Abbildung 6.2 in Abhängigkeit von der angebotenen Verkehrsrate des gleichverteilten Hintergrundverkehrs bei einer Hot-Spot-Rate von h=0.001 und h=0.01 gezeigt ist. Zum Leistungsvergleich ist die Auslastung des Netzes unter gleichverteiltem Verkehr (h=0) ebenfalls gezeigt. Wie in Abschnitt 4.1.2.1 erläutert, bildet sich ein Sättigungsbaum unter dem angenommenen Hot-Spot-Verkehr erst für Verkehrsraten $\lambda > \lambda_0$ mit $\lambda_0 = 0.495$ für h=0.001 und $\lambda_0 = 0.089$ für h=0.01 aus (siehe Gleichung (2) und Abbildung 6.2). Oberhalb dieser Rate existiert ein statischer Sättigungsbaum im Netz, der zu einer reduzierten Netzauslastung führt. Je größer der heiße Verkehrsanteil h am Gesamtverkehr ist, desto niedriger ist die maximal erreichbare Auslastung. Eine obere Schranke der Auslastung von Delta-Netzen unter statisch ungleichverteilten Verkehren wird in Abschnitt 6.3.6.1 analytisch hergeleitet.

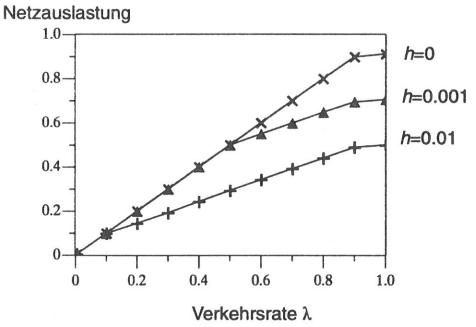


Abbildung 6.2: *Mittlere Netzauslastung vs. Verkehrsrate in einem 1024 × 1024 Netz (keine Quellpuffer) mit 2 × 2 ausgangsgepufferten Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr und unter gleichverteiltem Verkehr.*

In Abbildung 6.3 sind die Abhängigkeiten des relativen Durchsatzes (Abbildung 6.3a) und der Paketdurchlaufzeit (Abbildung 6.3b) aller Pakete von der Rate des gleichverteilten Hintergrundverkehrs gezeigt. Mit steigendem Hot-Spot-Verkehrsanteil sowie mit steigender Verkehrsrate nimmt der Paketdurchsatz ab. Die Paketdurchlaufzeit zeigt ein komplexeres Verhalten auf. Dies ist auf den unterschiedlichen Einfluß des Sättigungsbaums auf die Tree- und Nontree-Pakete zurückzuführen. Zum besseren Verständnis ist in Abbildung 6.4 beispielhaft das Durchsatz- und Durchlaufzeitverhalten dieser beiden Paketklassen bei einem statischen Hot-Spot-Verkehr mit h=0.01 gezeigt. Es werden demnach nur die Tree-Pakete vom Sättigungsbaum blockiert,

während die Nontree-Pakete nicht beeinflußt werden (dieses Verhalten wird in Abschnitt 6.3.2 näher untersucht). Bei Verkehrsraten, die etwas höher als λ_0 sind, können noch relativ viele Tree-Pakete in das Netz gelangen, die jedoch eine große Durchlaufzeit im Netz erhalten. Dies führt zu einer relativ hohen mittleren Durchlaufzeit aller Pakete. Wird die Verkehrsrate vergrößert, so nimmt die Zahl der Tree-Pakete mit deren hoher Durchlaufzeit ab, so daß die mittlere Durchlaufzeit sinkt. Bei hohen Raten erhalten auch die Nontree-Pakete eine höhere Durchlaufzeit, so daß die Gesamtdurchlaufzeit wieder steigt (siehe Abbildung 6.3b, h=0.01).

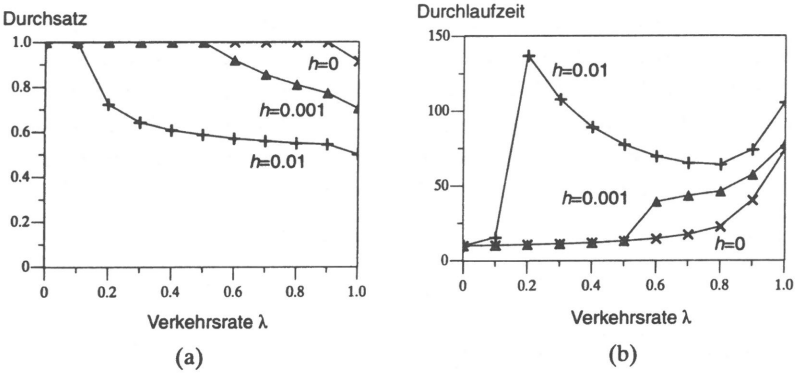


Abbildung 6.3: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit aller Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr und unter gleichverteiltem Verkehr (h=0).*

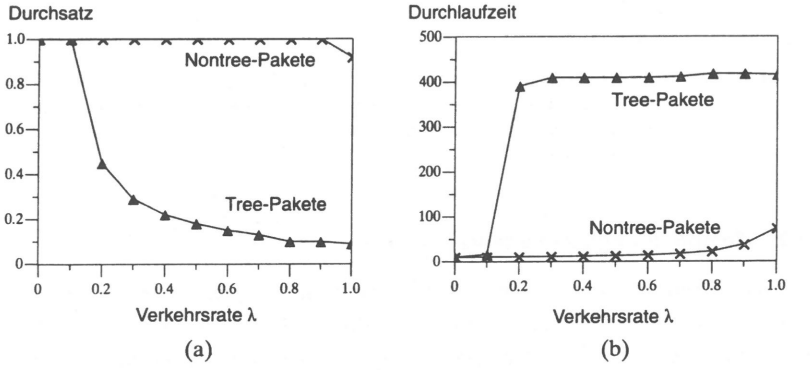


Abbildung 6.4: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit der Nontree- und Tree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr mit h=0.01.*

In Abbildung 6.5 ist die Abhängigkeit des relativen Durchsatzes und der Durchlaufzeit von der Gesamtpufferkapazität C pro Koppelelement unter statischen Hot-Spot-Verkehren bei einer Verkehrsrate von λ=0.5 gezeigt. Bei dem Hot-Spot-Verkehr mit h=0.001 baut sich ein Sättigungsbaum erst für Raten größer ca. 0.495 auf (siehe Gleichung (2)), so daß bei einer Rate von λ=0.5 und einer Pufferkapazität von C> 10 Paketen nur eine kaum wahrnehmbare Auswirkung des Sättigungsbaums auf die Netzleistung erkennbar sind. Ab einer Pufferkapazität von C>10 bei λ=0.5 ist der Paketdurchsatz für alle gezeigten Hot-Spot-Raten unabhängig von C, während die Paketdurchlaufzeit für h=0.01 mit steigendem C wächst. Bei h=0.01 und λ=0.5 (λ_0 = 0.089, siehe Gleichung (2)) existiert ein statischer Sättigungsbaum voller Puffer vom Hot-Spot bis zu den Netzeingängen, so daß der Durchsatz der Tree-Pakete unabhängig von der Größe der Koppelelementpuffer ist. Da alle Baumpuffer gefüllt sind, steigt die Durchlaufzeit der Tree-Pakete, die diese gefüllten Puffer durchqueren müssen, mit steigender Puffergröße. Für eine Gesamtpufferkapazität von C<10 bei λ=0.5 treten größere Konflikte unter den Nontree-Paketen im Netz auf, so daß der Paketdurchsatz sinkt und die Durchlaufzeit steigt.

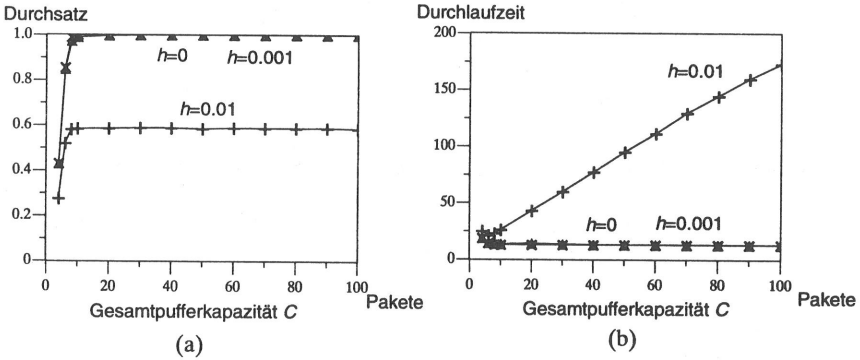


Abbildung 6.5: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit aller Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen, unter statischen Hot-Spot-Verkehren mit λ=0.5 und unter gleichverteiltem Verkehr (h=0).*

6.1.2 Temporär ungleichverteilte Verkehre

Unter diesen Verkehrsszenarien können sich temporäre Sättigungsbäume in den Netzen aufbauen, die bis zu den Netzeingängen reichen können und Pakete, die in das Netz gelangen wollen, temporär behindern. Da Sättigungsbäume sich nur temporär bilden, können Puffer in den Quellen verwendet werden, um Pakete, die nicht vom Netz akzeptiert werden, temporär zwischenzuspeichern, so daß keine Pakete verlorengehen. Es wird angenommen, daß jede Quelle einen unendlichen Puffer besitzt. Jeder dieser **physikalischen Puffer** ist in mehrere parallele **logische Queues** (Warteschlangen) unterteilt, wie in [Tze91a] vorgeschlagen wurde. Die Anzahl

der parallelen Queues richtet sich nach der Anzahl der Eingangspuffer pro Eingangsport in eingangsgepufferten Koppelelementen bzw. nach der Anzahl der Ausgangspuffer pro Koppelelement in ausgangsgepufferten Elementen die im am Netz angeschlossenen Koppelelement vorhanden sind. Jede Queue puffert hierbei nur Pakete, die für einen bestimmten Eingangs- bzw. Ausgangspuffer dieses Koppelelements bestimmt sind. Im Fall von ausgangsgepufferten 2×2 Koppelelementen sind deshalb zwei parallele Queues pro Quelle vorgesehen (siehe Abbildung 6.6). Hierbei werden Pakete, die für den oberen Ausgangsport des an der Quelle angeschlossenen Koppelelements bestimmt sind, in der oberen logischen Queue gepuffert, während Pakete, die zum unteren Ausgangsport des Koppelelements geroutet werden, in der anderen Queue zwischengepuffert werden. Es ist eine zyklische Auswahl der logischen Queues in einer Quelle in jedem Netzzyklus angenommen; ist die gewählte Queue leer, so wird die andere Queue ausgewählt, um eine möglichst hohe Vermittlungsleistung zu erzielen.

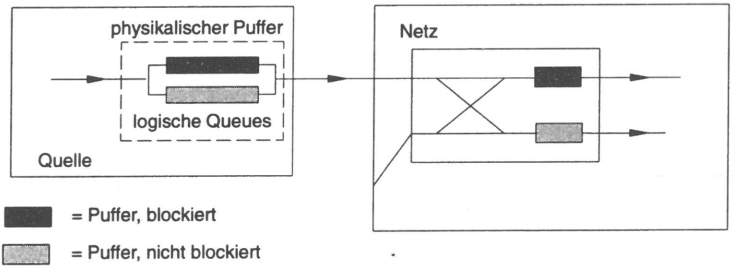


Abbildung 6.6: *Physikalische und logische Paketpuffer in einer Quelle.*

Man betrachte das System in Abbildung 6.6 unter ungleichverteilten Verkehren und nehme an, daß der obere Puffer des gezeigten Koppelelements blockiert (voll) sei. Da Pakete, die für diesen blockierten Puffer bestimmt sind, nur in der oberen logischen Queue der angeschlossenen Quelle zwischengepuffert werden, wird nur diese Queue blockiert. Pakete aus der unteren logischen Queue können weiterhin vermittelt werden, die bei der Verwendung nur einer Queue pro Quelle ebenfalls blockiert wären. Die parallelen logischen Queues vermeiden demnach Blockierungen von Nontree-Paketen durch Tree-Pakete in den Quellen wenn ein Sättigungsbaum im Netz besteht, der bis zu den Netzeingängen reicht, so daß eine höhere Leistung unter ungleichverteilten Verkehren erreicht werden kann als mit nur einer Queue pro Quelle.

Zur Verdeutlichung der Effekte von temporär ungleichverteilten Verkehren in Delta-Netzen wird nun beispielhaft die Paketdurchlaufzeit in einem 1024×1024 Netz aufgebaut aus 2×2 ausgangsgepufferten Koppelelementen (C=40) unter normalverteiltem Hot-Spot-Verkehr mit λ=0.5, μ=1000 und σ=50 untersucht. In Abbildung 6.7 ist der zeitliche Verlauf der Durchlaufzeiten der heißen (Abbildung 6.7a) und der kalten Pakete (Abbildung 6.7b) gezeigt. Hierbei ent-

spricht einem Simulationsresultat zum Zeitpunkt t dem Mittelwert der letzten 100 Resultate (von Zyklus t–99 bis Zyklus t) von 10 unabhängigen Simulationsläufen.

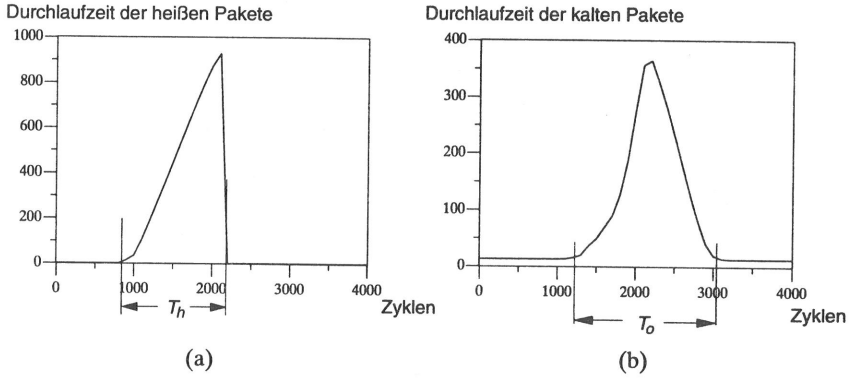


Abbildung 6.7: *Durchlaufzeiten der (a) heißen und (b) kalten Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen (C=40), unter normalverteiltem Hot-Spot-Verkehr mit λ=0.5, μ=1000 und σ=50.*

Wenn kein Sättigungsbaum vorhanden ist, erfahren die kalten Pakete eine Durchlaufzeit von ca. 13 Zyklen (ein 1024 × 1024 Netz aufgebaut aus 2 × 2 Koppelelementen besitzt 10 Stufen). Die temporär gefüllten Baum- und Quellpuffer während der Hot-Spot-Phase verursachen einen Anstieg der Durchlaufzeit der kalten Pakete auf bis zu 365 Zyklen. Nachdem fast alle heißen Pakete ihr Ziel erreicht haben (zum Zeitpunkt 2200 sind alle heißen Pakete an ihrem Ziel angelangt), beginnen sich die Baum- und Quellpuffer zu leeren, so daß die Durchlaufzeit der kalten Pakete wieder absinkt. Es können somit zwei sich überlappende Phasen während temporär ungleichverteilter Verkehrsszenarien definiert werden: 1) die **Hot-Spot-Phase** (das Zeitintervall der Länge T_h von der Generierung des ersten heißen Pakets bis alle heißen Pakete an ihrer Senke angekommen sind (Abbildung 6.7a)), und 2) die **Überlastphase** (das Zeitintervall der Länge T_o vom Zeitpunkt der Detektion bis zur vollständigen Auflösung des Sättigungsbaums (Abbildung 6.7b)). Der Zeitpunkt der Detektion des Sättigungsbaums wird durch den Netzzyklus festgelegt, an dem die Durchlaufzeit der kalten Pakete das erste mal um 2% höher liegt, als die Durchlaufzeit der Pakete unter gleichverteiltem Verkehr. Der Sättigungsbaum hat sich zu dem Zeitpunkt aufgelöst, an dem die Durchlaufzeit der kalten Pakete erstmalig weniger als 2% von der Paketdurchlaufzeit unter gleichverteiltem Verkehr abweicht.

Die Abhängigkeiten der einzelnen Phasenlängen und der Paketdurchlaufzeiten von der Verkehrsrate und der Intensität des Hot-Spots (Standardabweichung σ) sind in den Abbildungen 6.8 und 6.9 gezeigt. Bei einer sinkenden Hot-Spot-Intensität werden die heißen Pakete in einem größeren Zeitintervall erzeugt, so daß eine längere Hot-Spot-Phase resultiert (siehe Abbildung

6.8a). Der Sättigungsbaum wird dabei abgeschwächt, was zu einer etwas kürzeren Überlastphase (dies ist durch den großen Maßstab in Abbildung 6.8b kaum zu erkennen) und zu einer kleineren Durchlaufzeit der heißen, wie auch der kalten Pakete (siehe Abbildung 6.9) führt. Eine Erhöhung der Verkehrsrate verstärkt den Sättigungsbaum und erhöht somit die Phasenlängen und Paketdurchlaufzeiten. Für σ=0 hat die Rate im gezeigten Bereich keinen Einfluß auf die Hot-Spot-Phase und die Durchlaufzeit der heißen Pakete, da hierbei alle heißen Pakete gleichzeitig in das Netz gelangen und sich in den hinteren Netzstufen aufstauen, so daß kalte Pakete diese nur unwesentlich beeinflussen können.

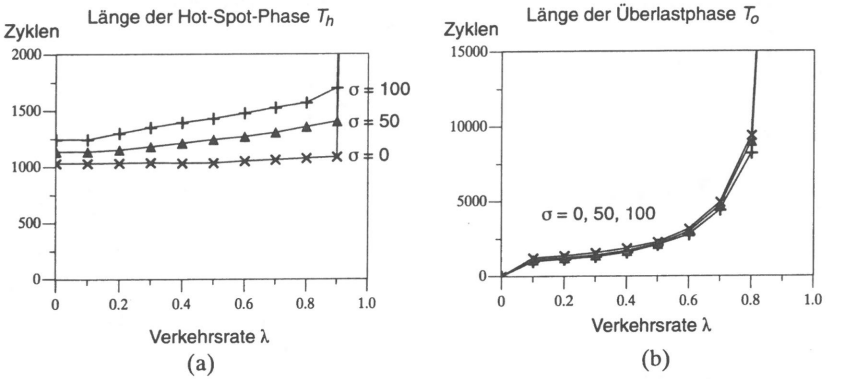


Abbildung 6.8: *Länge der (a) Hot-Spot- und (b) Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen (C=40), unter normalverteiltem Hot-Spot-Verkehr mit μ=1000.*

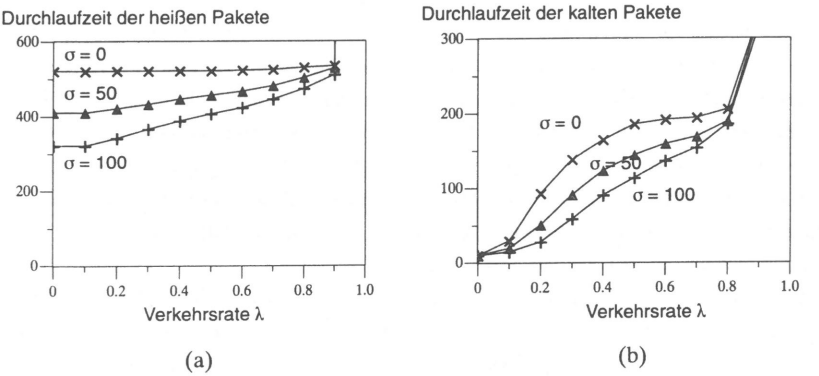


Abbildung 6.9: *Durchlaufzeit der (a) heißen und (b) kalten Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen (C=40), unter normalverteiltem Hot-Spot-Verkehr mit μ=1000.*

Die Länge der Überlastphase und die Durchlaufzeit der kalten Pakete in Abhängigkeit von der Gesamtpufferkapazität C pro Koppelelement sind in Abbildung 6.10 gezeigt. Der Sättigungsbaum während einer Hot-Spot-Phase bildet sich dadurch, daß heiße Pakete am Hot-Spot-Ausgang des Netzes aufgestaut werden. Dieser Effekt hängt nur unwesentlich von der Länge der Koppelelementpuffer ab, so daß nur eine geringe Abhängigkeit der Länge der Hot-Spot-Phase und der Durchlaufzeit der heißen Pakete von der Pufferkapazität besteht (deshalb sind diese Abhängigkeiten hier nicht gezeigt).

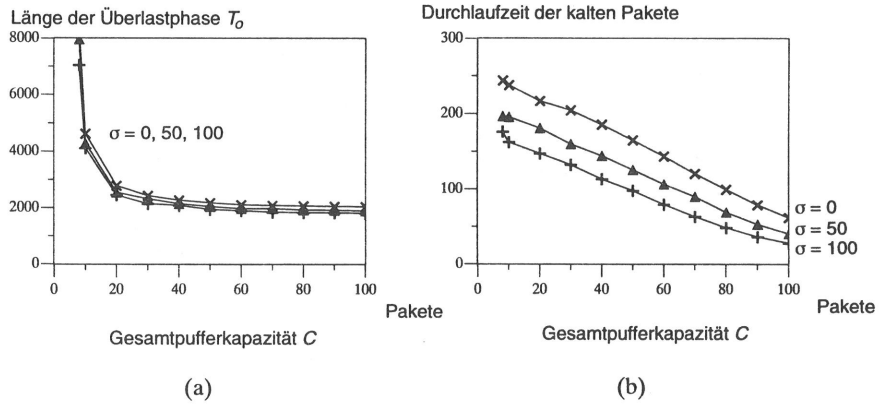


Abbildung 6.10: *(a) Länge der Überlastphase und (b) Durchlaufzeit der kalten Pakete in einem 1024 × 1024 Netz mit 2 × 2 ausgangsgepufferten Koppelelementen, unter normalverteiltem Hot-Spot-Verkehr mit λ=0.5 und μ=1000.*

Die maximale Länge des Sättigungsbaums (d.h. die maximale Anzahl von Pufferplätzen im Netz und in den Quellen, die der Baum vom Hot-Spot aus gerechnet einnimmt) ist primär von der Verkehrsrate λ und der Standardabweichung σ der heißen Pakete abhängig. Die Pufferkapazität C bestimmt im wesentlichen, über wieviele Netzstufen sich der Baum maximal im Netz ausbreitet, bzw. wieweit der Baum in die Quellpuffer reicht, wenn er die Netzeingänge erreicht. Je kürzer die Koppelelementpuffer sind, desto weiter wird der Baum sich im Netz bzw. in den Quellpuffern erstrecken, während sich der Baum bei einer Vergrößerung der Puffer immer weiter in das Netz zurückzieht. Dieser Effekt hat primär Auswirkungen auf die Länge der Überlastphase und die Durchlaufzeit der kalten Pakete. Je weniger Stufen der Baum im Netz belegt, desto mehr Tree-Pakete können durch das Netz transferiert werden, ohne daß sie auf den Baum stoßen und verzögert werden. Deshalb verkleinert sich die Durchlaufzeit der kalten Pakete bei einer steigenden Pufferkapazität (siehe Abbildung 6.10b). Die Länge der Überlastphase wird hiervon für C>20 nur in einem geringen Maße beeinflußt, während sie für C<20 stark anwächst (siehe Abbildung 6.10a), da der Baum dann bis in die Quellpuffer reicht und alle Tree-Pakete blockiert.

Diese Untersuchungen verdeutlichen, daß zusätzliche Leistungsmerkmale in indirekten Netzen unter temporär ungleichverteilten Datenverkehren, wie die Längen der Hot-Spot- und Überlastphasen, untersucht werden müssen, um die Dynamik dieser Verkehre und deren transiente Auswirkungen im Netz mit zu berücksichtigen.

6.2 Maßnahmen zur Verminderung von Sättigungsbaumeffekten in Delta-Netzen unter ungleichverteilten Verkehren

In der Literatur sind verschiedene Konzepte zur Verminderung von Sättigungsbaumeffekten in Verbindungsnetzen unter ungleichverteilten Datenverkehren vorgestellt worden. Diese Konzepte lassen sich in drei Klassen einteilen: 1) kombinierende Netze, 2) Flußkontrollmechanismen und 3) erweiterte Koppelelementarchitekturen, die in den folgenden Abschnitten kurz beschrieben werden.

6.2.1 Kombinierende Netze

Um die negativen Auswirkungen von Synchronisationsverkehren, die z. B. durch die Benutzung von Synchronisationsvariablen entstehen, auf die Netzleistung zu verringern, wurden im NYU-Ultracomputer [EdG85, GoG83] und im IBM RP3 Parallelrechner [BrM85, PfB85] kombinierende Netze eingeführt. Das Konzept des Kombinierens von Datenpaketen wird nun anhand des Modells eines kombinerenden 2×2 Koppelelements [PfN85], das in Abbildung 6.11 dargestellt ist, aufgezeigt. Hierbei wird eine Prozessor-zu-Speicher-Parallelrechnerorganisation angenommen.

Leseanforderungen an die Speicher, die als Nachrichtenpakete von den Prozessoren zu den Speichern geschickt werden, werden in den Eingangslatches eines Netzkoppelelements gepuffert und über einen 2×2 Crossbar auf die beiden Ausgangspuffer verteilt. Befindet sich in einem Ausgangspuffer bereits ein Leserequest, welcher zur gleichen Speicheradresse geroutet werden soll, wie der eintreffende Leserequest, so wird es mit dem eintreffenden Request zu einer Lesenachricht kombiniert. Diese Zusammenführung wird im Wartepuffer festgehalten. Wenn die Antwort auf die kombinierte Nachricht vom Speicher wieder in dem Koppelelement eintrifft, erkennt das Koppelelement durch Abfrage des Wartepuffers, daß die Antwort zu zwei in diesem Element kombinierten Leserequests gehört. Die Antwort wird dann im Koppelelement dupliziert und an die beiden Prozessoren geschickt, deren Leserequest kombiniert wurden. Da in aufeinanderfolgenden Netzstufen kombinierte Nachrichten wiederum kombiniert werden können, produziert die Generierung von duplizierten Antworten einen dynamischen Multicast von Daten zu mehreren Prozessoren. Durch Kombination wird also die Anzahl der heißen Datenpakete im Netz unter Hot-Spot-Verkehren signifikant verringert, so daß die Bildung von Sättigungsbäumen erschwert wird. Das Kombinieren von Paketen funktioniert allerdings nur, wenn diese zur gleichen Speicheradresse gerichtet sind. So lassen sich nur bestimmte Hot-Spot-Verkehre, die

z. B. aus einer Prozessorsynchronisation entstehen, kombinieren, während Pakete von anderen Verkehrsarten überhaupt nicht kombiniert werden können. So ist es z. B. nicht möglich, Pakete von Bit-Reverse-Permutationsverkehren zu kombinieren, bei denen sich, wie bei den Hot-Spot-Verkehren, Sättigungsbäume in Delta-Netzen bilden können, die zu Netzleistungseinbrüchen führen.

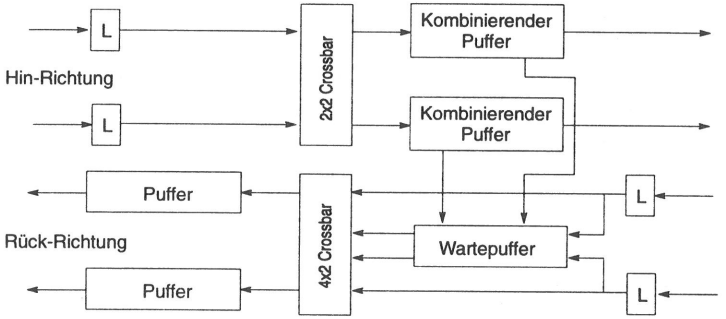


Abbildung 6.11: *Struktur eines kombinierenden 2×2 Koppelelements nach [PfN85].*

6.2.2 Flußkontrollmechanismen

In der Literatur sind verschiedene Flußkontrollmechanismen zur Steuerung von Sättigungsbaumeffekten in indirekten Verbindungsnetzen unter ungleichverteilten Datenverkehren vorgestellt worden. Diese können die Sättigungsbaumeffekte nur vermindern, nicht eliminieren, während die Netzleistung unter gleichverteilten Verkehren gegenüber Netzen ohne Flußkontrolle in vielen Fällen vermindert wird. Im folgenden werden einige Mechanismen kurz vorgestellt,

In [ScS90a] wird zur Verminderung von Sättigungsbaumeffekten eine Gegenkopplung in Parallelrechnersystemen mit Prozessor-zu-Speicher-Organisation vorgeschlagen, wie in Abbildung 6.12 gezeigt. Hierbei wird die Länge des Paketpuffers an jedem Speichermodul überwacht. Überschreitet diese Länge ein Limit T, so ist das Speichermodul heiß, anderenfalls kalt. Die Statusinformationen der einzelnen Speichermodule (kein Übergang, Übergang von kalt zu heiß, Übergang von heiß zu kalt) werden pro Modul den Prozessoren über einen $\log_2 N + 2$ bit breiten Gegenkopplungsbus ($\log_2 N$ bits für die Moduladresse und 2 Statusbits) mitgeteilt. Jeder Prozessor mißt die Zeitspanne, die ein Speichermodul kalt war und schickt Pakete zu diesem Modul in einem Zeitabstand, der umgekehrt proportional zu der Zeitspanne ist. Wenn ein Speichermodul öfter heiß wird und somit die Wahrscheinlichkeit groß ist, daß ein Sättigungsbaum entstehen kann, so senden die Prozessoren weniger Pakete zu diesem Modul und vermindern die Verkehrsrate zu diesem Speicher. Diese Methode kann die Bildung von Sättigungsbäumen unter

ungleichverteilten Verkehren einschränken und die Auswirkungen auf den kalten Verkehr vermindern. Da die Regelung jedoch immer eine gewisse Zeit braucht, um den Zustand des Netzes zu ändern, kann die temporäre Bildung von Sättigungsbäumen nicht verhindert werden. Auch kann die Regelung nicht zwischen gleich- und ungleichverteilten Verkehren unterscheiden, so daß in manchen Fällen die Netzleistung unter gleichverteilten Verkehren vermindert wird.

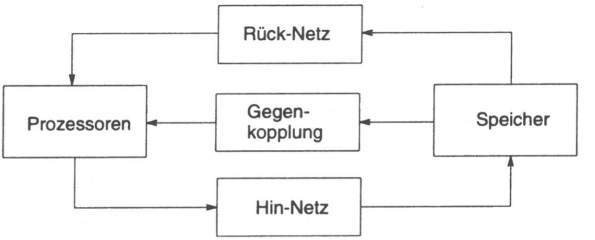


Abbildung 6.12: *Prozessor-zu-Speicher-Parallelrechnersystem mit Gegenkopplung.*

In [DiK89] ist eine Methode vorgestellt worden, die das Füllen der Puffer in den Koppelelementen eines Netzes bei ungleichverteilten Verkehren verhindert. Ein Paket wird hierbei aus einem Koppelelement einer Netzstufe zu einem Zielkoppelelement der nächsten Stufe nur weitergeleitet, wenn im Zielpuffer des Zielelements nicht mehr als T Pakete gepuffert sind, welche zur gleichen Senke wollen wie das zu sendende Paket. Kann ein Paket nicht weitervermittelt werden, so wird es an das Ende seines Puffers rotiert. Diese Methode stellt sicher, daß Puffer sich nicht mit heißen Paketen füllen und daß Pakete, die nicht weitergeleitet werden können, ihren Puffer nicht blockieren. Die Autoren zeigen, daß durch diese Methode eine Verbesserung der Netzleistung unter ungleichverteilten Verkehren erreicht werden kann. Allerdings wird die Netzleistung unter gleichverteilten Verkehren vermindert, da Pakete aufgehalten werden können, obwohl keine Unsymmetrien im Datenverkehr bestehen. Auch kann diese Methode nur durch einen großen Hardwareaufwand realisiert werden, da in jedem Netzzyklus die Paketpuffer durchsucht und die Adressen der gepufferten Pakete mit der Adresse des empfangenden Pakets verglichen werden müssen.

Da in Delta-Netzen nur jeweils ein Weg von einer Quelle zu einer Senke existiert und sich Wege Verbindungsleitungen und Puffer teilen, werden kalte Pakete zwangsweise durch heiße blockiert. In Mehrpfadnetzen, die mehrere Wege von einer Quelle zu einer Senke besitzen, treten deshalb weniger Blockierungen unter ungleichverteilten Verkehren auf. In [WaS95] sind Routingstrategien in Extra-Stage Cube Netzen (ESC) [AdS82] unter ungleichverteilten Verkehren untersucht worden, die zu einer Verminderung von Sättigungsbaumeffekten führen. Das ESC ist eine fehlertolerante Ausführung des Multistage Cube Netzes [SiN89] und besitzt zwei Wege von jeder Quelle zu jeder Senke, die sich weder Puffer noch Zwischenleitungen teilen. Wenn die

heißen Pakete auf vorbestimmten Wegen durch das Netz geroutet werden und die kalten Pakete auf Wegen transferiert werden, die durch keine oder nur wenige heiße Puffer führen, kann eine entscheidende Verbesserung des Netzleistungsverhaltens unter ungleichverteilten Verkehren erreicht werden, während die Netzleistung unter gleichverteilten Verkehren nicht vermindert wird.

6.2.3 Erweiterte Koppelelementarchitekturen

Eine Klasse von Methoden zur Leistungssteigerung in Netzen unter ungleichverteilten Datenverkehren zeichnet sich durch erweiterte Koppelelementarchitekturen aus, wobei durch Verwendung von parallelen logischen Puffern im Netz versucht wird, Datenpakete am Sättigungsbaum vorbeizuführen, so daß weniger Pakete vom Baum beeinflußt werden können. Diese Koppelelementarchitekturen können z. B. in [LiD88, PeL93, TaF92, Tze91b] gefunden werden. Exemplarisch sind in Abbildung 6.13 zwei 2×2 Koppelelementarchitekturen aus [Tze91b] und [PeL93] gezeigt, die sich durch zwei parallele Eingangspuffer pro Eingangsport (Abbildung 6.13a) bzw. durch jeweils zwei parallele Eingangslatches pro Eingangsport und zwei Ausgangspuffer pro Ausgangsport (Abbildung 6.13b) auszeichnen. Die Paketadressen bestimmen hierbei, in welchem der parallelen Puffer ein Paket an einem Koppelelementport zwischengepuffert wird. Diese Koppelelementarchitekturen werden im nächsten Kapitel ausführlich untersucht und nach Architektur- und Leistungsgesichtspunkten klassifiziert. Auch wird gezeigt, daß die Leistung der Elementarchitektur aus [PeL93] (Abbildung 6.13b) signifikant verbessert werden kann, wenn entweder die Eingangslatches weggelassen oder vier parallele Latches pro Eingangsport verwendet werden.

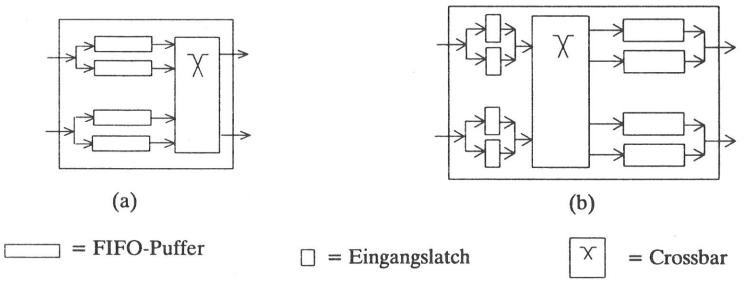


Abbildung 6.13: *Erweiterte Koppelelementarchitekturen vorgeschlagen in (a) [Tze91b] und (b) [PeL93].*

6.3 Netzklassifizierung mittels der HOL-Blockierungen höherer Ordnung

Zur Verminderung von Sättigungsbaumeffekten und somit zur Leistungssteigerung in mehrstufigen Delta-Netzen unter ungleichverteilten Verkehren wurden in der Literatur verschiedene

Erweiterungen von bestehenden Koppelelementarchitekturen vorgeschlagen (wie z. B. in [LaK88, LiD88, PeL93, TaF88, TaF92, Tze91a]). Diese Erweiterungen beinhalten unter anderem parallele Puffer pro Ausgangs- und/oder Eingangsport eines Koppelelements (siehe Abschnitt 6.2.3). Eine Klassifizierung und ein Leistungsvergleich solcher Koppelelementarchitekturen wurde jedoch nicht durchgeführt. Deshalb wird im folgenden das Konzept der Head-of-Line-Blockierungen höherer Ordnung (HOL-Blockierungen höherer Ordnung) eingeführt. Dieses Konzept basiert auf HOL-Blockierungseffekten innerhalb eines Netzes unter ungleichverteilten Verkehren, die in isolierten Koppelelementen nicht auftreten. Hierbei werden Netze nach der minimalen Länge von Wegen durch das Netz, die keine gemeinsamen Puffer besitzen, klassifiziert. Die HOL-Blockierungsordnung H einer Klasse ist hierbei proportional zu dieser minimalen Länge. Innerhalb jeder Klasse weisen Netze gleicher Topologie eine vergleichbare Leistung bei vergleichbarem Hardwareaufwand unter ungleichverteilten Datenverkehren auf, während die Leistung der Netze mit steigender HOL-Blockierungsordnung zunimmt.

Im nächsten Abschnitt werden zunächst konventionelle HOL-Blockierungen diskutiert, die in isolierten Koppelelementen auftreten. Werden Koppelelemente zu einem Netz zusammengefügt, so entstehen zusätzliche Blockierungseffekte, die in Abschnitt 6.3.2 beschrieben und durch HOL-Blockierungen höherer Ordnung modelliert werden. In Abschnitt 6.3.3 wird gezeigt, wie Delta-Netze in bezug auf deren HOL-Blockierungsordnung klassifiziert werden können; das Leistungsverhalten dieser Netze wird in den Abschnitten 6.3.5 und 6.3.6 diskutiert. Hierbei werden gleichverteilte Verkehre und Hot-Spot- und Bit-Reverse-Permutationsverkehre (statisch und temporär) simulativ und analytisch untersucht. Abschließend wird das Konzept der HOL-Blockierungen höherer Ordnung zur Leistungsoptimierung einer in der Literatur vorgestellten Koppelelementarchitektur verwendet.

6.3.1 Konventionelle HOL-Blockierungen

Um das Zustandekommen und die Beseitigung von konventioneller Head-of-Line-Blockierung in Koppelelementen zu diskutieren, wird ein Element betrachtet, dessen Ausgangsports nicht blockiert sind, so daß in jedem Netzzyklus jeweils ein Datenpaket über jeden Ausgangsport transferiert werden kann, wenn Pakete vorhanden sind. In solch einem Koppelelement tritt eine konventionelle HOL-Blockierung auf, wenn das Kopfpaket eines FIFO-Puffers innerhalb des Koppelelements durch einen Konflikt auf einer Verbindungsleitung im Koppelelement blockiert ist, und ein Paket, welches hinter dem Kopfpaket im FIFO-Puffer gepuffert ist, warten muß, obwohl es vermittelt werden könnte, wenn es am Kopf des FIFO-Puffers wäre.

Um das Auftreten von konventioneller HOL-Blockierung zu demonstrieren, betrachte man ein 2×2 eingangsgepuffertes Koppelelement, wie er in Abbildung 6.14 gezeigt ist. Man nehme an, daß die Kopfpakete beider FIFO-Eingangspuffer für Ausgangsport 0 bestimmt sind und das die Koppelelementsteuerlogik das Kopfpaket des Eingangspuffers QI_1 auswählt, um es zu Ausgangsport 0 zu transferieren. Das Kopfpaket des Eingangspuffers QI_0 muß in diesem Zyklus

warten, da die Verbindungsleitung am Ausgangsport 0 bereits belegt ist. Ein Paket, welches für Ausgangsport 1 bestimmt ist und hinter dem Kopfpaket des Eingangspuffers QI_0 gepuffert ist, ist auch blockiert, obwohl es zu Ausgangsport 1 transferiert werden könnte, wenn es am Kopf seines Puffers wäre. Dieser Effekt erklärt die relativ geringe Leistung von eingangsgepufferten Koppelelementen unter gleichverteiltem Verkehr [HlK88], wie in Abschnitt 3.1 erwähnt.

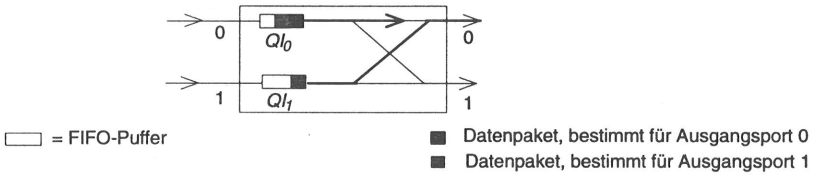


Abbildung 6.14: *Konventionelle HOL-Blockierung in einem eingangsgepufferten Koppelelement.*

Eine Methode, die konventionelle HOL-Blockierung zu eliminieren, ist die Verwendung von Ausgangspuffern im Koppelelement, wie in Abbildung 6.15 dargestellt. In diesem Element kann jeder Ausgangspuffer bis zu zwei Pakete gleichzeitig empfangen, vorausgesetzt, es sind mindestens zwei Plätze im Puffer frei. So können bis zu zwei Pakete gleichzeitig durch das Koppelelement transferiert werden, unabhängig von auftretenden Leitungskonflikten innerhalb des Elements. Somit existieren keine blockierten Pakete im Koppelelement, so daß konventionelle HOL-Blockierungen nicht auftreten können, solange mindestens jeweils zwei freie Plätze in den Puffern existieren und die Ausgangsports extern nicht blockiert sind. Diese Diskussion zeigt, daß konventionelle HOL-Blockierungen in isolierten Koppelelementen (ohne Interaktionen mit anderen Koppelelementen) auftreten können und daß eine Eliminierung diese Blockierungen durch eine geeignete Koppelelementarchitektur möglich ist.

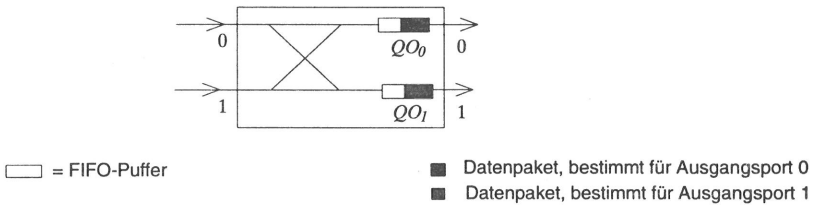


Abbildung 6.15: *Ausgangsgepuffertes Koppelelement, frei von konventioneller HOL-Blockierung.*

6.3.2 HOL-Blockierungen höherer Ordnung

Wenn Koppelelemente untereinander in einem Verbindungsnetz verbunden sind, können zusätzliche HOL-Blockierungseffekte in Koppelelementen, die frei von konventionellen HOL-Blokkierungen sind, auftreten. Diese zusätzlichen HOL-Blockierungseffekte werden **HOL-Blockierungen höherer Ordnung** (**HOL_k-Blockierungen**) genannt [JuS94], und Verbindungsnetze können mittels ihrer HOL-Blockierungsordnung klassifiziert werden. In diesem Abschnitt werden die HOL-Blockierungen höherer Ordnung definiert. Hierzu werden zuerst Begriffe definiert, die in der Diskussion benötigt werden. Ein FIFO-Puffer, der sein Kopfpaket während eines Netzzyklus nicht weitervermitteln kann, ist **blockiert**. In einem blockierten Puffer sind alle Pakete blockiert; die Pakete verbleiben im Puffer in unveränderter Reihenfolge. Der Ausgangsport eines blockierten Puffers bleibt während der Blockade ungenutzt. Weiterhin betrachte man zwei willkürlich ausgewählte Wege P_1 und P_2 durch das Netz. Wird ein Paket auf Weg P_1 und ein Paket auf Weg P_2 durch einen gemeinsamen Puffer Q transferiert, dann **teilen** sich die Wege P_1 und P_2 diesen Puffer Q.

Zur Einführung in die HOL-Blockierungen höherer Ordnung betrachte man zwei miteinander verbundene Koppelelemente in den Stufen s–1 und s–2 eines mehrstufigen indirekten Netzes. Wie in Abbildung 6.16 dargestellt, werden drei verschiedene Koppelelementarchitekturen angenommen (um die folgende Diskussion zu vereinfachen, sind nur die für die Diskussion relevanten Koppelelementkomponenten in der Abbildung gezeigt). Weiterhin wird angenommen, daß im Koppelelement b_2 des jeweiligen Netzes sich ein voller Puffer befindet, welcher einen Puffer in Koppelelement b_1 blockiert (siehe Abbildung 6.16). Im Netz in Abbildung 6.16(a) werden alle Pakete, die am Eingangsport 0 ankommen, im gleichen Puffer abgelegt. Demnach teilen sich alle Wege, die von der Quelle Q zu allen gezeigten Ausgangsports führen, den blockierten Puffer in Koppelelement b_1. So müssen alle Pakete, die in der Quelle Q generiert werden, den blockierten Puffer passieren, so daß diese Pakete ebenfalls blockiert werden. Dieser Blockierungseffekt tritt unabhängig von konventionellen Blockierungen auf. Netze, die durch diesen Blockierungseffekt charakterisiert sind, haben die HOL_1-Blockierungsordnung und werden HOL_1-Netze genannt.

Nun betrachte man das Netz in Abbildung 6.16(b). Pakete, die für verschiedene Ausgangsports eines Koppelelements bestimmt sind, werden hier in verschiedenen Puffern zwischengepuffert. Deshalb teilen sich nur die Wege, die von der Quelle Q zu den Ausgangsports von Koppelelement b_2 führen, den blockierten Puffer in Koppelelement b_1. So werden nur Pakete auf diesen Wegen blockiert, während Pakete auf dem Weg von der Quelle Q zu dem Ausgangsport 2 von Koppelelement b_1 keine Blockierung erfahren. Netze, die durch diesen Blockierungseffekt charakterisiert sind, haben die HOL_2-Blockierungsordnung und werden HOL_2-Netze genannt.

Als letztes betrachte man das Netz in Abbildung 6.16(c). In diesem Netz werden Pakete, die zu verschiedenen Ausgangsports in den Stufen s–1 und s–2 transferiert werden sollen, in getrennten Puffern zwischengepuffert. Die Wege von Quelle Q zu den Ausgangsports 2 und 4 teilen sich

nicht den blockierten Puffer, so daß nur Pakete, die für Ausgangsport 3 des Koppelelements b_2 bestimmt sind, blockiert werden. Netze, die durch diesen Blockierungseffekt charakterisiert sind, haben die HOL_3-Blockierungsordnung und werden HOL_3-Netze genannt.

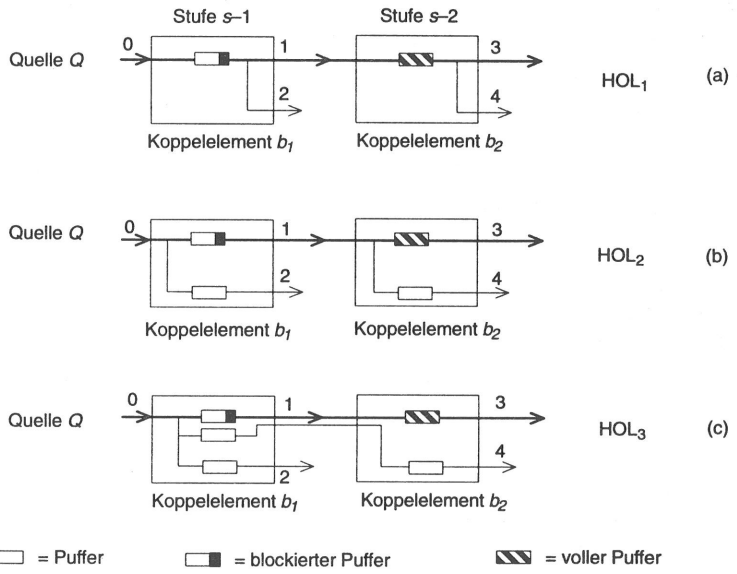


Abbildung 6.16: *Teile von Netzen mit unterschiedlicher HOL-Blockierungsordnung.*

Die Diskussion zeigt, daß die maximale Länge von Wegen in einem Netz, die sich blockierte Puffer teilen, die Blockierungseigenschaften des Netzes beeinflußt. Diese Eigenschaft wird nun in Definition 1 formuliert.

Definition 1: PS_1 sei die Menge von Wegsegmenten der Länge k (k>0) in einem mehrstufigen Netz, die an einem beliebigen Eingangsport der Netzstufe j beginnen und am Eingangsport l_1 der Stufe j–k enden (k–1 ≤ $j < s, 0 \leq l_1 < N$). PS_2 sei die Menge von Wegsegmenten der Länge k in einem mehrstufigen Netz, die an einem beliebigen Eingangsport der Netzstufe j beginnen und am Eingangsport $l_2 \neq l_1$ der Stufe j–k enden. Ein Netz weist die **HOL_k-Blockierungseigenschaft** auf, wenn ein Wegsegment $p_1 \in PS_1$ mit einem blockierten Puffer BP existiert, so daß durch den blockierten Puffer BP Pakete blockiert werden, die sich auf einem Wegsegment $p_2 \in PS_2$ befinden.

Untersucht man z. B. die Netze in Abbildung 6.16(b) und 6.16(c) unter Berücksichtigung von Definition 1, so sieht man, daß alle Wege der Länge 1 in Abbildung 6.16(b) getrennte Puffer benutzen, so daß ein blockierter Puffer keinen Einfluß auf andere Wege der Länge 1 hat. Andererseits teilen sich Wege der Länge 2 Puffer (z. B. der Weg von Port 0 zu Port 3 und der

Weg von Port 0 zu 4), so daß, wenn diese Puffer blockiert sind, Pakete, die sich auf den Wegen der Länge 2 befinden, blockiert werden. Nach Definition 1 weist dieses Netz demnach die HOL_2-Blockierungseigenschaft auf. In dem Netz in Abbildung 6.16(c) benutzen alle Wege der Länge 1 und 2 getrennte Puffer, während längere Wege sich Puffer teilen. Pakete auf diesen längeren Wegen werden dann blockiert, wenn die den Wegen gemeinsamen Puffer blockiert sind. Dieses Netz weist deshalb nach Definition 1 die HOL_3-Blockierungseigenschaft auf.

Im folgenden werden mehrstufige Delta-Netze bestehend aus s Netzstufen mit $B \times B$ Koppelelementen angenommen, die genau einen Weg von jeder Quelle zu jeder Senke aufweisen. In Definition 1 wurden Pufferblockierungen, die vom Verkehr abhängen, benutzt, um Netze zu klassifizieren. Um ein Klassifizierungskonzept zu erhalten, das unabhängig vom Verkehr ist, muß ein topologisches Kriterium gefunden werden, welches nicht von Pufferblockierungen abhängt. Dies ist in Theorem 1 formuliert:

Theorem 1: PS_1 sei die Menge von Wegsegmenten der Länge k (k>0) in einem mehrstufigen Netz, die an einem beliebigen Eingangsport der Netzstufe j beginnen und am Eingangsport l_1 der Stufe j–k enden (k–1 ≤ $j < s, 0 \leq l_1 < N$). PS_2 sei die Menge von Wegsegmenten der Länge k in einem mehrstufigen Netz, die an einem beliebigen Eingangsport der Netzstufe j beginnen und am Eingangsport $l_2 \neq l_1$ der Stufe j–k enden. Ein Netz weist dann und nur dann die **HOL_k-Blockierungseigenschaft** auf, wenn ein Wegsegment $p_1 \in PS_1$ sich mit einem Wegsegment $p_2 \in PS_2$ Puffer teilt.

Beweis. Dann: Es wird angenommen, daß die Wegsegmente p_1 und p_2 sich Puffer teilen. Pakete, die auf p_1 und p_2 durch das Netz gelangen, gelangen auch durch die von den Wegen geteilten Puffer. In Definition 1 wird ein blockierter Puffer auf p_1 betrachtet. Ohne Beschränkung der Allgemeinheit sei angenommen, daß einer der von p_1 und p_2 geteilten Puffer blockiert ist. Da Pakete, die auf p_1 und p_2 durch das Netz gelangen, auch durch den blockierten Puffer gelangen müssen, werden die Pakete von diesem Puffer blockiert. Nach Definition 1 ist dieses Netz demnach HOL_k-blockierend.

Nur dann: Es wird angenommen, daß die Wegsegmente p_1 und p_2 nur getrennte Puffer benutzen. Ohne Beschränkung der Allgemeinheit sei angenommen, daß ein Puffer des Weges p_1 blockiert ist und daß somit der Ausgangsport dieses Puffers ungenutzt ist. Der Ausgangsport dieses blokkierten Puffers kann deshalb Pakete auf p_2, die diesen Ausgangsport benutzen könnten, nicht beeinflussen oder blockieren. Auch folgt aus der Annahme, daß p_1 und p_2 nur getrennte Puffer benutzen, daß der blockierte Puffer auf Weg p_1 Puffer auf Weg p_2 nicht beeinflussen oder blokkieren kann. So wird ein blockierter Puffer auf dem Weg p_1 niemals Pakete blockieren, die sich auf dem Weg p_2 befinden, so daß dieses Netz nach Definition 1 nicht HOL_k-blockierend sein kann. □

Theorem 1 besagt, daß, wenn ein Netz für ein bestimmtes $k=k_0$ HOL_k-blockierend ist, Wegsegmente $p_1 \in PS_1$ sich mit Wegsegmenten $p_2 \in PS_2$ Puffer teilen. Wenn diese Wegsegmente auf eine Länge von $k>k_0$ verlängert werden, teilen sie sich immer noch diese Puffer. Deshalb wird

ein Netz, welches für $k=k_0$ HOL_k-blockierend ist, auch für $k>k_0$ HOL_k-blockierend sein. Mittels dieser Charakteristik kann eine Netz-HOL-Blockierungsordnung H definiert werden:

Definition 2: Die **HOL-Blockierungsordnung** H ($0 \leq H < s+1$) eines Netzes ist gleich dem minimalen k, für welches das Netz HOL_k-blockierend ist. Diese Netze werden **HOL_H-Netze** genannt.

Um die Anwendbarkeit des HOL-Blockierungskonzepts zu veranschaulichen, werden nun drei Netze in bezug auf deren Hol-Blockierungsordnungen untersucht: Netze mit (1) eingangsgepufferten Koppelelementen, (2) ausgangsgepufferten Koppelelementen und (3) ausgangsgepufferten Koppelelementen mit jeweils zwei parallelen Puffern pro Ausgangsport. Zuerst betrachte man zwei 2×2 eingangsgepufferte Koppelelemente in den Stufen j und j–1 eines Delta-Netzes, wie in Abbildung 6.17 gezeigt. Es wird angenommen, daß der Eingangspuffer QI_1 des Koppelelements b_1 leer ist, so daß keine konventionelle HOL-Blockierung auftreten kann. Außerdem wird angenommen, daß der Eingangspuffer QI_0 des Koppelelements b_2 voll ist und dieser den Eingangspuffer QI_0 des Koppelelements b_1 blockiert (siehe Abbildung 6.17). Da die Wege der Länge 1 vom Eingangsport 0 zu Ausgangsport 2, und vom Eingangsport 0 zu Ausgangsport 3 sich einen Puffer teilen (den blockierten Eingangspuffer QI_0 des Koppelelements b_1), werden Pakete, die am Eingangsport 0 ankommen und für die Ausgangsports 2 oder 3 bestimmt sind, blockiert. Nach Definition 1 ist dieses Netz HOL_1-blockierend, so daß die HOL-Blockierungsordnung dieses Netzes 1 beträgt.

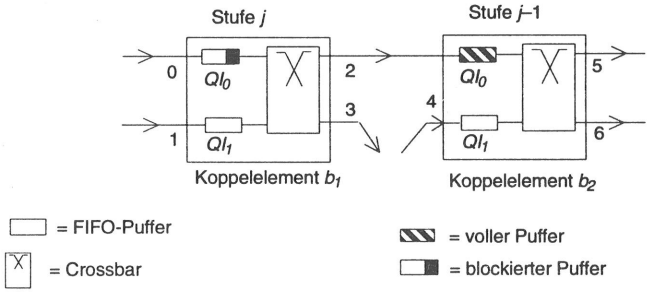


Abbildung 6.17: *Teil eines HOL_1-Netzes, bestehend aus eingangsgepufferten Koppelelementen.*

Als nächstes betrachte man zwei 2×2 ausgangsgepufferte Koppelelemente in den Stufen j und j–1 eines Delta-Netzes, wie in Abbildung 6.18 gezeigt. Es wird angenommen, daß der Puffer QO_0 des Koppelelements b_2 voll ist und dieser den Puffer QO_0 des Koppelelements b_1 blockiert. Da weder die Wege der Länge 1 von 0–>2 und von 0–>3, noch die Wege von 1–>2 und von 1–>3 sich Puffer teilen, ist dieses Netz nicht HOL_1-blockierend. Die Wege der Länge 2 von 0–>5 und von 0–>6 teilen sich jedoch einen Puffer (den blockierten Puffer QO_0 des Koppelelements b_1),

so daß Pakete, die an Port 0 eintreffen und für Port 5 oder 6 bestimmt sind, blockiert werden. Deshalb ist dieses Netz HOL_2-blockierend mit der HOL-Blockierungsordnung 2.

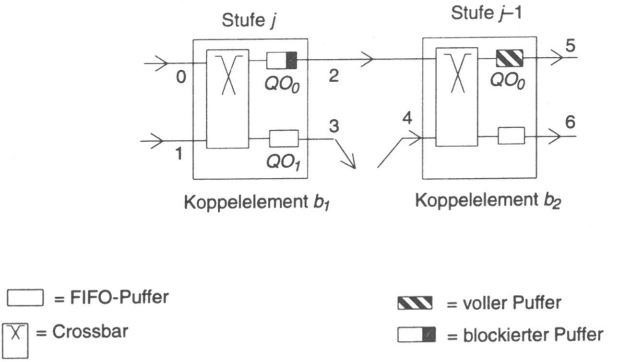


Abbildung 6.18: *Teil eines HOL_2-Netzes, bestehend aus ausgangsgepufferten Koppelelementen.*

Als letztes betrachte man 2×2 ausgangsgepufferte Koppelelemente mit jeweils zwei parallelen Puffern pro Ausgangsport, wie in Abbildung 6.19 dargestellt. Pakete, die an einem Koppelelementausgangsport eintreffen, werden im oberen Ausgangspuffer gepuffert (im Puffer $QO_{0,u}$ von Koppelelement b_1), wenn sie für den oberen Ausgangsport des Koppelelements in der nächsten Stufe bestimmt sind. Anderenfalls werden sie im unteren Puffer zwischengepuffert (im Puffer $QO_{0,d}$ von Koppelelement b_1). So werden zum Beispiel Pakete, die am Port 2 von Koppelelement b_1 ankommen und für Port 5 von Koppelelement b_2 bestimmt sind, im Puffer $QO_{0,u}$ von Koppelelement b_1 gepuffert, während Pakete, die für Port 6 von Koppelelement b_2 bestimmt sind, im Puffer $QO_{0,d}$ von Koppelelement b_1 zwischengepuffert werden. Analog dazu werden Pakete, die für Port 7 von Koppelelement b_3 bestimmt sind, im Puffer $QO_{0,u}$ von Koppelelement b_2 gepuffert, während Pakete, die für Port 8 von Koppelelement b_3 bestimmt sind, im Puffer $QO_{0,d}$ von Koppelelement b_2 gepuffert werden.

Nun wird angenommen, daß der Puffer $QO_{0,u}$ von Koppelelement b_1 durch einen vollen Puffer $QO_{0,u}$ von Koppelelement b_2 blockiert wird (siehe Abbildung 6.19). Dieses Netz ist nicht HOL_1-blockierend, da weder die Wege von 0–>2 und 0–>3, noch die Wege von 1–>2 und 1–>3 sich Puffer teilen. Die Wege von 0–>5 und 0–>6 und die Wege von 1–>5 und 1–>6 teilen sich ebenfalls keine Puffer, so daß dieses Netz auch nicht HOL_2-blockierend sein kann. Jedoch teilen sich die Wege von 0–>7 und 0–>8 einen Puffer (den blockierten Puffer $QO_{0,u}$ von Koppelelement b_1). Deshalb werden Pakete, die bei Port 0 ankommen und für die Ports 7 oder 8 bestimmt sind, durch den blockierten Puffer ebenfalls blockiert. Somit ist dieses Netz HOL_3-blockierend mit der HOL-Blockierungsordnung 3.

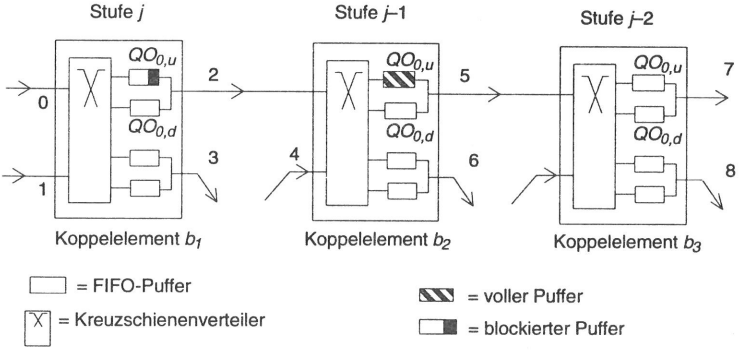


Abbildung 6.19: *Teil eines HOL_3-Netzes, bestehend aus Koppelelementen mit jeweils zwei parallelen Puffern pro Ausgangsport.*

6.3.3 Klassifikation von Delta-Netzen unter Verwendung des Konzepts der HOL-Blockierungen höherer Ordnung

In diesem Abschnitt wird gezeigt, wie Delta-Netze in bezug auf deren HOL-Blockierungsordnung klassifiziert werden können. Es werden Delta-Netze mit drei verschiedenen Koppelelementarchitekturen untersucht: Koppelelemente mit parallelen Eingangspuffern, Koppelelemente mit parallelen Ausgangspuffern und Koppelelemente mit parallelen Eingangslatches und parallelen Ausgangspuffern.

Als erstes werden Netze betrachtet, die aus eingangsgepufferten Koppelelementen aufgebaut sind. In diesen Koppelelementen ist ein Eingangspuffer direkt mit einem Eingangsport eines Koppelelements verbunden. Bedingt durch die Topologie mehrstufiger Delta-Netze führen $B^{j-(j-H+1)}=B^{H-1}$ verschiedene Wege von einem Eingangsport p_i eines Koppelelements in Stufe j ($H-2 \le j < s$) zu den Eingangslinks der Stufe $j-H+1$. In HOL_H-Netzen mit der HOL-Blockierungsordnung H dürfen sich diese Wege keine Puffer teilen. Deshalb müssen Pakete, die am Eingangsport p_i ankommen, in B^{H-1} verschiedenen Eingangspuffern an diesem Port gepuffert werden. Dies muß für alle Koppelelemente in einer Netzstufe j mit $H-2 \le j < s$ gelten, um ein HOL_H-Netz zu erhalten. Von einem Eingangsport p_i eines Koppelelements in Stufe j mit $0 \le j < H-2$ führen allerdings nur B^{j+1} verschiedene Wege zu den Ausgangsports der Stufe 0. Diese Wege dürfen sich ebenfalls keine Puffer in einem HOL_H-Netz teilen, so daß Pakete, die am Eingangsport p_i ankommen, in B^{j+1} verschiedenen Eingangspuffern an diesem Port zwischengepuffert werden müssen. So muß die Anzahl der parallelen Eingangspuffer N_{ib} jedes Eingangsports eines $B \times B$ Koppelelements in Stufe j

$$N_{ib} = \begin{cases} B^{H-1} & , \; H-2 \le j < s \\ B^{j+1} & , \; 0 \le j < H-2 \end{cases} \tag{10}$$

betragen, damit ein Delta-Netz aufgebaut aus eingangsgepufferten Koppelelementen zur Klasse der HOL_H-Netze mit der HOL-Blockierungsordnung H gehört. Ein Paket, welches an einem Eingangsport eines Koppelelements in Stufe j ankommt, wird dabei in dem Eingangspuffer gepuffert, welcher durch diejenigen $\lceil \log_2 (N_{ib}) \rceil$ Bits der Paketzieladresse selektiert wird, die den Weg des Pakets von Stufe j zu Stufe j–H+2 ($j \geq H$–2) oder zu Stufe 0 ($j < H$–2) bestimmen. Wird angenommen, daß ein s-stufiges Netz aus identischen Koppelelementen mit gleichem N_{ib} in allen Stufen aufgebaut ist, so kann die HOL-Blockierungsordnung dieses Netzes zu

$$H = \text{MIN}(\lfloor \log_B (N_{ib}) \rfloor + 1, s+1) \tag{11}$$

berechnet werden.

Als nächstes werden Netze betrachtet, die aus ausgangsgepufferten Koppelelementen aufgebaut sind. In diesen Koppelelementen ist ein Ausgangspuffer direkt mit einem Ausgangsport eines Elements verbunden. Von einem Ausgangsport p_o eines Koppelelements in Stufe j (H–2 $\leq j < s$), der direkt mit einem Eingangsport der Stufe j–1 verbunden ist, führen, bedingt durch die Delta-Netztopologie, $B^{(j-1)-(j-H+1)}=B^{H-2}$ verschiedene Wege zu den Eingangslinks der Stufe j–H+1. Diese Wege dürfen sich in HOL_H-Netzen keine Puffer teilen. Deshalb müssen Pakete, die an einem Ausgangsport p_o ankommen, in B^{H-2} verschiedenen Puffern an diesem Port gepuffert werden. Dies muß für alle Koppelelemente in einer Netzstufe j mit H–2 $\leq j < s$ gelten, um ein HOL_H-Netz zu erhalten. Von einem Ausgangsport p_o eines Koppelelements in Stufe j mit $0 \leq j < H$–2 führen nur B^j verschiedene Wege zu den Ausgangsports der Stufe 0. Diese Wege dürfen sich ebenfalls keine Puffer in einem HOL_H-Netz teilen, so daß Pakete, die am Ausgangsport p_o ankommen, in B^j verschiedenen Ausgangspuffern an diesem Port zwischengepuffert werden müssen. So muß die Anzahl der parallelen Ausgangspuffer N_{ob} jedes Ausgangsports eines $B \times B$ Koppelelements in Stufe j

$$N_{ob} = \begin{cases} B^{H-2} & , \; H-2 \leq j < s \\ B^j & , \; 0 \leq j < H-2 \end{cases} \tag{12}$$

betragen, damit ein Delta-Netz aufgebaut aus ausgangsgepufferten Koppelelementen zur Klasse der HOL_H-Netze mit der HOL-Blockierungsordnung H gehört. Ein Paket, welches an einem Ausgangsport eines Koppelelements in Stufe j ankommt, wird in dem Ausgangspuffer gepuffert, welcher durch diejenigen $\lceil \log_2 (N_{ob}) \rceil$ Bits der Paketzieladresse selektiert wird, die den Weg des Pakets von Stufe j zu Stufe j–H+2 ($j \geq H$–2) oder zu Stufe 0 ($j < H$–2) bestimmen. Wird angenommen, daß ein s-stufiges Netz aus identischen Koppelelementen mit gleichem N_{ob} in allen Stufen aufgebaut ist, so kann die HOL-Blockierungsordnung dieses Netzes zu

$$H = \text{MIN}(\lfloor \log_B (N_{ob}) \rfloor + 2, s+1) \tag{13}$$

berechnet werden. Zum Beispiel ist ein fünfstufiges Delta-Netz, welches aus ausgangsgepufferten Koppelelementen mit einem Ausgangspuffer pro Ausgangsport besteht (s=5, N_{ob}=1), ein HOL_2-Netz (H=MIN(2,6)=2), wie im vorherigen Abschnitt diskutiert worden ist.

Koppelelemente mit Zentralpuffer haben eine Blockierungsordnung H=1, da alle Wege der Länge 1 durch ein solches Koppelelement sich Puffer (den Zentralpuffer) teilen. Um ein HOL_H-Netz (H>1) mit Zentralpufferelementen zu erhalten, muß jedem Koppelelementausgang ein kleiner Teil des Puffers zugeteilt sein, auf den kein anderer Ausgang zugreifen kann. Dieser reservierte Teil muß wieder in logische Queues nach Gleichung (12) bzw. (13)unterteilt sein.

Nun betrachte man Netze, die aus Koppelelementen mit Eingangslatches und Ausgangspuffern aufgebaut sind. Eingangslatches in ausgangsgepufferten Koppelelementen vereinfachen die Datenflußsteuerung durch ein Koppelelement und zwischen Koppelelementen, welches zu einer besseren Auslastung der Koppelelementausgangspuffer führt (siehe Abschnitt 3.2.2). In mehrstufigen Delta-Netzen, die $B \times B$ Koppelelemente benutzen, hat die Eingangspufferkonfiguration keine Auswirkung auf die Benutzung der Ausgangspuffer von verschiedenen Wegen durch das Koppelelement, und vice versa. Damit ein Delta-Netz, das aus Koppelelementen mit Eingangslatches und Ausgangspuffern aufgebaut ist, zur Klasse der HOL_H-Netze mit der HOL-Blockierungsordnung H gehört, muß deshalb die Konfiguration der Eingangslatches der Koppelelemente der Gleichung (10) folgen, während die Konfiguration der Ausgangspuffer durch Gleichung (12) bestimmt wird. Wird angenommen, daß ein s-stufiges Netz aus identischen Koppelelementen mit gleichem N_{ib} und N_{ob} in allen Stufen aufgebaut ist, so kann die HOL-Blokkierungsordnung dieses Netzes zu

$$H = \text{MIN} (\lfloor \log_B (N_{ib}) \rfloor + 1, \lfloor \log_B (N_{ob}) \rfloor + 2 , s+1) \qquad (14)$$

berechnet werden.

Um die obige Diskussion zusammenzufassen, betrachte man zum Beispiel die Klasse der HOL_2-Netze, aufgebaut aus 2×2 Koppelelementen. Werden Eingangspuffer benutzt, so müssen, wie in Abbildung 6.20(a) gezeigt, jeweils zwei parallele Puffer pro Koppelelementeingangsport vorhanden sein (siehe Gleichung (10)), während konventionelle ausgangsgepufferte Koppelelemente, wie in Abbildung 6.20(b) gezeigt, ausreichend sind (siehe Gleichung (12)). Wenn Koppelelemente mit Eingangslatches und Ausgangspuffern verwendet werden, so müssen zwei Eingangslatches pro Eingangsport und ein Puffer pro Ausgangsport, wie in Abbildung 6.20(c) dargestellt, vorhanden sein (siehe Gleichungen (10) und (12)).

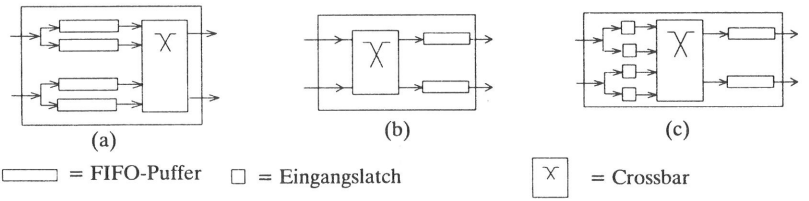


Abbildung 6.20: *HOL_2-Netze, bestehend aus 2×2 Koppelelementen mit (a) parallelen Eingangspuffern, (b) parallelen Ausgangspuffern und (c) parallelen Eingangslatches und Ausgangspuffern.*

Betrachtet man die Klasse der HOL_3-Netze, so müssen entweder eingangsgepufferte Koppelelemente mit vier parallelen Eingangspuffern pro Eingangsport (siehe Abbildung 6.21a), ausgangsgepufferte Koppelelemente mit zwei parallelen Puffern pro Ausgangsport (siehe Abbildung 6.21b) oder Koppelelemente mit vier parallelen Latches pro Eingangsport und zwei parallelen Puffern pro Ausgangsport (siehe Abbildung 6.21c) benutzt werden.

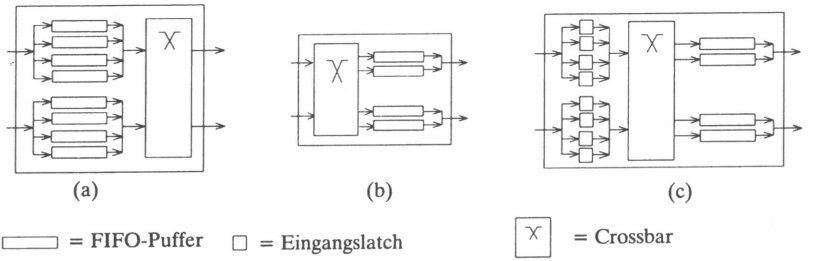


Abbildung 6.21: *HOL_3-Netze, bestehend aus 2×2 Koppelelementen mit (a) parallelen Eingangspuffern, (b) parallelen Ausgangspuffern und (c) parallelen Eingangslatches und Ausgangspuffern.*

Gleichungen (10) und (12) zeigen, daß bei einem s-stufigen Netz eine maximale HOL-Blockierungsordnung von $H_{max}=s+1$ erreicht werden kann. Dies entspricht einer getrennten Pufferung von allen N Wegen, die von jeweils einer Quelle ausgehen und zu allen N Senken führen.

6.3.4 VLSI-Realisierungsaspekte von Koppelelementen für HOL_H-Netze

Wie in Abschnitt 6.3.3 diskutiert, nimmt die Komplexität von HOL_H-Koppelelementen mit steigender HOL-Ordnung zu. Diese Komplexitätszunahme ist jedoch von der Koppelelementstruktur abhängig und wird in den nächsten Abschnitten näher beschrieben. Die für die folgende Diskussion wichtigen Annahmen für HOL_H-Koppelelemente sind hier noch einmal zusammengefaßt: 1) alle betrachteten Koppelelemente besitzen die gleiche Gesamtpufferkapazität C (Summe aus den Längen der einzelnen Eingangs- und Ausgangspuffer), und 2) in einem $B \times B$ Koppelelement können pro Zyklus maximal B Pakete (jeweils maximal ein Paket pro Elementeingangsport) durch das Koppelelement vermittelt werden.

6.3.4.1 Realisierungsaspekte der Koppelelementausgangspuffer

Man betrachte zum Beispiel ein 2×2 HOL_4-Koppelelement ohne Eingangslatches. Wie aus Gleichung (12) hervorgeht, benötigt dieses Koppelelement jeweils 4 parallele Ausgangspuffer pro Ausgangsport, wie in Abbildung 6.22a) dargestellt. Im Vergleich zu einem konventionellen

ausgangsgepufferten Koppelelement (siehe Abbildung 6.20b), welches einen 2×2 Crossbar benutzt, muß in dem HOL_4-Koppelelement mit Ausgangspuffern ein 2×8 Crossbar verwendet werden (siehe Abbildung 6.22a). Dies führt durch die Vervierfachung der Anzahl der Kreuzungspunkte im Crossbar zu einer immensen Steigerung des Hardwareaufwandes im Datenpfad (zusätzliche Schaltpunkte und zusätzlicher Verdrahtungsaufwand). Dieser zusätzliche Hardwareaufwand im Datenpfad entfällt jedoch vollständig, wenn nicht dedizierte Ausgangspuffer sondern ein Zentralpuffer im Koppelelement verwendet wird. Hierbei wird der Zentralpuffer in logische Queues aufgespalten, wie in Abbildung 6.22b) gezeigt. Diese logische Aufspaltung erfolgt allein durch eine Änderung des Kontrollpfades, während der Datenpfad des Koppelelements unverändert bleibt. Dies wird im folgenden näher beschrieben.

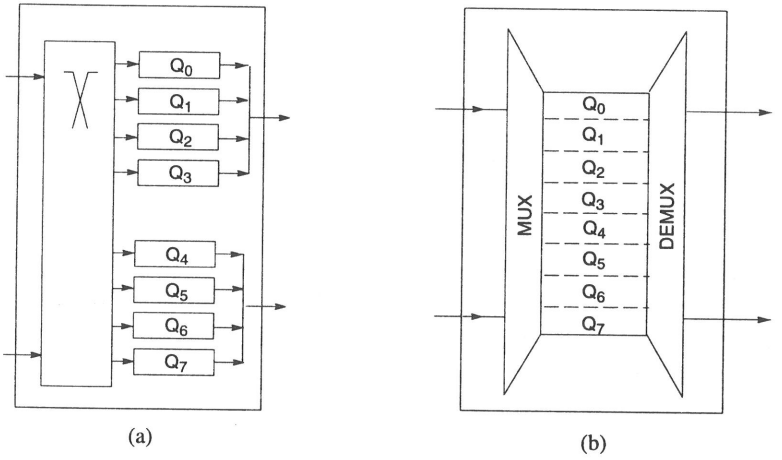


Abbildung 6.22: *2×2 HOL_4-Koppelelement mit (a) Ausgangspuffern und (b) Zentralpuffer.*

In einem konventionellen Koppelelement mit Zentralpuffer wird der Zentralpuffer in B logische Queues (eine pro Koppelelementausgangport) unterteilt, die bis zu jeweils Z Pakete (Größe des Zentralpuffers) puffern können. Die Verwaltung einer logischen Queue wird im Kontrollpfad des Koppelelements durch eine FIFO-Adreßqueue übernommen, in der die Zentralpufferadressen der Pakete, die für die zugehörige logische Queue bestimmt sind, in FIFO-Ordnung gepuffert sind (siehe Abbildung 3.16). Die Länge der Adreßqueue bestimmt, wieviele Pakete für den zugehörigen Koppelelementausgang im Zentralpuffer maximal gepuffert werden können. Wird die Größe dieser Adreßqueues auf jeweils Z/B beschränkt, so entsteht ein ausgangsgepuffertes Koppelelement (siehe Abschnitt 3.6.2). Werden nun nicht eine, sondern mehrere Adreßqueues pro Koppelelementausgang verwendet, so können parallele logische Queues pro Ausgangsport implementiert werden, die für HOL_H-Koppelelemente benötigt werden. Diese Aufspaltung einer Adreßqueue kann durch eine einfache Änderung der Ansteuerung der Queue erreicht werden.

Eine Möglichkeit der Ansteuerung einer FIFO-Adreßqueue durch die Verwendung von Schreib- und Lesezeigern wurde in Abschnitt 3.6.2 (siehe Abbildung 3.17) diskutiert. Um diese Adreßqueue in i kleinere Queues aufzuspalten, müssen nur die Schreib- und Lesezeiger in jeweils i kleinere Zeiger aufgespalten werden, wie in Abbildung 6.23 anhand einer in vier kleinere Queues unterteilten Adreßqueue gezeigt. Dabei bleibt die Puffermatrix mit den Ein- und Ausgangsbussen unverändert. Da nur maximal ein Paket pro Netzzyklus ein Koppelelementausgangsport verlassen kann, jedoch mehrere Ausgangsqueues pro Ausgangsport vorhanden sind, muß eine Arbitrierung in jedem Zyklus eine der parallelen Queues auswählen, die ein Paket über den zugehörigen Ausgangsport weiterleiten darf. Hierfür ist eine Queueselektion vorgesehen, die in jedem Zyklus einen der parallelen Lesezeiger aktiviert (siehe Abbildung 6.23). Die Auswahl kann zum Beispiel zyklisch durch einen Modulo-Zähler erfolgen. Zur Implementierung eines $B \times B$ HOL_H-Koppelelements (H>2) werden dann zusätzlich B mal (mod B^{H-2})-Zähler für die Queueselektionen benötigt, die aus insgesamt $\lceil B*\log_2 (B^{H-2})\rceil$ Flip-Flops aufgebaut sind. Dies entspricht einem zusätzlichen Aufwand von

$$T_Q = 24 * B * \lceil \log_2 (B^{H-2})\rceil \qquad (15)$$

Transistoren (ein D-Flip-Flop kann aus 24 Transistoren aufgebaut werden; siehe IMS-Standardzellenkatalog [BeH91]).

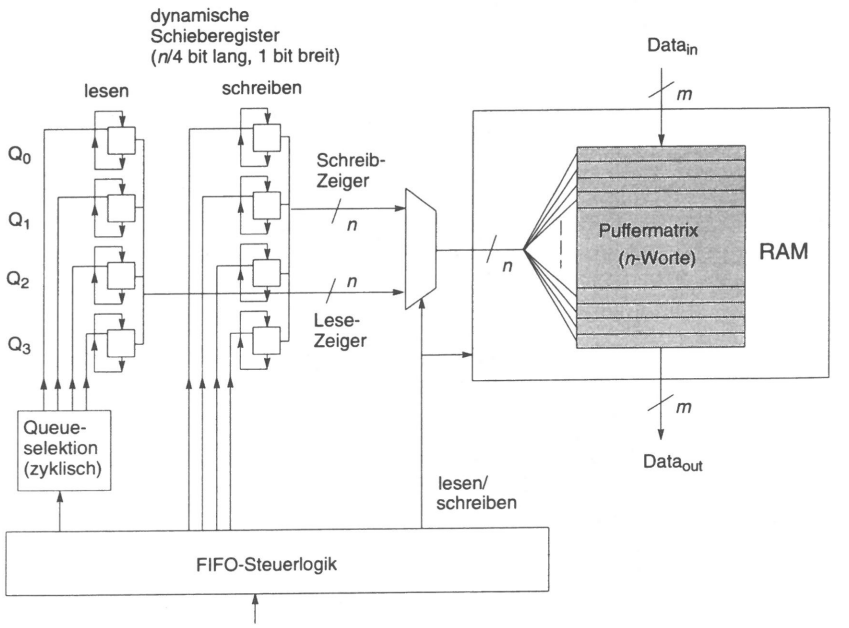


Abbildung 6.23: *Blockschaltbild von vier parallelen FIFO-Queues mit jeweils n/4 Worten, aufgebaut aus einem RAM mit Lese- und Schreibzeigeradressierung.*

Da der n-bit lange Zeiger, welcher eine Pufferzelle auswählt, nur eine 1 beinhalten darf, die parallelen Lese- bzw. Schreibzeiger jedoch zu diesem Wort zusammengefaßt werden, darf nur jeweils eins der parallelen Lese- und Schreibzeiger selektiert sein; die anderen Zeiger müssen Nullen an die Puffermatrix schicken. Dies kann zum Beispiel durch zusätzliche AND_2-Gatter (AND-Gatter mit zwei Eingängen) geschehen, die durch die Queueselektion bzw. durch die FIFO-Steuerlogik gesteuert werden. Eine mögliche Implementierung der Lesezeiger ist in Abbildung 6.24 dargestellt. Während einer Leseoperation wird einer der parallelen Lesezeiger durch eine logische 1 auf der zugehörigen Steuerleitung des Queueselektionsmoduls angesteuert; die anderen Steuerleitungen sind logisch 0. Gleichzeitig wird der Wert des ausgewählten Zeigers an den Multiplexer weitergegeben, während die restlichen Stellen des Gesamtlesezeigers durch die AND-Gatter auf 0 gesetzt sind. AND_2-Gatter wurden hier nur gewählt, um die Maskierung der Zeiger einfach zu verdeutlichen. Bei einer tatsächlichen Implementierung würden $NAND_2$-Gatter verwendet werden (NAND-Gatter können aus weniger Transistoren als AND-Gatter aufgebaut werden), wobei dann eine Zeile in der Puffermatrix mit einer logischen 0 ausgewählt wird.

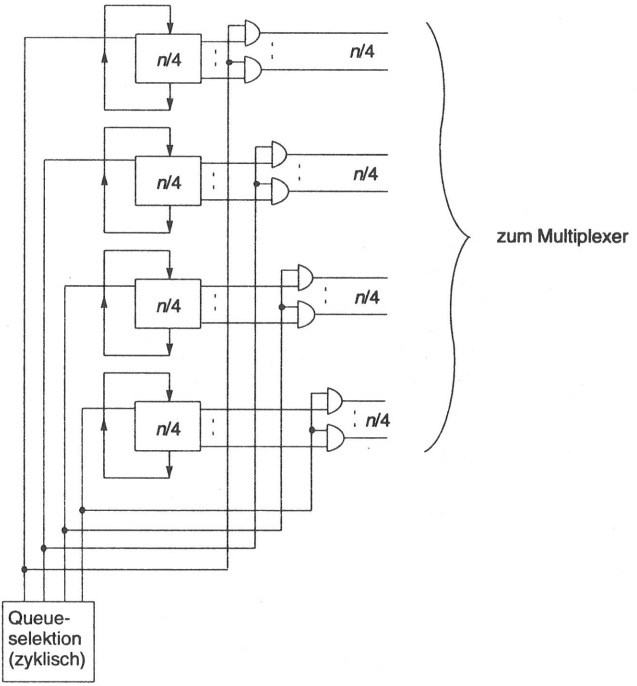


Abbildung 6.24: *Blockschaltbild der vier Lesezeiger einer Adreßqueue für vier parallele logische Ausgangspuffer.*

Zur Implementierung eines $B \times B$ HOL_H-Koppelelements mit einer Zentralpuffergröße von Z Paketen werden also zusätzlich $2*Z$ $NAND_2$-Gatter für die Lese- und Schreibzeiger benötigt. Dies entspricht einem zusätzlichen Aufwand von

$$T_N = 8*Z \tag{16}$$

Transistoren (ein $NAND_2$-Gatter kann aus 4 Transistoren aufgebaut werden; siehe IMS-Standardzellenkatalog [BeH91]).

Für die Schreibzeiger kann das gleiche Prinzip wie für die Lesezeiger verwendet werden, wobei die NAND-Gatter durch die Adressen der zu puffernden Pakete gesteuert wird. Hierzu ist ein 1-aus-B^{H-2} Decoder pro Koppelelementausgangsport nötig, der die logische Queue zum Puffern eines Pakets auswählt. In [Wen91] wurde gezeigt, wie allgemein ein 1-aus-X Decoder effizient aus $X^* \log_2 X$ Schaltern mit jeweils 3 Transistoren aufgebaut werden kann. So werden zur Ansteuerung der Schreibzeiger in einem $B \times B$ HOL_H-Koppelelement insgesamt

$$T_S = 3 * B^{H-1} * \lceil \log_2 (B^{H-2}) \rceil \tag{17}$$

benötigt.

Die Anzahl T der zusätzlichen Transistoren, die benötigt werden, um ein Zentralpufferkoppelelement mit einer Zentralpuffergröße von Z Paketen in ein HOL_H-Koppelelement (ohne Eingangslatches) zu überführen, kann somit zu

$$T = T_Q + T_N + T_S = (24 * B + 3* B^{H-1}) * \lceil \log_2 (B^{H-2}) \rceil + 8 * Z \tag{18}$$

berechnet werden. In Tabelle 6.1 ist die Anzahl der zusätzlichen Transistoren von $B \times B$ HOL_H-Koppelelement ohne Eingangslatches (Zentralpuffergröße Z=64 Pakete) aufgelistet (für $H \leq 2$ werden keine zusätzlichen Transistoren benötigt, da bei diesen Koppelelementen nur jeweils eine logische Queue pro Ausgangsport implementiert werden muß, welches einem konventionell ausgangsgepufferten Koppelelement entspricht).

B \ H	3	4	5	6
2	348	432	576	864
4	576	1440	5472	25632
8	1440	10656	112608	1182240

Tabelle 6.1: *Anzahl T der zusätzlichen Transistoren bei einem $B \times B$ HOL_H-Koppelelement ohne Eingangslatches (Zentralpuffergröße Z = 32 Pakete).*

Ein 4×4 Koppelelement mit einem konventionellen Zentralpuffer dieser Größe kann aus ca. 56000 Transistoren aufgebaut werden (siehe Abschnitt 3.6.2). Nimmt man einen Mehraufwand

der Hardware von maximal 10% (ca. 5600 Transistoren) zur Überführung des Koppelelements in ein HOL_H-Element in Kauf, so können 2×2 Koppelelemente bis zu einer HOL-Ordnung von H=8, 4×4 Koppelelemente bis zu einer HOL-Ordnung von H=5, und 8×8 Koppelelemente bis zu einer HOL-Ordnung von H=3 aufgebaut werden. Es wird hierbei vereinfachend angenommen, daß der Zentralpuffer die meisten Transistoren benötigt, so daß Koppelelemente mit beliebiger Größe B, aber mit gleicher Zentralpuffergröße Z ca. 56000 Transistoren benötigen; diese Annahme führt zu einer konservativen Abschätzung der Implementierbarkeit für HOL_H-Koppelelemente mit B>4, da größere Koppelelemente durch ihre steigende Komplexität mehr als 56000 Transistoren benötigen.

Wie in Abschnitt 6.3.5 gezeigt wird, müssen die einzelnen logischen Ausgangsqueues eine Mindestlänge von D_{min} >1 besitzten, damit das Koppelelement eine optimale Leistung unter gleichverteiltem Verkehr erzielt. Da ein $B \times B$ HOL_H-Koppelelement insgesamt B^{H-1} logische Ausgangsqueues besitzt, muß für die minimale Größe des Zentralpuffers Z_{min} gelten:

$$Z_{min} = D_{min} * B^{H-1}. \qquad (19)$$

Diese Mindestgröße des Zentralpuffers ist somit ebenfalls ein limitierender Faktor von HOL_H-Koppelelementen. Unter der Annahme, daß jede logische Ausgangsqueue eine Mindestlänge von vier Paketen haben muß, sind für verschiedene Koppelelementgrößen und HOL-Blokkierungsordnungen die Mindestgrößen des Zentralpuffers in Tabelle 6.2 aufgelistet. Es ist zu erkennen, daß für höhere HOL-Ordnungen sowie für größere Koppelelemente der Zentralpuffer sehr groß gewählt werden muß. Wird von einem vertretbaren Hardwareaufwand von 64 Zentralpufferplätzen pro Koppelelement ausgegangen, so können 2×2 Koppelelemente bis zu einer HOL-Ordnung von H=5, 4×4 Koppelelemente bis zu einer HOL-Ordnung von H=3, und 8×8 Koppelelemente bis zu einer HOL-Ordnung von H=2 aufgebaut werden. Ein Vergleich mit dem oben diskutierten Mehraufwand an Transistoren zeigt, daß die Mindestzentralpuffergröße der entscheidende Faktor bei der Abschätzung der Implementierbarkeit von HOL_H-Koppelelementen (ohne Eingangslatches) ist.

B \ H	1	2	3	4	5
2	4	8	16	32	64
4	4	16	64	256	1024
8	4	32	256	2048	16384

Tabelle 6.2: *Minimale Größe Z_{min} des Zentralpuffers bei einem $B \times B$ HOL_H-Koppelelement, wenn jede logische Queue eine Mindestlänge von D_{min}=4 Paketen hat.*

6.3.4.2 *Realisierungsaspekte der Koppelelementeingangslatches*

In diesem Abschnitt wird gezeigt, daß bei der Überführung eines Koppelelements ohne Eingangslatches in ein Koppelelement mit Eingangslatches keine zusätzliche Hardware benötigt wird, sondern Hardware eingespart werden kann. Da davon ausgegangen wird, daß in einem $B \times B$ Koppelelement pro Zyklus maximal B Pakete (jeweils maximal ein Paket pro Elementeingangsport) durch das Koppelelement vermittelt werden können, kann auch bei der Benutzung von mehreren parallelen Eingangslatches pro Eingangsport das Zentralpufferkonzept für die Ausgangsqueues verwendet werden (der Zentralpuffer muß in einem Netzzyklus maximal B Pakete puffern und B Pakete auslesen, unabhängig von der Anzahl der parallelen Eingangslatches).

Man betrachte ein HOL_H-Koppelelement ohne Eingangslatches, dessen logische Ausgangsqueues aus einem modifizierten Zentralpuffer, wie im vorherigen Abschnitt beschrieben, aufgebaut ist. Da Eingangslatches fehlen, muß der Transfer von Paketen zwischen zwei Koppelelementen durch jeweils B Handshake-Leitungen pro Koppelelementport gesteuert werden, um eine optimale Netzleistung zu erzielen (siehe Abschnitt 3.2.2). Dieser hohe Systemaufwand kann auf jeweils eine Handshake-Leitung pro Koppelelementport verringert werden, wenn Eingangslatches verwendet werden (siehe Abschnitt 3.2.2). Da davon ausgegangen wird, daß alle betrachteten Koppelelemente die gleiche Gesamtpufferkapazität C (Summe aus den Längen der einzelnen Eingangspuffer und des Zentralpuffers) besitzen, wird für Eingangslatches kein zusätzlicher Pufferplatz benötigt. Vielmehr wird bei der Überführung eines Koppelelements ohne Eingangslatches in ein Koppelelement mit Eingangslatches der Zentralpuffer im Koppelelement ohne Eingangslatches um die Anzahl der Latches verkleinert und diese freigewordenen Pufferplätze dann für die Latches verwendet. In Abbildung 6.25a) sind die zu einem Zentralpufferplatz zugehörigen Schaltmatrixen an den Pufferein- und Ausgängen gezeigt. Hierfür werden also $2*B$ Schaltpunkte benötigt. Wird dieser Zentralpufferplatz nun in ein Eingangslatch umgewandelt, so wird nur ein Schaltpunkt benötigt, wie in Abbildung 6.25b) gezeigt. Es werden somit $2*B-1$ Schaltpunkte eingespart. Auch im Kontrollpfad entfällt Hardware (z. B. verkleinern sich die Register für den nächsten Ein- und Ausgang, und die Ein- und Ausgangsselektionsregister). Mit diesen Hardwareeinsparungen ist es möglich, den Mehraufwand an Hardware zu kompensieren, welcher bei der Implementierung der logischen Ausgangsqueues entsteht (siehe vorherigen Abschnitt). Dies wird hier jedoch nicht weiter ausgeführt, da die minimale Gesamtpufferkapazität eines Koppelelements und nicht der Mehraufwand an Hardware der entscheidende limitierende Faktor der Implementierung eines HOL_H-Koppelelements ist. Dies verstärkt sich noch, wenn Eingangslatches implementiert werden,wie im folgenden gezeigt wird.

Die Gesamtpufferkapazität C eines Koppelelements setzt sich aus der Summe der Längen der einzelnen Eingangspuffer und des Zentralpuffers zusammen. Wie im vorherigen Abschnitt diskutiert, muß der Zentralpuffer eine Mindestgröße Z_{min} aufweisen. Werden Eingangslatches verwendet, so benötigt ein $B \times B$ HOL_H-Koppelelement B^{H-1} parallele Latches pro Eingangsport.

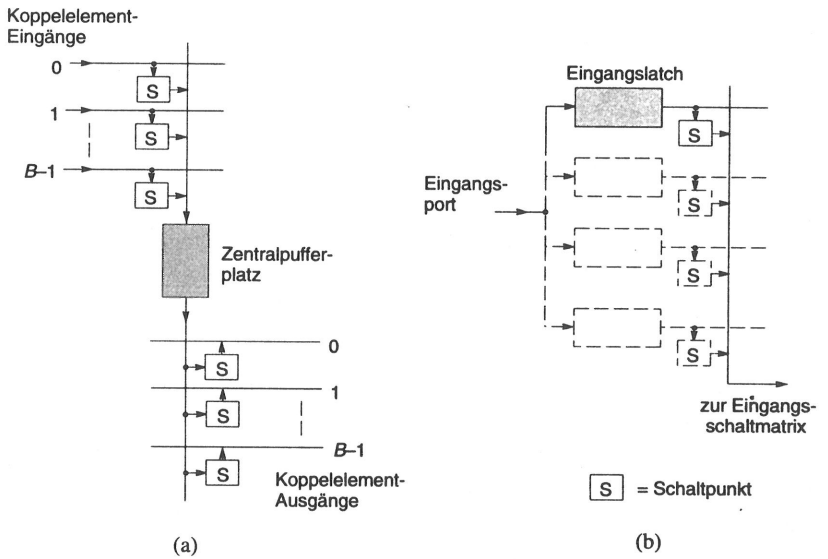


Abbildung 6.25: *Teile der zugehörigen Schaltmatrixen (a) eines Zentralpufferplatzes und (b) eines Eingangslatches.*

So kann die minimale Gesamtpufferkapazität C_{min} eines $B \times B$ HOL_H-Koppelelements mit Eingangslatches zu

$$C_{min} = Z_{min} + B^H = D_{min} * B^{H-1} + B^H \qquad (20)$$

berechnet werden. Unter der Annahme, daß jede logische Ausgangsqueue eine Mindestlänge von vier Paketen haben muß, sind für verschiedene Koppelelementgrößen und HOL-Blockierungsordnungen die Mindestpufferkapazitäten der Koppelelemente in Tabelle 6.3 aufgelistet.

B \ H	1	2	3	4	5
2	6	12	24	48	96
4	8	32	128	512	2048
8	12	96	768	6144	49152

Tabelle 6.3: *Minimale Gesamtpufferkapazität C_{min} eines $B \times B$ HOL_H-Koppelelements mit Eingangslatches, wenn jede logische Ausgangsqueue eine Mindestlänge von D_{min}=4 Paketen hat.*

Wird wieder von einem vertretbaren Hardwareaufwand von 64 Pufferplätzen pro Koppelelement ausgegangen, so können 2×2 Koppelelemente bis zu einer HOL-Ordnung von H=4, 4×4 Koppelelemente bis zu einer HOL-Ordnung von H=2, und 8×8 Koppelelemente bis zu einer HOL-Ordnung von H=1 aufgebaut werden.

6.3.4.3 *Realisierungsaspekte der Koppelelementeingangspuffer*

Auch die einzelnen logischen Eingangsqueues müssen eine Mindestlänge von $D_{min} > 1$ besitzten, damit das Koppelelement eine optimale Leistung unter gleichverteiltem Verkehr erzielt. Da ein $B \times B$ HOL_H-Koppelelement insgesamt B^H logische Eingangsqueues besitzt, muß für die minimale Gesamtpufferkapazität des Koppelelementes gelten:

$$C_{min} = D_{min} * B^H. \tag{21}$$

Unter der Annahme, daß jede logische Eingangsqueue eine Mindestlänge von vier Paketen haben muß, sind für verschiedene Koppelelementgrößen und HOL-Blockierungsordnungen die minimale Gesamtpufferkapazität der Koppelelemente in Tabelle 6.4 aufgelistet. Es ist zu erkennen, daß für höhere HOL-Ordnungen sowie für größere Koppelelemente die Gesamtpufferkapazität wiederum sehr groß gewählt werden muß. Wird von einem vertretbaren Hardwareaufwand von 64 Pufferplätzen pro Koppelelement ausgegangen, so können 2×2 Koppelelemente bis zu einer HOL-Ordnung von H=4, 4×4 Koppelelemente bis zu einer HOL-Ordnung von H=2, und 8×8 Koppelelemente bis zu einer HOL-Ordnung von H=1 aufgebaut werden.

B \ H	1	2	3	4	5
2	8	16	32	64	128
4	16	64	256	1024	4096
8	32	256	2048	16384	131072

Tabelle 6.4: *Minimale Gesamtpufferkapazität C_{min} eines $B \times B$ HOL_H-Koppelelements mit Eingangspuffern, wenn jede logische Eingangsqueue eine Mindestlänge von D_{min}=4 Paketen hat.*

Zusammenfassend kann festgestellt werden, daß der limitierende Hardwarefaktor die minimale Gesamtpufferkapazität des Koppelelements ist; dies gilt für HOL_H-Koppelelemente mit Ausgangspuffern (mit und ohne Eingangslatches), wie auch für Koppelelemente mit Eingangspuffern. Ein Koppelelement mit konventionellem Zentralpuffer kann hierbei ohne Schwierigkeiten in ein HOL_H-Koppelelement mit Ausgangspuffern (mit und ohne Eingangslatches) umwandelt werden. Eine relativ hohe HOL-Blockierungsordnung von 4 bis 5 kann bei einem vertretbaren Hardwareaufwand nur für 2×2 Koppelelemente erreicht werden.

Zusammenfassend kann festgestellt werden, daß sich ein Koppelelement mit konventionellem Zentralpuffer ohne Schwierigkeiten in ein HOL_H-Koppelelement umwandeln lassen kann. Der limitierende Faktor hierbei ist die minimale Gesamtpufferkapazität mit der das Koppelelement ausgestattet werden muß; dies gilt für HOL_H-Koppelelemente mit und ohne Eingangslatches. Eine relativ hohe HOL-Blockierungsordnung von 4 bis 5 kann bei einem vertretbaren Hardwareaufwand nur für 2×2 Koppelelemente erreicht werden.

Nachdem in diesem und in den vorherigen Abschnitten die Klassifikation von HOL_H-Netzen eingeführt und Realisierungsaspekte diskutiert wurden, werden in den nächsten Abschnitten die Leistungsfähigkeit solcher Netze unter gleich- und ungleichverteilten Verkehren untersucht.

6.3.5 HOL_H-Verbindungsnetze unter gleichverteilten Verkehren

In diesem Abschnitt werden die Leistungscharakteristiken von vier Netzen aus verschiedenen HOL-Blockierungsklassen unter gleichverteiltem Verkehr studiert: ein HOL_1-Netz (ein Netz mit 2×2 eingangsgepufferten Koppelelementen), ein HOL_2-Netz (ein Netz mit 2×2 ausgangsgepufferten Koppelelementen), ein HOL_3-Netz (ein Netz mit 2×2 ausgangsgepufferten Koppelelementen mit zwei parallelen Puffern pro Ausgangsport) und ein HOL_4-Netz (ein Netz mit 2×2 ausgangsgepufferten Koppelelementen mit vier parallelen Puffern pro Ausgangsport). Drei Leistungsparameter werden hierbei zum Leistungsvergleich herangezogen, und zwar die mittlere Netzauslastung, der mittlere relative Paketdurchsatz und die mittlere Paketdurchlaufzeit (siehe Abschnitt 5.1). Es wird davon ausgegangen, daß Pakete, die nicht in das Netz eingespeist werden können, in den PEs verworfen werden.

In den Koppelelementen mit parallelen Ausgangspuffern muß ein Entscheidungsmechanismus in jedem Netzzyklus einen der Ausgangspuffer pro Ausgangsport auswählen, um dessen Kopfpaket zur nächsten Netzstufe weiterzuleiten. Im folgenden wird angenommen, daß ein Puffer zufällig ausgewählt wird. Wenn dieser ausgewählte Puffer leer ist und ein anderer Puffer dieses Ausgangsports ein Paket gepuffert hat, so wird dieser nicht-leere Puffer ausgewählt, um keine Netzzyklen zu verschwenden. Simulationen haben gezeigt, daß andere Auswahlverfahren, wie z. B. eine zyklische Auswahl, zu einer mit der zufälligen Auswahl vergleichbaren Netzleistung führen. Des weiteren wird angenommen, daß jeder Koppelelementport eine Steuerleitung zur Datenflußsteuerung besitzt. Ist ein Koppelelement in der Lage, ein Paket zu senden, so wird das Paket auf den Ausgangsport gelegt und die Steuerleitung des Ports auf 1 gelegt. Wenn das Empfangskoppelelement in der Lage war, daß Paket zu empfangen, so legt dieses Koppelelement die Steuerleitung auf 0, um dem sendenden Element den Paketempfang zu bestätigen.

Theorem 1 besagt, daß HOL-Blockierungen höherer Ordnung von der Benutzung von Puffern durch verschiedene Wege in einem Netz, jedoch nicht von der Pufferungsstrategie in den Puffern beeinflußt wird. Um die Leistungsuntersuchung von HOL_H-Netzen einfach zu halten, werden im folgenden Puffer mit der FIFO-Pufferstrategie angenommen. Andere Strategien, wie z. B.

ein zufälliger Pufferzugriff, führen zu ähnlichen Netzleistungen, sind jedoch meistens hardwareaufwendiger als die FIFO-Strategie.

In Abbildung 6.26 ist die mittlere Netzauslastung über der Verkehrsrate von 1024 × 1024 Netzen mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von jeweils 40 Paketen pro Koppelelement (d.h. 20 Pakete pro Puffer in den HOL_1- und HOL_2-Netzen, 10 Pakete pro Puffer im HOL_3-Netz und 5 Pakete pro Puffer im HOL_4-Netz) aufgetragen. Der Gesamtpufferplatz pro Koppelelement ist als konstant für die verschiedenen Netze angenommen, um die Leistungen von Netzen mit ähnlichem Hardwareaufwand zu vergleichen.

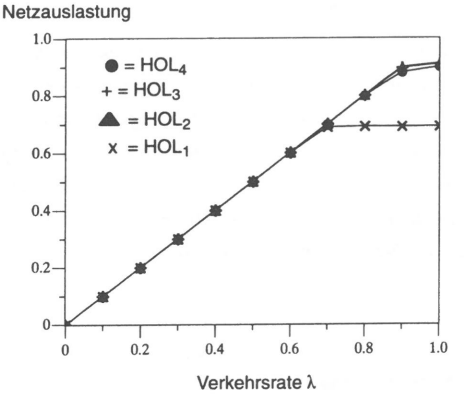


Abbildung 6.26: *Mittlere Netzauslastung vs. Verkehrsrate in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter gleichverteiltem Verkehr.*

Das HOL_1-Netz weist eine geringere maximale Auslastung als alle anderen Netze auf, da nur in diesem Netz konventionelle HOL-Blockierungen auftreten, die die Netzleistung beeinträchtigen. Die HOL_2-, HOL_3- und HOL_4-Netze zeigen eine vergleichbare Auslastungscharakteristik auf, welches nun erklärt wird. Man betrachte zum Beispiel das HOL_3-Netz in Abbildung 6.19. In dieser Koppelelementarchitektur kann ein Konflikt auftreten, wenn in beiden parallelen Ausgangspuffern eines Ausgangsports Pakete gepuffert sind. Ein Ausgangsport kann nur ein Paket pro Netzzyklus weiterleiten. Deshalb muß ein Datenflußkontrollmechanismus jeweils einen der Puffer pro Zyklus auswählen, der sein Kopfpaket weiterleiten darf, während der andere Puffer in diesem Zyklus blockiert ist. In dieser Untersuchung wird angenommen, daß ein Puffer zufällig

ausgewählt wird. Unter gleichverteiltem Verkehr haben Pakete, die an den Eingängen von Koppelelement b_l ankommen und für z. B. den oberen Ausgangsport bestimmt sind, zufällige Zieladressen, so daß sie zufällig auf die beiden Puffer $QO_{0,u}$ und $QO_{0,d}$ dieses Ausgangsports verteilt werden. Da in jedem Zyklus einer dieser Puffer zufällig vom Datenflußkontrollmechanismus ausgewählt wird, ist die Leistungscharakteristik dieses Koppelelements unter gleichverteiltem Verkehr vergleichbar mit der eines normal ausgangsgepufferten Koppelelements, wenn beide Koppelelemente die gleiche Gesamtpuffergröße besitzen. Diese Erklärung trifft auch für Koppelelemente mit mehr als zwei parallelen Puffern pro Elementausgangsport zu, solange die Längen der individuellen Puffer nicht zu klein sind (sie sollten länger als vier Pakete sein). Koppelelemente mit kürzeren Puffern erleiden erhöhte Konflikte durch gefüllte Puffer unter höheren Verkehrsraten, so daß mit Leistungseinbußen zu rechnen ist.

Der mittlere relative Durchsatz und die Durchlaufzeit aller Pakete von 1024 × 1024 Netzen mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von jeweils 40 Paketen pro Koppelelement sind in den Abbildungen 6.27(a) und 6.27(b) in Abhängigkeit von der Netzauslastung dargestellt. Die Netzauslastung wurde gewählt, um Differenzen in der Leistungsfähigkeit der einzelnen Netze besser erkennen zu können. Die HOL_2-, HOL_3- und HOL_4-Netze zeigen wieder eine vergleichbare Leistungscharakteristik auf, während im HOL_1-Netz konventionelle HOL-Blokkierungen auftreten, die die Netzleistung beeinträchtigen.

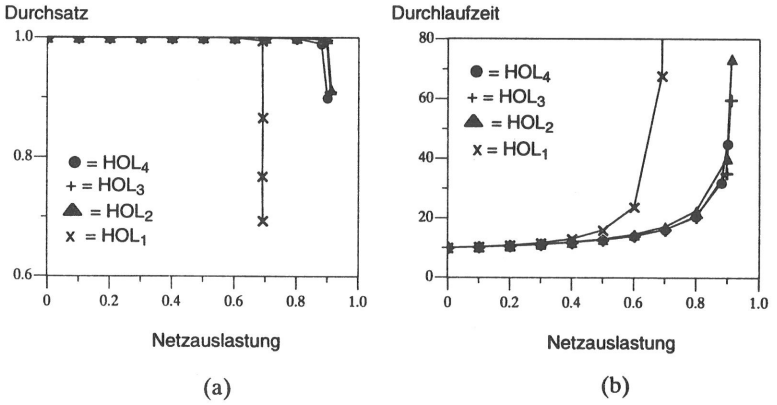


Abbildung 6.27: *Mittlerer (a) relativer Durchsatz und (b) Paketdurchlaufzeit in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter gleichverteiltem Verkehr.*

6.3.6 Leistungsbewertung von HOL_H-Verbindungsnetzen unter statisch ungleichverteilten Verkehren

Die Vorteile von HOL_H-Netzen höherer Ordnung werden unter ungleichverteilten Datenverkehren deutlich. Eine obere Schranke der Auslastung von HOL_H-Netzen unter statisch ungleichverteilten Datenverkehren wird im folgenden hergeleitet. Danach wird die Leistung von HOL_H-Netzen mittels Simulation bestimmt und mit den analytischen Resultaten verglichen. Hierbei werden statische Hot-Spot- und Bit-Reverse-Permutationsverkehre betrachtet. Diese Verkehre wurden gewählt, da sie zu einem Netzausgangskonflikt (Hot-Spot-Verkehr) und zu Konflikten innerhalb des Netzes (Bit-Reverse-Permutationsverkehr) führen, die starke Leistungseinbußen in Delta-Netzen hervorrufen (siehe Abschnitte 4.1.2 und 4.1.3.1).

6.3.6.1 Analyse von HOL_H-Verbindungsnetzen unter statisch ungleichverteilten Verkehren

Bei dieser Analyse werden statisch ungleichverteilte Verkehre angenommen, bei denen ein Sättigungsbaum an Ausgangslinks der Stufe s–R ($0 < R \leq s$) beginnt und sich bis zu den Netzeingängen erstreckt. Beispiele solcher Verkehre sind der Hot-Spot-Verkehr mit $R = s$ (siehe Abschnitt 4.1.2.1) und der Bit-Reverse-Permutationsverkehr mit $R = \lfloor s/2 \rfloor$ (siehe Abschnitt 4.1.3.1). R repräsentiert hierbei die Tiefe des Sättigungsbaums. λ_0 sei die maximale Verkehrsrate, bei der sich noch kein Baum im Netz unter statischem ungleichverteilten Verkehr bildet. Für Raten größer λ_0 wird sich ein Sättigungsbaum von einem Ausgangslink der Stufe s–R ($0 < R \leq s$) aufbauen, der bis zu den Netzeingängen reicht. Im folgenden wird ein idealisiertes Netz angenommen, in dem unter gleichverteiltem Verkehr keine Vermittlungskonflikte entstehen. Diese Annahme führt zu einer oberen Schranke der Netzauslastung. Unter den gemachten Annahmen trifft Theorem 2 zu:

Theorem 2: Die obere Schranke A_{ub} der Auslastung eines mehrstufigen HOL_H-Delta-Netzes unter statischem ungleichverteilten Verkehr ist gegeben durch:

$$A_{ub} = \begin{cases} \lambda & , \ \lambda \leq \lambda_0 \quad (22a) \\ \lambda_0 + \left(1 - \frac{1}{B^{H-1}}\right) * (1-h) * (\lambda - \lambda_0) & , \ \lambda > \lambda_0, \ 1 \leq H \leq R+1 \quad (22b) \\ \lambda_0 + \left(1 - \frac{1}{B^{R}}\right) * (1-h) * (\lambda - \lambda_0) & , \ \lambda > \lambda_0, \ H > R+1 \quad (22c) \end{cases}$$

Beweis von (22a): Bei Verkehrsraten kleiner λ_0 existiert noch kein Sättigungsbaum im Netz, der bis zu den Netzeingängen reicht. Die heißen Pakete beeinflussen die kalten nur unwesentlich, so daß die Netzauslastung nur minimal geringer als unter gleichverteiltem Verkehr ist. Die obere Schranke der Auslastung ist deshalb A_{up}=λ.

Beweis von (22b): Bei Verkehrsraten größer λ_0 existiert ein Sättigungsbaum bis zu allen Netzeingängen, der die Koppelelementpuffer, die zum Baum gehören, blockiert und füllt. Definition 2 besagt, daß ein HOL_H-Netz HOL_H-blockierend, aber nicht HOL_H-blockierend ist. In diesen Netzen teilen sich also Wege, die sich von Eingangslinks der Stufe s–1 bis zu Eingangslinks der Stufe s–H erstrecken, keine Puffer, während Wege, die sich von Eingangslinks der Stufe s–1 bis zu Eingangslinks der Stufe s–H–1 erstrecken, Puffer teilen (siehe Theorem 1). Man nehme nun an, es sei $H \leq R+1$. In diesem Fall sind die Längen der Wege, die sich keine Puffer teilen, kleiner oder gleich der Länge des Sättigungsbaums, so daß eine Veränderung von H die Effekte des Sättigungsbaums im Netz verändert. Ein Sättigungsbaum, bestehend aus gefüllten Puffern in einem HOL_H-Netz, wird nur Pakete blockieren, die von einem Eingangslink der Stufe s–1 zu einem heißen Eingangslink der Stufe s–H propagieren (siehe Definition 1). Bedingt durch die Delta-Netztopologie ist die Wahrscheinlichkeit P_{HL}, daß ein Datenpaket mit einer gleichverteilten Zieladresse an einen bestimmten Eingangslink der Stufe s–H in einem Delta-Netz, bestehend aus $B \times B$ Koppelelementen, adressiert ist:

$$P_{HL} = \frac{1}{B^{s-1-(s-H)}} = \frac{1}{B^{H-1}} . \tag{23}$$

Nun erinnere man sich, daß bei ungleichverteilten Verkehren ein Paket mit der Wahrscheinlichkeit h heiß ist, während es mit der Wahrscheinlichkeit 1–h gleichverteilt ist (siehe Abschnitt 4.1.2.1). Die Wahrscheinlichkeit P_{HP}, daß ein Paket auf einem heißen Weg vom Netzeingang bis zu einem Eingangslink von Stufe s–H propagiert, ist $P_{HP} = h + (1-h)*P_{HL}$, und die Wahrscheinlichkeit P_{NHP}, daß ein Paket nicht auf einem heißen Weg vom Netzeingang bis zu einem Eingangslink von Stufe s–H propagiert, ist:

$$P_{NHP} = 1-P_{HP} = (1-h)*(1-P_{HL}) = (1-h)*(1-\frac{1}{B^{H-1}}). \tag{24}$$

Die angebotene Verkehrsrate kann also in einen heißen und einen kalten Teil unterteilt werden:

$$\lambda = \underbrace{P_{NHP} * \lambda}_{\text{kalter Teil}} + \underbrace{(1-P_{NHP})* \lambda}_{\text{heißer Teil}} \tag{25}$$

Nun betrachte man eine Verkehrsrate größer λ_0. Dann ist $\lambda=\Delta + \lambda_0$, mit $\Delta>0$, und

$$\lambda = \underbrace{P_{NHP} * \lambda}_{\text{kalter Teil}} + \underbrace{(1-P_{NHP})*\Delta + (1-P_{NHP})*\lambda_0}_{\text{heißer Teil}} . \tag{26}$$

Da der Sättigungsbaum alle Netzeingänge erreicht, wird nur der Teil $(1-P_{NHP})*\lambda_0$ des heißen Verkehrsteils, der den Sättigungsbaum hervorruft, in das Netz akzeptiert. Der Verkehrsteil

$(1-P_{NHP})*\Delta$ wird in den Quellen verworfen. Der kalte Teil der Verkehrsrate wird nicht vom Sättigungsbaum beeinflußt und wird, bedingt durch die Annahme eines idealisierten Netzes, vollständig vom Netz akzeptiert und an die Ausgänge weitergeleitet. So ist die obere Schranke der Auslastung eines HOL_H-Netzes ($H \leq R+1$) unter ungleichverteiltem Verkehr gleich:

$$A_{ub} = P_{NHP} * \lambda + (1-P_{NHP})*\lambda_0 = \lambda_0 + P_{NHP} *(\lambda - \lambda_0). \quad (27)$$

Einsetzen von Gleichung (24) in Gleichung (27) ergibt schließlich:

$$A_{ub} = \lambda_0 + \left(1 - \frac{1}{B^{H-1}} \right) * (1 - h) * (\lambda - \lambda_0). \quad (28)$$

Beweis von (22c): $H > R+1$. In diesem Fall sind die Längen der Wege in einem HOL_H-Netz, die sich keine Puffer teilen, länger als die Länge des Sättigungsbaums, so daß eine Veränderung von H die Effekte des Sättigungsbaums im Netz nicht verändern kann. Die Netzauslastung ist somit unabhängig von H und wird nur von der Länge R des Sättigungsbaums bestimmt. Diese Netze haben also ein Auslastungsverhalten von HOL_{R+1}-Netzen, so daß deren Auslastung durch Gleichung (22b) mit $H=R+1$ bestimmt ist. □

Die Gleichungen (22a-c) zeigen, daß eine größer werdende HOL-Blockierungsordnung H zu einer steigenden maximalen Netzauslastung führt, so lange $H \leq R+1$ ist. Für höhere Verkehrsraten wird die Auslastung von HOL_H-Netzen unter ungleichverteilten Verkehren kleiner sein als in Gleichungen (22a-c) berechnet. Diese Gleichungen sind für idealisierte Netze hergeleitet, in denen keine Konflikte unter gleichverteiltem Verkehr auftreten. Bei hohen Raten trifft diese Annahme nicht mehr zu, so daß nur eine kleinere Auslastung erreicht werden kann. In den folgenden Abschnitten wird das Verhalten von HOL_H-Netzen unter statischen Hot-Spot- und Bit-Reverse-Permutationsverkehren diskutiert. Hierbei wird die analytische obere Schranke der Auslastung (Gleichungen (22a-c)) mit Simulationsergebnissen verglichen.

6.3.6.2 *Leistung von HOL_H-Verbindungsnetzen unter statischem Hot-Spot-Verkehr*

In Abbildung 6.28 ist die mittlere Netzauslastung über der Verkehrsrate von 1024×1024 Netzen mit 2×2 Koppelelementen mit einer Gesamtpuffergröße von jeweils $C=40$ Paketen pro Koppelelement aufgetragen. Es wurde ein Hot-Spot-Verkehr mit $h=0.01$ (Abbildung 6.28a) und mit $h=0.1$ (Abbildung 6.28b) simuliert. Auch sind die Auslastung eines HOL_2-Netzes unter gleichverteiltem Verkehr und die analytische obere Schranke der Auslastung (Gleichungen (2) und (22)) unter statischem Hot-Spot-Verkehr gezeigt. Die Leistung eines HOL_2-Netzes (siehe Abbildung 6.20b) unter gleichverteiltem Verkehr ist angegeben, um die erreichbaren Leistungssteige-

rungen durch höhere HOL-Blockierungsordnungen beurteilen zu können. In Abschnitt 6.3.5 wurde gezeigt, daß die Leistungen von HOL_H-Netzen mit $H \geq 2$ unter gleichverteiltem Verkehr vergleichbar sind. So wurde für die folgende Diskussion ein HOL_2-Netz unter gleichverteiltem Verkehr gewählt, da ein HOL_H-Netz unter ungleichverteilten Verkehren nie leistungsfähiger sein kann als ein HOL_H-Netz mit $H \geq 2$ unter gleichverteiltem Verkehr.

Die Erhöhung der HOL-Blockierungsordnung H führt zu einer höheren maximalen Netzauslastung unter statischem Hot-Spot-Verkehr (siehe Abbildung 6.28). Dieses Verhalten kann, wie von der Analyse her zu erwarten war, bei beiden Verkehrsszenarien beobachtet werden. Auch ist zu sehen, das der Leistungsgewinn durch eine Erhöhung der HOL-Blockierungsordnung kleiner wird für größere Blockierungsordnungen. Deshalb ist der zu erreichende Leistungszuwachs mit dem verbundenen vergrößerten Hardwareaufwand für manche HOL-Blockierungsordnungen nicht mehr zu rechtfertigen.

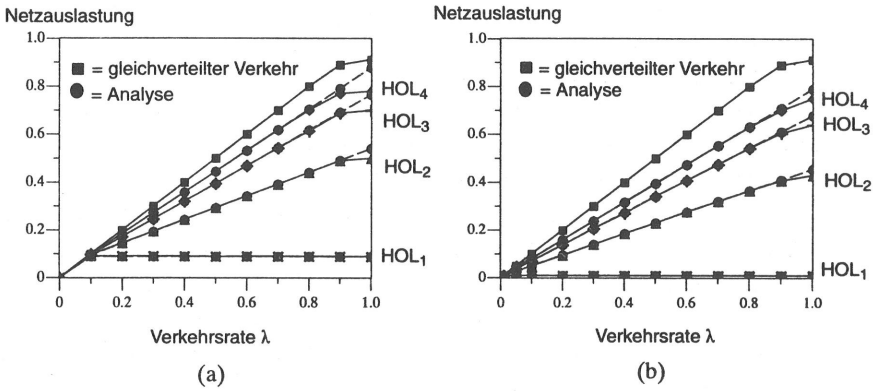


Abbildung 6.28: *Mittlere Netzauslastung vs. Verkehrsrate in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr mit (a) h=0.01 und (b) h=0.1.*

In den Abbildungen 6.29 und 6.30 sind der relative Durchsatz und die Durchlaufzeit der Nontree- (Abbildung 6.29) und Tree-Pakete (Abbildung 6.30) der vier verschiedenen HOL-Blockierungsnetze unter statischem Hot-Spot-Verkehr und des HOL_2-Netzes unter gleichverteiltem Verkehr in Abhängigkeit von der Netzauslastung dargestellt. Die Netzauslastung wurde gewählt, um Differenzen in der Leistungsfähigkeit der einzelnen Netze besser erkennen zu können. Im HOL_1-Netz müssen die Nontree-Pakete durch einen heißen Puffer in der ersten Netzstufe vermittelt werden, so daß sie vom Sättigungsbaum blockiert werden (in den HOL_2-, HOL_3- und HOL_4-Netzen tritt dies nicht auf). Diese Blockierungen erklären den geringen Durchsatz und die hohe Durchlaufzeit der Nontree-Pakete im HOL_1-Netz (siehe Abbildung 6.29). Nun betrachte man Tree-Pakete, die den Sättigungsbaum in Stufe s–2 verlassen. Im HOL_1-Netz müs-

sen diese Pakete durch zwei heiße Puffer (in den Stufen s–1 und s–2), im HOL_2-Netz treffen die Pakete auf einen heißen Puffer (in Stufe s–1), während die Pakete in den HOL_3- und HOL_4-Netzen in keinen heißen Puffern gepuffert werden (siehe auch Abbildung 6.16). Demnach verlassen Tree-Pakete in Netzen mit steigender HOL-Blockierungsordnung den Sättigungsbaum früher. Dies erklärt die Charakteristiken des Durchsatzes und der Durchlaufzeit der Tree-Pakete in den vier Netzen in Abbildung 6.30. Die Abbildungen 6.28, 6.29 und 6.30 verdeutlichen, daß eine steigende HOL-Blockierungsordnung zu einer steigenden Netzauslastung, einem steigenden Paketdurchsatz und einer sinkenden Paketdurchlaufzeit führt. Deshalb ist ein Netz aus einer höheren HOL-Blockierungsklasse immer leistungsfähiger unter statischem Hot-Spot-Verkehr als ein Netz mit gleichem N und B aus einer niedrigeren Klasse.

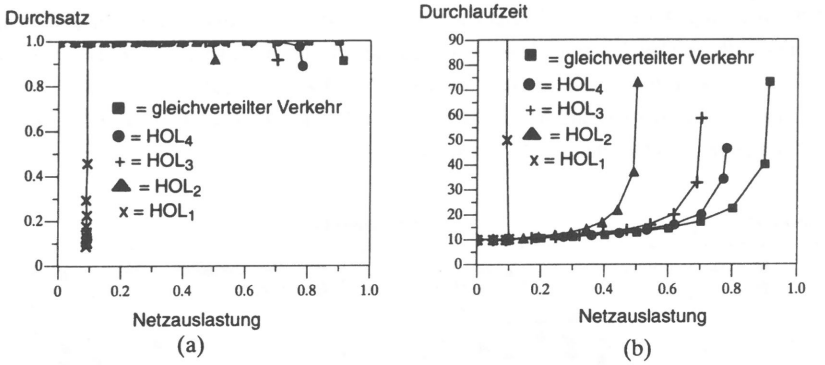


Abbildung 6.29: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit der Nontree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr mit h=0.01.*

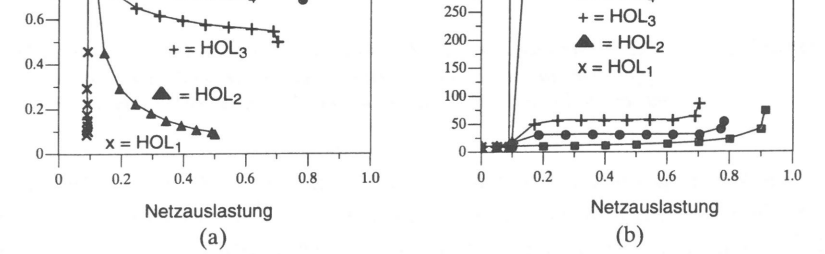


Abbildung 6.30: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit der Tree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Hot-Spot-Verkehr mit h=0.01.*

6.3.6.3 Leistung von HOL_H-Verbindungsnetzen unter statischem Bit-Reverse-Permutationsverkehr

In diesem Abschnitt wird der Effekt von Bit-Reverse-Permutationsverkehren in HOL_H-Netzen diskutiert. In Abbildung 6.31 ist die mittlere Netzauslastung über der Verkehrsrate von 1024×1024 Netzen mit 2×2 Koppelelementen mit einer Gesamtpuffergröße von jeweils C=40 Paketen pro Koppelelement aufgetragen. Es wurde ein Bit-Reverse-Permutationsverkehr mit h=0.1 (Abbildung 6.31a) und mit h=0.3 (Abbildung 6.31b) simuliert. Auch sind die Auslastung eines HOL_2-Netzes unter gleichverteiltem Verkehr und die analytische obere Schranke der Auslastung (Gleichungen (5) und (22)) unter statischem Bit-Reverse-Permutationsverkehr gezeigt. Wieder führt die Erhöhung der HOL-Blockierungsordnung zu einer Steigerung der maximalen Netzauslastung. Abbildung 6.31 zeigt auch, daß für h=0.3 die Analyse mit der Simulation auf bei hohen Raten übereinstimmt. In diesem Fall ist die maximale Verkehrsrate des gleichverteilten Verkehrs λ=1–h=0.7, und die Netze mit einer HOL-Blockierungsordnung von H>1 weisen bei diesen Raten noch eine ideale Auslastungscharakteristik auf (siehe Abbildung 6.26), wie bei der Herleitung von Gleichung (22) angenommen wurde.

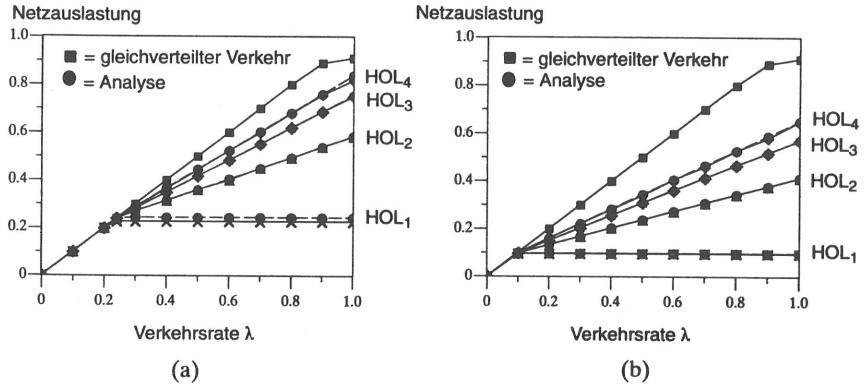


Abbildung 6.31: *Mittlere Netzauslastung vs. Verkehrsrate in einem 1024×1024 Netz mit 2×2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Bit-Reverse-Permutationsverkehr mit (a) h=0.1 und (b) h=0.3.*

In den Abbildungen 6.32 und 6.33 sind der relative Durchsatz und die Durchlaufzeit der Nontree- (Abbildung 6.32) und Tree-Pakete (Abbildung 6.33) der vier verschiedenen HOL-Blockierungsnetze unter statischem Hot-Spot-Verkehr und des HOL_2-Netzes unter gleichverteiltem Verkehr in Abhängigkeit von der Netzauslastung dargestellt. Die Netzauslastung wurde gewählt, um Differenzen in der Leistungsfähigkeit der einzelnen Netze besser erkennen zu können. Die Abbildungen 6.31 bis 6.33 verdeutlichen, daß eine steigende HOL-Blockierungsord-

nung zu einer steigenden Netzauslastung und einem steigenden Paketdurchsatz, und zu einer sinkenden Paketdurchlaufzeit auch unter Bit-Reverse-Permutationsverkehren führt. So ist ein Netz aus einer höheren HOL-Blockierungsklasse immer leistungsfähiger unter statischen Hot-Spot- und Bit-Reverse-Permutationsverkehren als ein Netz mit gleichem N und B aus einer niedrigeren Klasse.

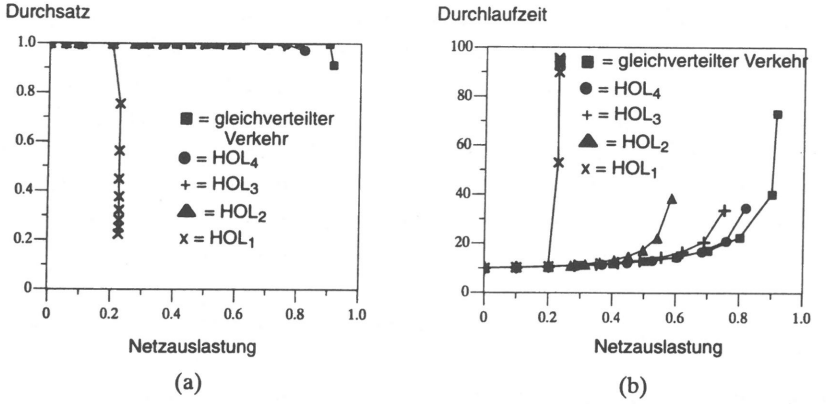


Abbildung 6.32: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit der Nontree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Bit-Reverse-Permutationsverkehr mit h=0.1.*

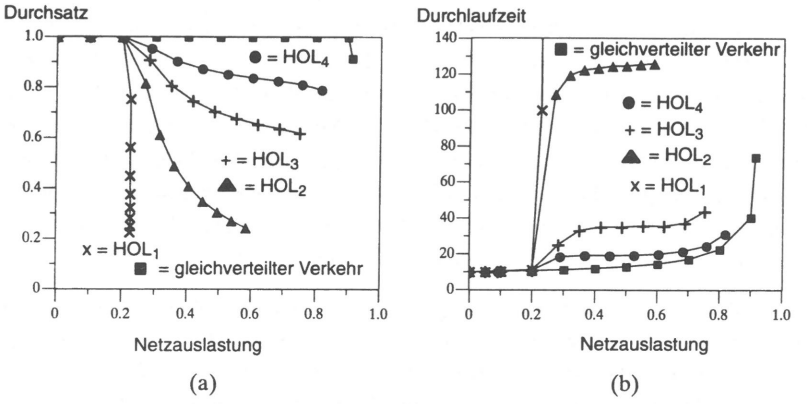


Abbildung 6.33: *Mittlerer (a) relativer Durchsatz und (b) Durchlaufzeit der Tree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter statischem Bit-Reverse-Permutationsverkehr mit h=0.1.*

6.3.7 Leistungsbewertung von HOL_H-Verbindungsnetzen unter temporär ungleichverteilten Verkehren

Um den Einfluß der HOL-Blockierungsordnung von Netzen auf das transiente Sättigungsbaumverhalten zu bestimmen, werden nun die Auswirkungen temporärer Hot-Spot- und Bit-Reverse-Permutationsverkehre auf die Netzleistung von HOL_1-, HOL_2-, HOL_3- und HOL_4-Netzen aufgebaut aus 2×2 eingangsgepufferten Koppelelementen (HOL_1), ausgangsgepufferten Koppelelementen (HOL_2), ausgangsgepufferten Koppelelementen mit zwei parallelen Puffern pro Ausgangsport (HOL_3) und ausgangsgepufferten Koppelelementen mit vier parallelen Puffern pro Ausgangsport (HOL_4) in den nächsten Abschnitten simulativ untersucht. Hierzu werden Systeme betrachtet, die Quellen mit unendlich langen Quellpuffern besitzen, die in mehrere logische Queues unterteilt sind. Hierbei besitzt, wie in Abschnitt 6.1.2 beschrieben, das HOL_1-System eine Queue pro Quelle, das HOL_2-System zwei Queues pro Quelle, das HOL_3-System vier Queues pro Quelle und das HOL_4-System acht Queues pro Quelle.

6.3.7.1 Leistung von HOL_H-Verbindungsnetzen unter normalverteiltem Hot-Spot-Verkehr

In den Abbildungen 6.34 und 6.35 sind die Längen der Hot-Spot- und Überlastphasen (Abbildung 6.34) und die mittleren Durchlaufzeiten der Nontree- und Tree-Pakete (Abbildung 6.35) der vier verschiedenen HOL-Blockierungsnetze unter einem normalverteilten Hot-Spot-Verkehr mit λ=0.5, μ=1000 und σ=50 dargestellt. Es ist zu erkennen, daß das HOL_1-Netz nur eine maximale Rate von λ=0.68 unter diesem Hot-Spot-Verkehr bewältigen kann, während die maximale Rate bei dem HOL_2- und HOL_3-Netz bei λ=0.9, und beim HOL_4-Netz bei λ=0.86 liegt. Die geringe maximale zu verarbeitende Verkehrsrate beim HOL_1-Netz ist auf die auftretenden konventionellen HOL-Blockierungen zurückzuführen, die in den restlichen Netzen nicht auftreten. Das HOL_4-Netz weist eine etwas geringere maximale Rate als das HOL_2- und HOL_3-Netz auf, da bei einer Gesamtpufferkapazität von 40 Paketen pro Koppelelement im HOL_4-Netz Koppelelementpuffer der Länge 5 verwendet werden, so daß es zu höheren Konflikten der kalten Pakete bei höheren Raten kommt, als in den HOL_2- und HOL_3-Netzen. Eine Erhöhung der HOL-Blokkierungsordnung hat nur einen unwesentlichen Einfluß auf die Länge der Hot-Spot-Phase und, für H>1, auch auf die Länge der Überlastphase (siehe Abbildung 6.34).

Die Durchlaufzeit der Nontree-Pakete zeigt, daß nur im HOL_1-Netz diese Pakete vom Sättigungsbaum blockiert und verzögert werden (siehe Abbildung 6.35a), wie es im vorherigen Abschnitt erläutert worden ist. In den HOL_1- und HOL_2-Netzen müssen alle Tree-Pakete durch einen heißen Puffer in der ersten Netzstufe propagieren (siehe Abbildungen 6.16a und 6.16b), so daß diese Netze einen vergleichbaren Durchsatz der Tree-Pakete aufweisen (siehe Abbildung 6.35b). In den HOL_3- und HOL_4-Netzen treffen die Tree-Pakete, die den Sättigungsbaum in der zweiten Netzstufe verlassen, auf keine heißen Puffer. Dies erklärt die sinkende Durchlaufzeit der Tree-Pakete bei steigender HOL-Blockierungsordnung (siehe Abbildung 6.35b).

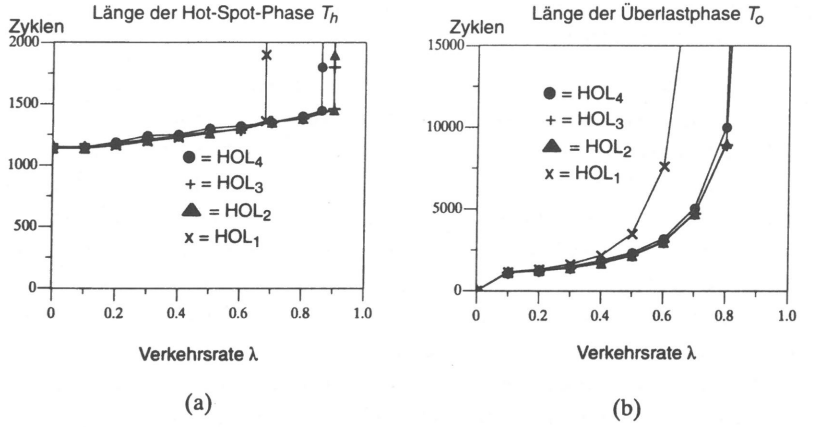


Abbildung 6.34: *Mittlere Länge der (a) Hot-Spot- und (b) Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter normalverteiltem Hot-Spot-Verkehr mit μ=1000 und σ=50.*

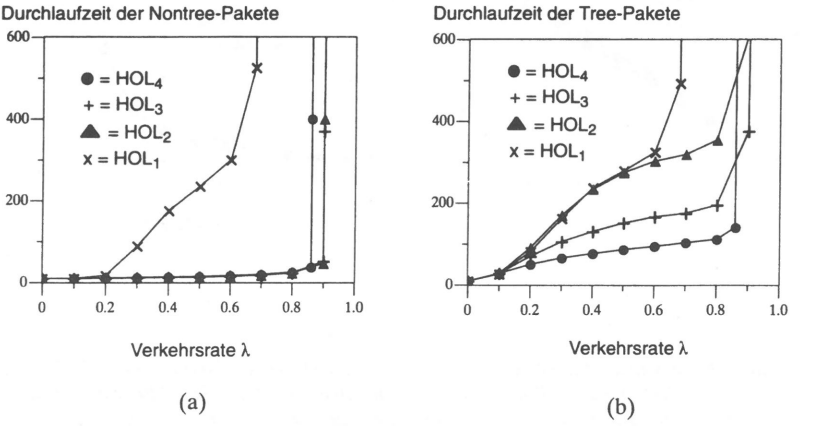


Abbildung 6.35: *Mittlere Durchlaufzeit der (a) Nontree- und (b) Tree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter normalverteiltem Hot-Spot-Verkehr mit μ=1000 und σ=50.*

6.3.7.2 Leistung von HOL_H-Verbindungsnetzen unter temporärem Bit-Reverse-Permutationsverkehr

Um den Einfluß der HOL-Blockierungsordnung auf das Netzverhalten unter temporären Verkehren, die zu Konflikten innerhalb des Netzes führen, zu untersuchen, wird nun ein temporärer Bit-Reverse-Permutationsverkehr mit h=0.1 von Zyklus 1000 bis Zyklus 1300 betrachtet. Die Abhängigkeiten der Phasenlängen von der Verkehrsrate sind in Abbildung 6.36 gezeigt, während die resultierenden Durchlaufzeiten der Nontree- und Tree-Pakete in Abbildungen 6.37 der vier verschiedenen HOL-Blockierungsnetze unter dem temporären Permutationsverkehr dargestellt sind. Die Leistungen der Netze verhalten sich, wie zu erwarten war, ähnlich denen unter normalverteiltem Hot-Spot-Verkehr. Wiederum ist der negative Effekt der konventionellen HOL-Blockierung im HOL_1-Netz deutlich zu erkennen. Die im vorherigen Abschnitt beobachteten Einflüsse der HOL-Blockierungsordnung auf die Paketdurchlaufzeiten unter normalverteiltem Hot-Spot-Verkehr treten ebenfalls unter dem temporären Permutationsverkehr auf.

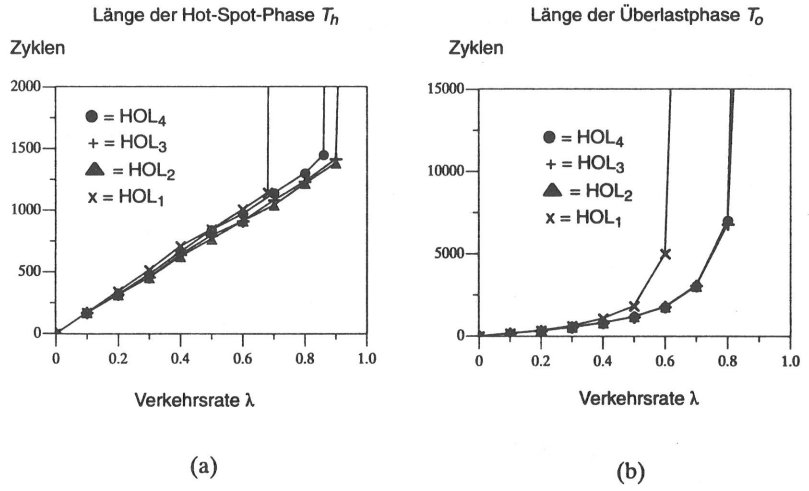


Abbildung 6.36: *Mittlere Länge der (a) Hot-Spot- und (b) Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter temporärem Bit-Reverse-Permutationsverkehr mit h=0.1 von Zyklus 1000 bis Zyklus 1300.*

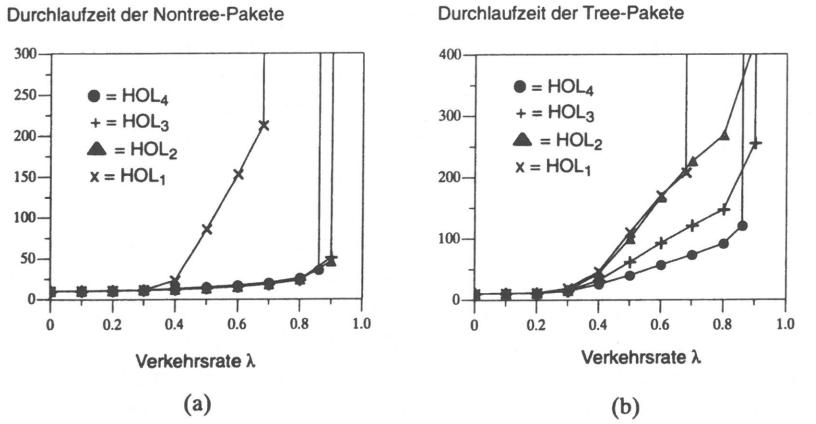


Abbildung 6.37: *Mittlere Durchlaufzeit der (a) Nontree- und (b) Tree-Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit einer Gesamtpuffergröße von 40 Paketen, unter temporärem Bit-Reverse-Permutationsverkehr mit h=0.1 von Zyklus 1000 bis Zyklus 1300.*

Somit führt eine steigende HOL-Blockierungsordnung eines Verbindungsnetzes zu einer steigenden Netzauslastung und steigendem Paketdurchsatz unter statisch ungleichverteilten Datenverkehren, und einer sinkenden Paketdurchlaufzeit unter statisch und temporär ungleichverteilten Datenverkehren. Die Längen der Hot-Spot- und der Überlastphase können mit steigender HOL-Blockierungsordnung für $H>1$ unter temporär ungleichverteilten Verkehren jedoch nicht verbessert werden.

6.3.8 Optimierung einer Koppelelementarchitektur unter Zuhilfenahme des HOL-Blockierungskonzepts

In den vorherigen Abschnitten über die Leistung von HOL_H-Netzen unter statisch und temporär ungleichverteilten Verkehren ist gezeigt worden, daß ein Netz aus einer höheren HOL-Blockierungsklasse immer leistungsfähiger unter diesen Verkehren ist als ein Netz aus einer niedrigeren Klasse. Um Koppelelemente und Netze in Hinblick auf deren Leistung unter ungleichverteilten Verkehren zu optimieren, sollte demnach versucht werden, die Netz- und/oder Koppelelementarchitektur so zu verändern, daß die neue Architektur eine höhere HOL-Blockierungsordnung aufweist, als die ursprüngliche. Dies wird nun beispielhaft an einer in der Literatur vorgestellten Koppelelementarchitektur durchgeführt.

Zur Verbesserung der Leistung von Delta-Netzen unter Hot-Spot-Verkehren haben Peir und Lee in [PeL93] die in Abbildung 6.13b in Abschnitt 6.2.3 dargestellte Koppelelementarchitektur vorgeschlagen. Diese Architektur zeichnet sich durch jeweils zwei parallele Eingangslatches pro Eingangsport und jeweils zwei parallele Ausgangspuffer pro Ausgangsport aus.

Delta-Netze, aufgebaut aus diesen Koppelelementen, können unter Zuhilfenahme der Gleichungen (10) und (12) (N_{ib}=N_{ob}=2) in die Klasse der HOL_2-Netze eingeordnet werden. Um eine höhere Netzleistung unter ungleichverteilten Verkehren zu erzielen, muß versucht werden, durch Änderung der Koppelelementarchitektur die HOL_3-Netzklasse zu erreichen. Diese Klasse zeichnet sich bei 2 × 2 Koppelelementen durch N_{ib}=4 und N_{ob}=2 aus (siehe Abschnitt 6.3.3). Um eine Erhöhung der HOL-Blockierungsordnung und somit eine Leistungssteigerung zu erhalten, muß demnach die Konfiguration der Eingangslatches geändert werden, so daß zwei alternative Koppelelementarchitekturen sinnvoll sind: ein 2 × 2 Koppelelement ohne Eingangslatches und jeweils zwei parallelen Puffern pro Ausgangsport (Abbildung 6.21b, **0LAT**), und ein 2 × 2 Koppelelement mit jeweils zwei parallelen Eingangslatches pro Eingangsport und vier parallelen Puffern pro Ausgangsport (Abbildung 6.21c, **4LAT**). In Abbildung 6.38 ist die mittlere Netzauslastung von 1024 × 1024 Netzen mit 2 × 2 Peir/Lee, 0LAT und 4LAT Koppelelementen mit C=40 Paketen unter statischem Hot-Spot-Verkehr mit h=0.01 und λ=0.5 aufgetragen.

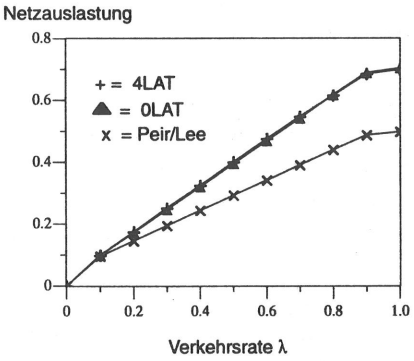


Abbildung 6.38: *Mittlere Netzauslastung vs. Verkehrsrate in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen unter statischem Hot-Spot-Verkehr mit h=0.01.*

Eine kleine Änderung der Koppelelementarchitektur von Peir/Lee führt zu einer signifikanten Auslastungssteigerung um bis zu 20% durch die Erhöhung der HOL-Blockierungsordnung von H=2 zu H=3.

Unter Zuhilfenahme des HOL-Blockierungskonzepts ist es also möglich, die Leistung von Netzen aufgebaut aus den Peir/Lee-Koppelelementen unter ungleichverteilten Verkehren signifikant zu steigern, wobei der Hardwareaufwand bei der Benutzung das 0LAT-Koppelelements sogar verkleinert werden kann.

6.4 Koppelelementarchitektur zur Verkehrssteuerung in Delta-Netzen

In diesem Kapitel wird eine Koppelelementarchitektur vorgestellt, mit der es möglich ist, Sättigungsbaumeffekte in synchronen store-and-forward paketvermittelnden Delta-Netzen unter ungleichverteilten Datenverkehren zu steuern. Nachdem die Prioritätskoppelelementarchitektur beschrieben ist, werden drei verschiedene Prioritätsmechanismen für das Koppelelement und deren Leistungsfähigkeit vorgestellt. Der dritte Mechanismus, die alternierende Transferpriorität, ist parametrisierbar und beinhaltet die ersten beiden Mechanismen. Mit der alternierenden Transferpriorität ist es möglich, das Verhältnis der Durchlaufzeiten von heißen und kalten Paketen durch den Prioritätsparameter einzustellen. So können durch die adaptive Wahl des Prioritätsparameters die Leistungscharakteristiken des Verbindungsnetzes an die Leistungsanforderungen der auf dem Parallelrechner laufenden Algorithmen und Applikationen angepaßt werden, wobei stufenweise zwischen einer minimalen Durchlaufzeit der heißen und der kalten Pakete gewählt werden kann. Eine minimale Durchlaufzeit der kalten Pakete wird hierbei durch die Minimierung der Sättigungsbaumeffekte im Parallelrechnernetz erreicht, indem die Bildung von Sättigungsbäumen bis zu bestimmten Verkehrsraten vollständig unterdrückt wird.

In vielen Anwendungen ist ein *a priori* Wissen über das Auftreten von Unsymmetrien im Datenverkehr vorhanden, so daß Datenpakete in Klassen eingeteilt werden können, bevor sie in das Netz geschickt werden. Dies wird im folgenden zur Steuerung des Sättigungsbaumeffekts in Delta-Netzen ausgenutzt. Es werden Verkehrsszenarien angenommen, unter denen sich temporäre Sättigungsbäume im Netz aufbauen können (z. B. temporäre Hot-Spot- oder Bit-Reverse-Permutationsverkehre). Des weiteren werden Parallelrechnersysteme betrachtet, deren Prozessoren Quellpuffer besitzen, in denen Datenpakete, die nicht sofort in das Verbindungsnetz eingespeist werden können, zwischengepuffert werden, so daß keine Pakete verlorengehen können.

Die in diesem Abschnitt vorgestellte Koppelelementarchitektur kann auch zur Verkehrssteuerung in ATM-Netzen eingesetzt werden. Dies wird anhand von MPEG-codierten Videoströmen in ATM-Netzen im letzten Abschnitt dieses Kapitels aufgezeigt.

6.4.1 Prioritätskoppelelementarchitektur

In Abschnitt 6.3.3 wurde gezeigt, daß bei einem s-stufigen Netz eine maximale HOL-Blockierungsordnung von $H_{max}=s+1$ erreicht werden kann. Dies entspricht einer getrennten Pufferung von allen N Wegen, die von jeweils einer Quelle ausgehen und zu allen N Senken führen. Es resultiert hierbei eine maximale Netzleistung unter ungleichverteilten Verkehren, da alle Pakete, die für den Hot-Spot bestimmt sind, separat von den restlichen Paketen gepuffert werden, so daß die heißen Pakete den gleichverteilten Verkehr nur minimal beeinflussen. Allerdings wird der Hardwareaufwand für solche Koppelelementarchitekturen für größere N zu hoch (es müssen

z. B. N parallele Eingangslatches pro Eingangsport vorhanden sein), so daß eine Implementierung unrealistisch erscheint.

Ist ein *a priori* Wissen über das Auftreten von Unsymmetrien im Datenverkehr vorhanden, so können Datenpakete in Klassen eingeteilt werden (heiß oder kalt), bevor sie in das Netz geschickt werden. Eine einfache Methode, die Beeinflussung der kalten durch die heißen Pakete zu minimieren, ist dann die Benutzung von zwei getrennten Netzen, eins für die heißen Pakete (heißes Netz, Abbildung 6.39) und eins für den gleichverteilten Verkehr (kaltes Netz, Abbildung 6.39). Falls sich heiße Pakete bis in die Quellen zurückstauen, müssen diese auch in den Quellen getrennt von den gleichverteilten Paketen gepuffert werden (Abbildung 6.39), um Blockierungen des gleichverteilten Verkehrs zu vermeiden.

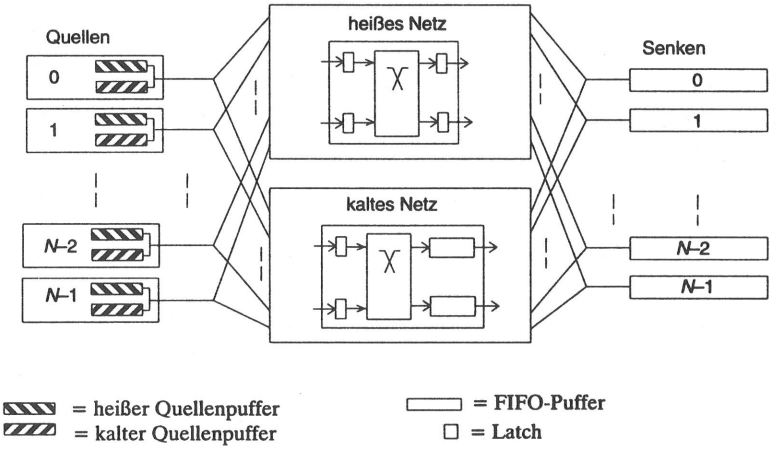


Abbildung 6.39: *Zwei parallele Netze zur Minimierung von Sättigungsbaumeffekten.*

Die Verwendung von zwei getrennten Netzen ist jedoch mit einem großen Systemhardwareaufwand verbunden, da nun doppelt so viele Verbindungsleitungen vorhanden sind (Ein- und Ausgangslinks, sowie netzinterne Links). Dies kann vermieden werden, indem die beiden Netze zu einem Netz zusammengeführt werden, wobei die Koppelelementarchitektur in Abbildung 6.40 verwendet wird [JuS95a]. In dieser Koppelelementarchitektur steht ein heißer Puffer an jedem Ein- und Ausgangsport parallel zu den kalten Puffern zur Verfügung. In den heißen Puffern werden nur heiße Pakete gepuffert, während die kalten Puffer nur kalte Pakete aufnehmen. Auf den Verbindungslinks zwischen Koppelelementen werden heiße und kalte Pakete vermittelt. Es wird angenommen, daß zwei separate bidirektionale Steuerleitungen (die eine für heiße Pakete, die andere für kalte Pakete) pro Koppelelementport zur Verfügung stehen. Diese Leitungen steuern den Datenfluß der beiden Paketklassen zwischen zwei Koppelelementen, wobei sie den Puffer

(kalt oder heiß) festlegen, in dem ein gesandtes Paket im Empfänger gepuffert wird. Man betrachte zwei miteinander verbundene Koppelelemente (Sendeelement und Empfangselement) in zwei aufeinanderfolgenden Netzstufen. Ist zum Beispiel der kalte Puffer des Eingangsports des Empfangselements leer, so legt dieses Koppelelement die dazugehörige Steuerleitung auf 0. Wenn das Sendeelement nun ein kaltes Paket senden kann, stellt es durch die Steuerleitung fest, ob das Empfangselement in der Lage ist, dieses Paket zu empfangen und zu puffern. Ist die Steuerleitung 0 (der Empfangspuffer ist leer), so legt das Sendeelement das Paket auf den Ausgangsportlink und zieht die dazugehörige Steuerleitung auf 1. Das Empfangselement erkennt das Senden eines Pakets an der Signaländerung auf der Steuerleitung und puffert das an seinem Eingangsport anliegende Datenpaket in dem korrespondierenden Eingangspuffer. Der Datenempfang wird dem Sender hierbei nicht bestätigt; der Sender geht davon aus, daß das Senden eines Pakets vom Empfänger erkannt worden ist [NiM93].

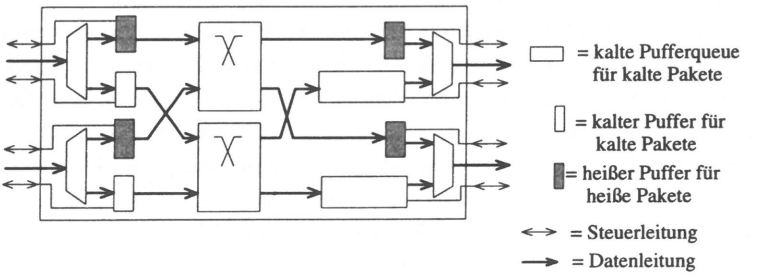


Abbildung 6.40: *Prioritätskoppelelementarchitektur.*

Verglichen mit konventionellen ausgangsgepufferten Koppelelementen resultiert die Prioritätskoppelelementarchitektur in einer komplexeren Hardware. Allerdings ist nur eine zusätzliche Verbindungsleitung pro Koppelelementport notwendig (die Steuerleitung für die heißen Pakete), so daß die Systemkomplexität nur geringfügig ansteigt. Da auch bei dieser Koppelelementarchitektur sich heiße Pakete bis in die Quellen zurückstauen können, müssen getrennte Puffer für heiße und kalte Pakete in den Quellen vorhanden sein, um gegenseitige Beeinflussungen beider Paketklassen zu vermeiden (siehe Abbildung 6.39).

Unter gleichverteiltem Verkehr ist diese Koppelelementarchitektur äquivalent zu einem ausgangsgepufferten Koppelelement mit Eingangslatches, da bei diesen Verkehren nur die kalten Puffer benutzt werden. Unter ungleichverteilten Verkehren stellt diese Koppelelementarchitektur sicher, daß heiße Pakete höchstens einen Pufferplatz pro Koppelelementport belegen können und daß heiße Pakete nie kalte Pakete blockieren können, da diese nie hinter heißen Paketen gepuffert werden. Allerdings führt das gemeinsame Benutzen der Verbindungslinks zwischen Koppelelementen von beiden Paketklassen zu Hot-Spoteffekten, die, wie in den nächsten Abschnitten gezeigt, durch geeignete Transferprioritätsmechanismen gesteuert werden können.

6.4.2 Transferprioritätsmechanismen und deren Leistungsfähigkeit in Parallelrechnernetzen

An einem Ausgangsport des Prioritätskoppelelements konkurrieren zwei Puffer, die kalte Pufferqueue und der heiße Puffer, um den Ausgangslink. Deshalb muß ein Transferprioritätsmechanismus in jedem Netzzyklus bestimmen, welcher Puffer ein Paket über den Ausgangslink senden darf. In den nächsten Abschnitten werden drei Transfermechanismen (die niedrige, hohe und alternierende Transferpriorität) vorgestellt und deren Fähigkeit zur Steuerung von Sättigungsbaumeffekten untersucht und verglichen.

6.4.2.1 Niedrige Transferpriorität der heißen Pakete

In diesem Abschnitt werden Koppelelemente angenommen, in denen an jedem Ausgangsport die heißen Pakete eine **niedrigere Transferpriorität** haben als die kalten Pakete. Man betrachte den Transfer eines Pakets von einem Quellkoppelelement zu einem Zielelement in der nächsten Netzstufe. In solchen Koppelelementen wird ein heißes Paket nur dann vom Ausgangsport des Quellelements über den Verbindungslink zum Eingangsport des Zielelements transferiert, wenn 1) im heißen Puffer des Zielports ein Pufferplatz frei ist, und 2) Pakete im kalten Quellpuffer nicht transferiert werden können, entweder weil dieser Puffer leer ist, oder weil der korrespondierende kalte Puffer des Zielports voll ist. Der Algorithmus zur Auswahl und Weiterleitung eines Pakets an einem Koppelelementausgangsport in jedem Netzzyklus in Koppelelementen mit niedriger Transferpriorität der heißen Pakete lautet demnach:

```
für jeden Ausgangsport eines Koppelelements in jedem Netzzyklus
{
if ((Paket im kalten Puffer) und (kein Paket im heißen Puffer) und (kalter
      Zielpuffer leer))
            then transferiere Paket aus kaltem Puffer;

else if ((kein Paket im kalten Puffer) und (Paket im heißen Puffer) und
          (heißer Zielpuffer leer))
            then transferiere Paket aus heißem Puffer;

else if ((Paket im kalten Puffer) und (Paket im heißen Puffer)){
            if (kalter Zielpuffer leer)
                  then transferiere Paket aus kaltem Puffer;

            else if (heißer Zielpuffer leer)
                  then transferiere Paket aus heißem Puffer;
            }

else idle;
}.
```

Da die Pakete in der kalten Pufferqueue eines Koppelelementausgangs Priorität gegenüber den heißen Paketen haben, können heiße Pakete die kalten Pakete nicht beeinflussen, und ein Sättigungsbaum voller kalter Puffer kann nicht entstehen. So kann dieser Prioritätsmechanismus den leistungsmindernden Einfluß der heißen Pakete auf die kalten eliminieren. Jedoch kann es,

besonders bei höheren Raten des gleichverteilten Verkehrs, dazu kommen, daß jede kalte Pufferqueue ein oder mehrere Pakete pro Netzzyklus gepuffert hat, wie in Abbildung 6.41 gezeigt. Da diese kalten Pakete gegenüber den heißen Paketen priorisiert sind, verhungern die heißen Pakete temporär im Netz, was zu einer in manchen Fällen unerwünscht langen Hot-Spot-Phase führt.

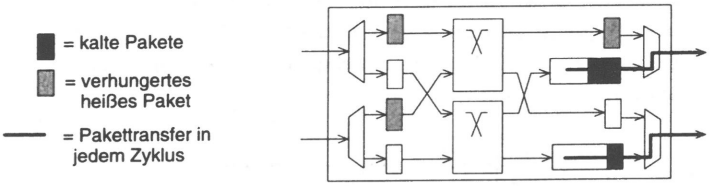


Abbildung 6.41: *Verhungern von heißen Paketen im Koppelelement mit niedriger Transferpriorität der heißen Pakete.*

6.4.2.2 Hohe Transferpriorität der heißen Pakete

In diesem Abschnitt werden Koppelelemente angenommen, in denen an jedem Ausgangsport die heißen Pakete eine **höhere Transferpriorität** haben als die kalten Pakete. In solchen Koppelelementen wird ein kaltes Paket nur dann vom Ausgangsport des Quellelements über den Verbindungslink zum Eingangsport des Zielelements transferiert, wenn 1) im kalten Puffer des Zielports ein Pufferplatz frei ist, und 2) Pakete im heißen Quellpuffer nicht transferiert werden können, entweder weil dieser Puffer leer ist, oder weil der korrespondierende heiße Puffer des Zielports voll ist. Der Algorithmus zur Auswahl und Weiterleitung eines Pakets an einem Koppelelementausgangsport in jedem Netzzyklus in Koppelelementen mit hoher Transferpriorität der heißen Pakete lautet demnach:

```
für jeden Ausgangsport eines Koppelelements in jedem Netzzyklus
{
if ((Paket im kalten Puffer) und (kein Paket im heißen Puffer) und (kalter
      Zielpuffer leer))
            then transferiere Paket aus kaltem Puffer;

else if ((kein Paket im kalten Puffer) und (Paket im heißen Puffer) und
          (heißer Zielpuffer leer))
            then transferiere Paket aus heißem Puffer;

else if ((Paket im kalten Puffer) und (Paket im heißen Puffer)){
            if (heißer Zielpuffer leer)
                  then transferiere Paket aus heißem Puffer;

            else if (kalter Zielpuffer leer)
                  then transferiere Paket aus kaltem Puffer;
            }

else idle;
}.
```

Man betrachte den Hot-Spot-Ausgang eines indirekten Netzes, aufgebaut aus diesen Koppelelementen, unter Hot-Spot-Verkehr. Da der heiße Ausgangspuffer Priorität über der kalten Pufferqueue hat, werden während der Hot-Spot-Phase nur heiße Pakete zum Hot-Spot transferiert, während die kalten Pakete in der kalten Pufferqueue verbleiben und verhungern, wie in Abbildung 6.42 gezeigt. So baut sich ein Sättigungsbaum voller kalter Puffer und Pufferqueues im Netz auf, was zu einem Leistungsabfall des Netzes während der Hot-Spot-Phase führt. Andererseits werden die heißen Pakete von den kalten nicht beeinflußt, so daß eine minimale Durchlaufzeit der heißen Pakete erreicht wird.

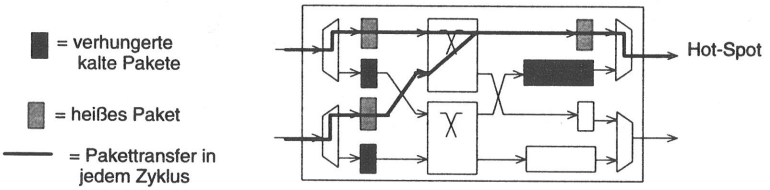


Abbildung 6.42: *Verhungern von kalten Paketen im Koppelelement mit hoher Transferpriorität der heißen Pakete.*

Der Mechanismus der hohen Transferpriorität wurde für ein 1024 × 1024 Delta-Netz mit 2 × 2 Koppelelementen (Gesamtpufferkapazität 20 Pakete) unter temporärem normalverteilten Hot-Spot-Verkehr mit λ=0.5, μ=1000 und σ=50 simuliert. In Abbildung 6.43 sind die zeitlichen Verläufe der Durchlaufzeiten der heißen (Abbildung 6.43a) und der kalten Pakete (Abbildung 6.43b) gezeigt. Während einer Zeitspanne von Zyklus 1300 bis Zyklus 2000 kommen keine kalten Pakete an den Netzausgängen an, da der Sättigungsbaum alle Netzeingänge erreicht hat und dadurch alle kalten Puffer blockiert sind. In diesem Zeitraum verhungern also die kalten Pakete im Netz.

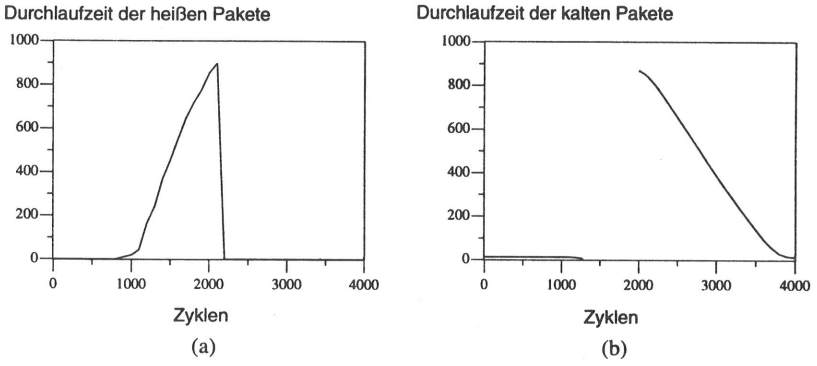


Abbildung 6.43: *Durchlaufzeit der (a) heißen und (b) kalten Pakete in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit hoher Transferpriorität unter Hot-Spot-Verkehr mit λ=0.5, μ=1000 und σ=50; (man beachte, daß von Zyklus 1300 bis Zyklus 2000 keine kalten Pakete an den Netzausgängen ankommen).*

6.4.2.3 *Alternierende Transferpriorität der heißen Pakete*

Nun werden Koppelelemente betrachtet, in denen an jedem Ausgangsport die heißen Pakete eine **alternierende Transferpriorität** bezüglich der kalten Pakete haben. In diesen Koppelelementen wird ein heißes Paket zum Transfer ausgewählt, wenn mindestens K kalte Pakete nach dem letzten Transfer eines heißen Pakets über den Ausgangslink transferiert wurden, vorausgesetzt, daß Pakete in den heißen und kalten Puffern eines Koppelelementausgangsports vorhanden sind. Man betrachte zum Beispiel den heißen Netzausgang unter einem Hot-Spot-Verkehr. Dann werden K kalte, ein heißes, K kalte, ein heißes Paket, usw. zum Hot-Spot transferiert, wenn heiße und kalte Pakete am heißen Netzausgang gepuffert sind. Sind nur Pakete in einer der beiden Puffer eines Koppelelementausgangsports vorhanden, so wird versucht, diese Pakete weiterzuleiten. Dieser Mechanismus kann durch die Verwendung eines zusätzlichen K-Zählers an jedem Koppelelementausgang implementiert werden. Der Algorithmus zur Auswahl und Weiterleitung eines Pakets an einem Koppelelementausgangsport in jedem Netzzyklus in Koppelelementen mit alternierender Transferpriorität der heißen Pakete lautet demnach:

```
für jeden Ausgangsport eines Koppelelements in jedem Netzzyklus
{

if ((Paket im kalten Puffer) und (kein Paket im heißen Puffer) und (kalter
     Zielpuffer leer))
            then transferiere Paket aus kaltem Puffer;

else if ((kein Paket im kalten Puffer) und (Paket im heißen Puffer) und
          (heißer Zielpuffer leer))
            then transferiere Paket aus heißem Puffer;

else if ((Paket im kalten Puffer) und (Paket im heißen Puffer)){
            if ((heißer Zielpuffer leer) und (kalter Zielpuffer voll)
                  then transferiere Paket aus heißem Puffer;
            else if ((heißer Zielpuffer voll) und (kalter Zielpuffer leer)
                  then transferiere Paket aus kaltem Puffer;
            else if ((heißer Zielpuffer leer) und (kalter Zielpuffer leer){
                  if (Zähler < K) then {
                        transferiere Paket aus kaltem Puffer;
                        Zähler=Zähler+1;
                        }
                  else {
                        transferiere Paket aus heißem Puffer;
                        Zähler=0;
                        }
                  }
            }

else idle;
}.
```

Der Zähler wird hierbei inkrementiert, wenn ein kaltes Paket den Port verläßt, und er wird zurückgesetzt, wenn ein heißes Paket den Port verläßt, vorausgesetzt, es sind Pakete in beiden Puffern eines Ausgangsports vorhanden, die nicht blockiert sind. Ein heißes Paket kann einen Ausgangsport nur dann verlassen, wenn entweder die kalte Pufferqueue des Ports leer ist, das Kopfpaket der kalten Pufferqueue blockiert ist (der kalte Eingangspuffer des Empfangselements in der nächsten Stufe also voll ist) oder der Zähler des Ausgangsports den Wert K hat. Analog dazu kann ein kaltes Paket einen Ausgangsport nur dann verlassen, wenn entweder der heiße Puffer des Ports leer ist, das Kopfpaket des heißen Puffers blockiert ist (der heiße Eingangspuffer des Empfangselements in der nächsten Stufe voll ist) oder der Wert des Zählers am Ausgangsport kleiner als K ist. Dieser Mechanismus wird auch in den Quell-PEs angewandt, um einen der Quellpuffer während eines Netzzyklus auszuwählen (siehe Abbildung 6.39).

Der Mechanismus der alternierenden Transferpriorität wurde für ein 1024×1024 Delta-Netz mit 2×2 Koppelelementen (Gesamtpufferkapazität 20 Pakete) unter temporärem normalverteilten Hot-Spot-Verkehr mit λ=0.7, μ=1000 und σ=50 simuliert. In Abbildung 6.44 sind die zeitlichen Verläufe der Durchlaufzeiten der heißen (Abbildung 6.44a) und der kalten Pakete (Abbildung 6.44b) für K=1, 2 und 3 gezeigt. Auf der einen Seite führt eine Erhöhung von K zu einer Verringerung der Durchlaufzeit der kalten Pakete und der Länge der Überlastphase, und auf der anderen Seite zu einer verlängerten Hot-Spot-Phase. Auch ist der alternierende Prioritätsmechanismus für K=3 und $\lambda \leq 0.7$ in der Lage, Sättigungsbaumeffekte vollständig zu unterdrücken. Diese Effekte werden nun im folgenden erklärt.

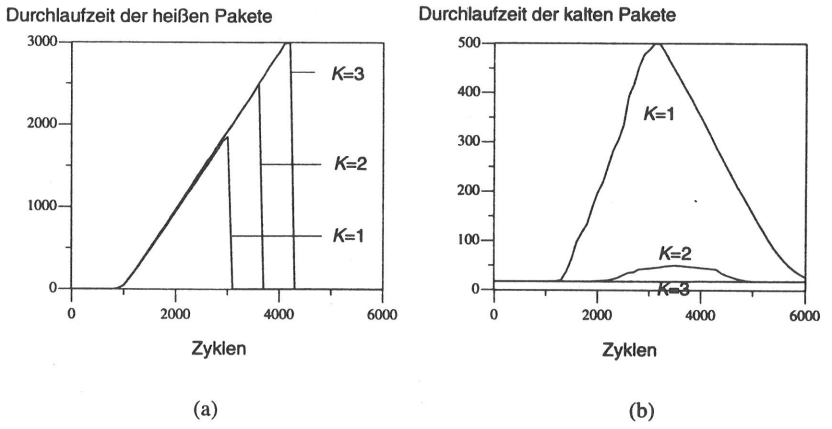


Abbildung 6.44: *Durchlaufzeit der (a) heißen und (b) kalten Pakete in einem 1024×1024 Netz mit 2×2 Koppelelementen mit alternierender Transferpriorität unter Hot-Spot-Verkehr mit λ=0.7, μ=1000 und σ=50.*

Wird eine maximale relative Linkbandbreite von 1 angenommen, so wird, unter der Bedingung, daß Pakete in beiden Puffern eines Ausgangsport vorhanden sind, durch den Prioritätsmechanismus den heißen Paketen eine Bandbreite von $1/(K+1)$ des Ausgangsportlinks zugewiesen, während die kalten Pakete eine Bandbreite von $K/(K+1)$ erhalten. Diese fest zugeteilten Bandbreiten für beide Paketklassen für $0 < K < \infty$ garantieren, daß weder kalte noch heiße Pakete verhungern können. Durch diese Bandbreitenzuteilung werden im Durchschnitt $K/(K+1)$ kalte Pakete pro Netzzyklus während der Hot-Spot-Phase über einen Ausgangsportlink transportiert. Ein Sättigungsbaum wird sich demnach nicht aufbauen, solange die angebotene Rate des gleichverteilten Verkehrs kleiner als λ_{max} ist, mit

$$\lambda_{max} = \frac{K}{K+1} \quad . \tag{29}$$

Die Abhängigkeit der maximalen Verkehrsrate, unter der sich noch keine Sättigungsbäume aufbauen können, von dem Prioritätsparameter K ist in Abbildung 6.45 gezeigt.

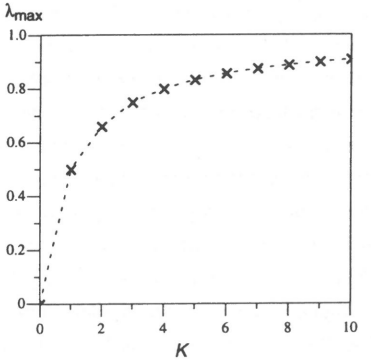


Abbildung 6.45: *Maximale Verkehrsrate, für die noch keine Sättigungsbäume entstehen.*

Wird eine größere Rate als λ_{max} dem Netz angeboten, so muß der Parameter K erhöht werden, um Sättigungsbäume zu vermeiden. Jedoch resultiert die Erhöhung von K in einer Verlängerung der Hot-Spot-Phase, da im schlechtesten Fall ein heißes Paket nur in jedem $(K+1)$ten Zyklus weitertransportiert wird. So führt die Wahl von K zu einem Kompromiß zwischen der maximalen Verkehrsrate, unter der sich keine Sättigungsbäume bilden, und der Länge der Hot-Spot-Phase. Für $K \rightarrow \infty$ ist die alternierende Transferpriorität äquivalent zur niedrigen Transferpriorität aus Abschnitt 6.4.2.1, während für $K=0$ ist die alternierende Transferpriorität äquivalent zur hohen Transferpriorität aus Abschnitt 6.4.2.2 ist.

Abbildungen 6.46 und 6.47 illustrieren den Einfluß der Verkehrsrate und des Parameters K auf die Leistung eines 1024×1024 Delta-Netzes mit 2×2 Koppelelementen (alternierende Priorität, Gesamtpufferkapazität 20 Pakete) unter temporärem normalverteilten Hot-Spot-Verkehr mit $\mu=1000$ und $\sigma=50$. Die Längen der Hot-Spot-Phase (Abbildung 6.46a) und der Überlastphase

(Abbildung 6.46b), sowie die Durchlaufzeiten der kalten Pakete (Abbildung 6.47a) und der kalt-heißen Pakete (Abbildung 6.47b) sind dargestellt (kalt-heiße Pakete sind kalte Pakete, die zum Hot-Spot geroutet werden). Auch die Phasenlängen und Paketdurchlaufzeiten in einem Netz mit konventionellen ausgangsgepufferten Koppelelementen mit Eingangslatches (Abbildung 3.7), in einem Netz mit Koppelelementen niedrigerer Transferpriorität ($K=\infty$) und in einem Netz mit Koppelelementen hoher Transferpriorität ($K=0$) sind zum Vergleich gezeigt. Die Netze wurden bis zu einer Verkehrsrate von $\lambda=0.8$ simuliert, da höhere Raten von den Netzen wegen der begrenzten Pufferlängen in den Koppelelementen nicht mehr bewältigt werden konnten.

Eine Erhöhung von K resultiert in einer Erhöhung der Länge der Hot-Spot-Phase und gleichzeitig in einer Verkleinerung der Durchlaufzeit der kalten und kalt-heißen Pakete. Für $K>0$ kann sich ein Sättigungsbaum nur bei Verkehren mit $\lambda>(K/K+1)$ aufbauen, während bei den konventionell ausgangsgepufferten Koppelelementen und im Falle von $K=0$ sich ein Sättigungsbaum schon bei geringen Verkehrsraten ausbildet, wie oben diskutiert. Dies erklärt das unterschiedliche Durchlaufzeitverhalten der ausgangsgepufferten und $K=0$ Netze, verglichen mit den Durchlaufzeiten aller anderen in Abbildung 6.47 gezeigten Netze. Für $K=0$ erhält man eine minimale Länge der Hot-Spot-Phase, aber eine maximale Länge der Überlastphase, verbunden mit einer maximalen Durchlaufzeit der kalten Pakete. Für Verkehrsraten kleiner als ca. 0.35 zeigt das Netz mit ausgangsgepufferten Koppelelementen jedoch eine etwas größere Durchlaufzeit der kalten Pakete als das $K=0$ Netz. In dem ausgangsgepufferten Netz werden die heißen Pakete zusammen mit den kalten Paketen in den Koppelelementpuffern gepuffert, während sie im $K=0$ Netz getrennt gepuffert sind. Deshalb erstreckt sich der Sättigungsbaum bei kleinen Verkehrsraten über weniger Stufen im $K=0$ Netz, verglichen mit dem ausgangsgepufferten Netz, so daß weniger kalte Pakete durch den Sättigungsbaum verzögert werden. Sobald der Baum die Netzeingänge erreicht hat, zeigt das $K=0$ Netz eine schlechtere Leistung als das ausgangsgepufferte Netz, da die heißen Pakete gegenüber den kalten Paketen priorisiert sind, wie im Abschnitt 6.4.2.2 näher erläutert worden ist.

Für $K=5$ und den gezeigten Bereich der Verkehrsraten wird die Bildung eines Sättigungsbaums vollständig unterdrückt, so daß die Durchlaufzeit der kalten Pakete minimal ist. Der leichte Anstieg der Durchlaufzeit der kalten Pakete im Bereich der Verkehrsrate von $0.7 \leq \lambda \leq 0.8$ im $K=5$ Netz resultiert nicht aus der Blockierung von kalten Paketen durch heiße, sondern durch die begrenzte Länge der Koppelelementpuffer, die zu einer höheren Blockierungswahrscheinlichkeit der kalten Pakete im Netz bei solchen hohen Raten führt [HlK88] (man beachte, daß für $K=5$ ein Sättigungsbaum erst entsteht, wenn $\lambda > K/(K+1)=5/6=0.833$ ist). Sogar die Durchlaufzeit der kalt-heißen Pakete kann in einem weiten Bereich der Verkehrsrate minimiert werden, wie aus Abbildung 6.47(b) ersichtlich ist. Desweiteren kann die Länge der Überlastphase bei einer Verkehrsrate von zum Beispiel $\lambda=0.8$ von 21000 Zyklen beim ausgangsgepufferten Netz auf 6000 Zyklen beim $K=5$ Netz reduziert werden (siehe Abbildung 6.46), welches einem signifikanten Leistungsgewinn entspricht.

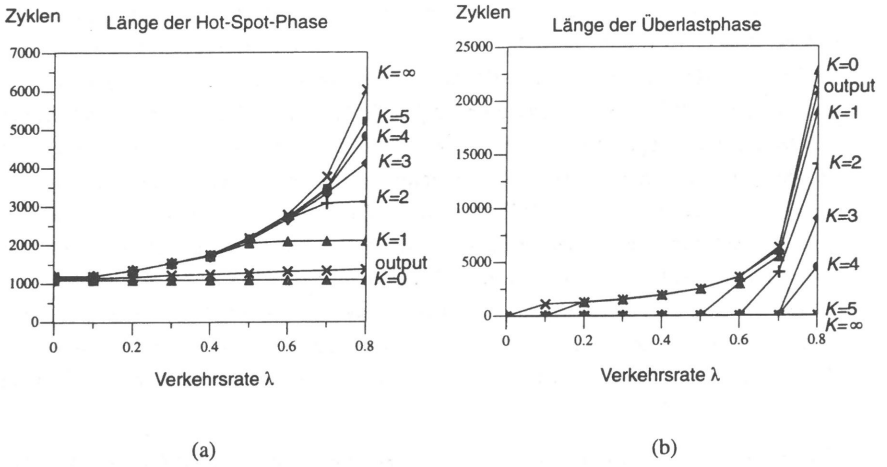


Abbildung 6.46: *Länge der (a) Hot-Spot- und (b) Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit alternierender Transferpriorität unter Hot-Spot-Verkehr mit μ=1000 und σ=50.*

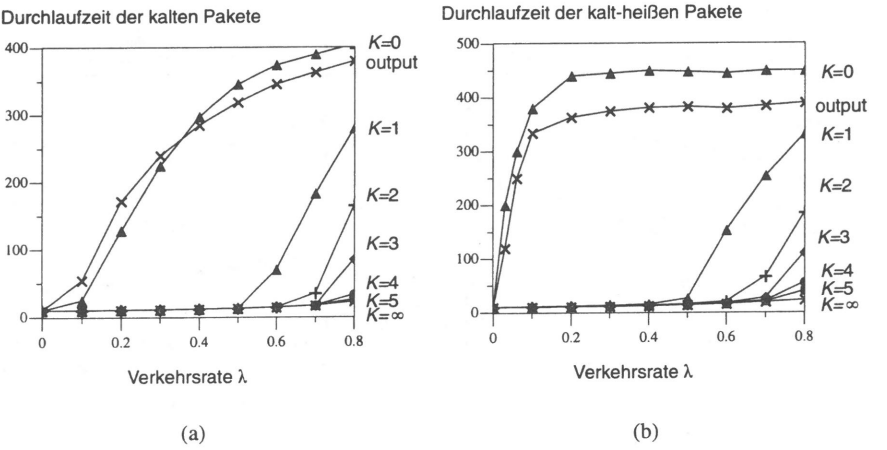


Abbildung 6.47: *Durchlaufzeit der (a) kalten und (b) kalt-heißen Pakete während der Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit alternierender Transferpriorität unter Hot-Spot-Verkehr mit μ=1000 und σ=50.*

Der Einfluß von verschiedenen Hot-Spot-Verkehrsintensitäten (Standardabweichung σ der Verteilung der heißen Pakete) auf die Phasenlängen ist in Abbildung 6.48, und auf die Paketdurchlaufzeiten in Abbildung 6.49 gezeigt. Hierbei ist ein ausgangsgepuffertes Netz einem K=3 Netz bei den Raten λ=0.5 und λ=0.7 gegenübergestellt. Die Abbildungen verdeutlichen, daß das Netz mit den Prioritätskoppelelementen in der Lage ist, die Netzleistung in einem weiten Bereich der Hot-Spot-Intensität zu verbessern. Auch erfahren die kalt-heißen Pakete im Prioritätsnetz nur eine Durchlaufzeit von maximal 29 Zyklen, während sie um bis zu 445 Zyklen im ausgangsgepufferten Netz verzögert werden. Die Simulationen zeigen, daß für ein größer werdendes σ sich die Leistungen der beiden Netze angleichen. Dies ist darauf zurückzuführen, daß bei steigendem σ die Verteilung der heißen Pakete sich immer mehr dem gleichverteilten Verkehr angleichen, so daß bei genügend großem σ die heißen Pakete nicht mehr von den Kalten unterscheidbar sind. Die Netze werden dann unter quasi-gleichverteiltem Verkehr betrieben und weisen dabei eine vergleichbare Leistung auf.

Simulationen von Netzen mit alternierender Transferpriorität unter anderen ungleichverteilten Verkehren, bei denen Sättigungsbäume entstehen können (z. B. Bit-Reverse-Permutationsverkehre), zeigen eine vergleichbare Leistungssteigerung. Netze, aufgebaut aus Koppelelementen mit alternierender Transferpriorität, können also Sättigungsbaumeffekte minimieren, während die Leistung unter gleichverteiltem Verkehr nicht gemindert wird.

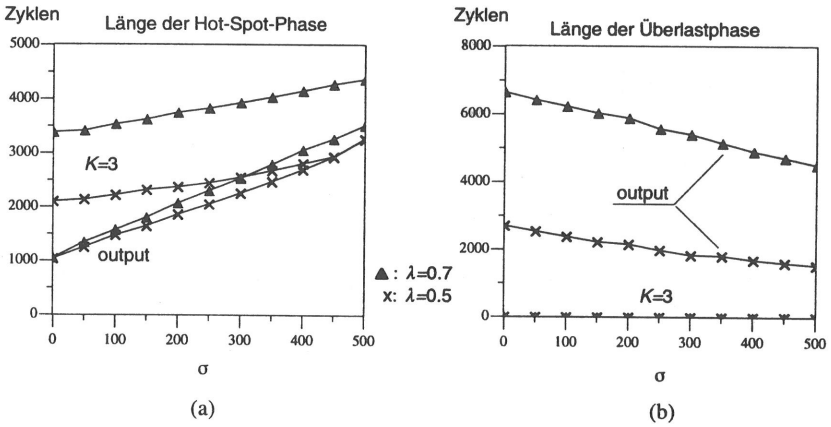


Abbildung 6.48: *Länge der (a) Hot-Spot- und (b) Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit alternierender Transferpriorität unter Hot-Spot-Verkehr mit μ=1000.*

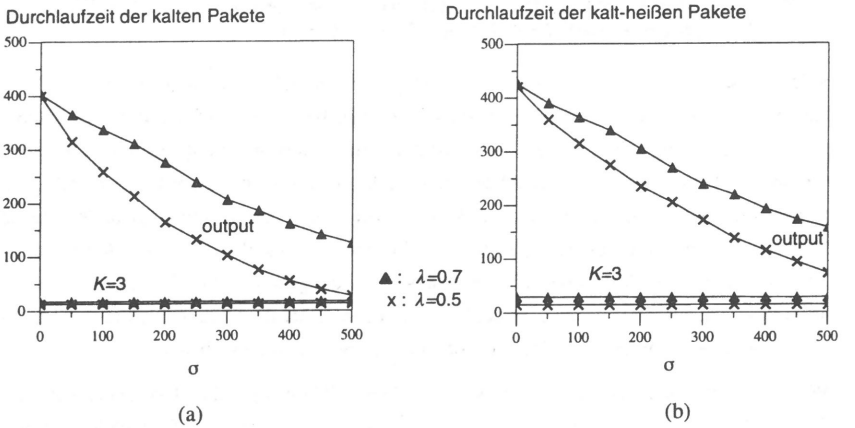


Abbildung 6.49: *Durchlaufzeit der (a) kalten und (b) kalt-heißen Pakete während der Überlastphase in einem 1024 × 1024 Netz mit 2 × 2 Koppelelementen mit alternierender Transferpriorität unter Hot-Spot-Verkehr mit μ=1000.*

6.4.2.4 Adaptive Alternierende Transferpriorität der heißen Pakete

In einem breiten Spektrum von Applikationen sind eine kurze Überlastphase und eine kleine Durchlaufzeit des gleichverteilten Hintergrundverkehrs wichtiger, als eine verlängerte Hot-Spot-Phase. Allerdings kann für manche Applikationen eine möglichst kurze Hot-Spot-Phase mit der damit verbundenen höheren Durchlaufzeit des gleichverteilten Verkehrs günstiger sein. Um die Anforderungen dieser Applikationen ebenfalls zu berücksichtigen, kann ein adaptiver alternierender Transferprioritätsmechanismus eingesetzt werden. Hierbei wird angenommen, daß der Prioritätsparameter in allen Koppelelementen eines Netzes zentral vom Parallelrechner eingestellt und während der Ausführung eines Applikationsprogramms verändert werden kann. Vom Parallelrechnerbenutzer und/oder einem intelligenten Compiler wird entschieden, ob eine minimale Länge der Hot-Spot oder der Überlastphase für Hot-Spot-produzierende Programmsegmente leistungsbestimmender ist. Das Parallelrechnernetz kann dann über den Prioritätsparameter auf diese wechselnden Anforderungen adaptiv während eines Programmlaufs eingestellt werden. Um eine minimale Länge der Hot-Spot-Phase eines Hot-Spot-produzierenden Programmsegments zu erreichen, muß der Prioritätsparameter K hierbei zu 0 gesetzt werden. Soll die Bildung von Sättigungsbäumen und die damit verbundene hohe Durchlaufzeit des gleichverteilten Verkehrs vermieden werden, so muß ein Parameter mit dem Wert $K=\lceil\lambda/(1-\lambda)\rceil$ gewählt werden.

6.4.3 Leistungsfähigkeit der Prioritätskoppelelementarchitektur in ATM-Netzen unter MPEG-codierten Videoströmen

Videodienste in Kommunikationssystemen stellen besondere Anforderungen an das ATM-Verbindungsnetz. Die vor der Verbindungsannahme ausgehandelten QoS-Parameter müssen in engen Grenzen eingehalten werden, wobei die Zellverlustwahrscheinlichkeit, die Zelldurchlaufzeit und der Durchlaufzeitjitter eine besondere Rolle spielen. Zu hohe Zellverluste oder ein zu hoher Durchlaufzeitjitter während einer Videoübertragung resultieren in einer in den meisten Fällen nicht mehr tolerierbaren Qualitätsverschlechterung des Videodienstes. Auch darf die Gesamtdurchlaufzeit der Videozellen durch das Verbindungsnetz in manchen Fällen ein Maximum nicht überschreiten, wenn z. B. der Videostrom von einer Realzeitapplikation erzeugt wird (die Videodaten kommen zu spät am Empfänger an und sind unbrauchbar).

Wenn Datenströme durch ein mehrstufiges indirektes ATM-Netz geschickt werden, können sich die einzelnen Verbindungen gegenseitig beeinflussen, da sie gemeinsame Koppelelementpuffer und Verbindungslinks innerhalb des Netzes benutzen. So müssen unter Umständen Verbindungsanfragen anderer Datenströme abgewiesen werden, da diese die vorhandenen Videoverbindungen negativ beeinflussen könnten, so daß die ausgehandelten QoS-Parameter der Verbindungen dann nicht mehr garantiert werden können.

Eine Möglichkeit die Qualitätsparameter einer Videoübertragung einhalten zu können, ohne andere Verbindungen abzuweisen, ist die priorisierte Übertragung des Videostroms durch das ATM-Netz. Dies kann z. B. durch den Einsatz von ATM-Koppelelementen mit Prioritätsmechanismen, wie in Abschnitt 3.5 vorgestellt, geschehen. Auch kann die in Abschnitt 6.4.1 für Parallelrechnernetze eingeführte Prioritätskoppelelementarchitektur eingesetzt werden. Die Videozellen werden hierbei als heiße Zellen deklariert und die hohe Transferpriorität der heißen Zellen wird benutzt (K=0). Wie in Abschnitt 6.4.2.2 beschrieben, wird so erreicht, daß die Videozellen nicht durch den restlichen Verkehr beeinflußt werden können. ATM-Netze sind in den meisten Fällen aus indirekten Drop-Netzen aufgebaut. Um das Prioritätskoppelelement zu benutzen, muß jedoch ein Block-Netz benutzt werden. In diesem Netz entstehten keine Datenverluste, jedoch müssen Zellen, die nicht in das Netz gelangen können, an den Eingängen des Netzes temporär in Puffern zwischengepuffert werden. Diese Puffer haben eine begrenzte Länge, so daß bei einem Überlauf dieser Puffer dann Zellverluste entstehen.

Zum Leistungsvergleich sind in Abbildung 6.50 die Gesamtverlustwahrscheinlichkeiten der MPEG-Zellen in Abhängigkeit von der Pufferlänge von einem paketsynchronen Drop-Netz mit konventionell ausgangsgepufferten 2×2 ATM-Koppelelementen, einem paketsynchronen Drop-Netz mit partial buffer sharing 2×2 ATM-Koppelelementen (die Videozellen dürfen den gesamten Pufferplatz benutzten, während alle anderen Zellen nur D_2=2 Plätze pro Puffer erhalten, siehe Abschnitt 3.5.2) und einem paketsynchronen Block-Netz mit 2×2 Prioritätskoppelelementen gezeigt. Bei den ATM-Koppelelementen mit Prioritätsmechanismus wird das Schwellwertverfahren und nicht das Verdrängungssystem betrachtet, da, wie in Abschnitt 3.5.2 näher

ausgeführt, das Schwellwertverfahren bei vergleichbarer Leistungsfähigkeit viel weniger Hardware benötigt als das Verdrängungssystem.

Es wurden 32 × 32 Delta-Netze simuliert, wobei an jedem Netzeingang ein Videostrom eingespeist wird, der für die Senke 0 bestimmt ist. Dieser Verkehr resultiert aus 32 unabhängigen Videoverbindungen, die zwischen den Netzeingängen und dem Netzausgang 0 bestehen. Als Videostrom wurde hierbei das Starwars-Video mit den in Abbildung 4.4 aufgeführten statistischen Bildgrößen verwendet. Es sind eine **Netzlinkbandbreite** von L=80Mbit/s, eine Videobildrate von 40 ms und eine Zellgröße von 53 Byte mit 47 Daten- und 5 Steuerbytes angenommen worden. Dies führt zu einer Videorate an der Senke 0 von λ_V=0.8175. Die gezeigte Pufferlänge bezieht sich beim Prioritätsnetz auf die heißen Puffer an den Eingängen des Netzes, während bei den ausgangsgepufferten und partial buffer sharing Netzen die heißen Puffer im Netz betrachtet wurden. Wie in Abbildung 6.50 zu erkennen ist, haben die Videozellen in den ausgangsgepufferten und den partial buffer sharing Netzen die gleiche Zellverlustrate, da unter reinem Videoverkehr beide Netze identisch sind. Verglichen mit diesen Netzen, kann im Prioritätsnetz eine um eine Größenordnung niedrigere Zellverlustrate erreicht werden. Bei der gezeigten Videorate erzeugt ein Prioritätsnetz mit Puffern der Länge 2 an den Netzeingängen eine vergleichbare Zellverlustwahrscheinlichkeit, wie ein ausgangsgepuffertes Netz mit 100 Puffern pro Koppelelementausgangsport.

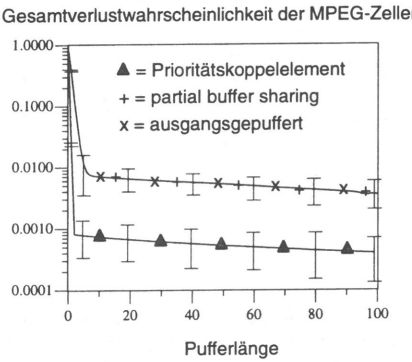


Abbildung 6.50: *Gesamtverlustwahrscheinlichkeit der MPEG-Zellen in einem 32 × 32 ATM-Netz mit 2 × 2 Koppelelementen unter MPEG-codiertem Starwars-Videoverkehr; (alle 32 Quellen senden Videodaten zu einer Senke; L=80Mbit/s; Videorate 0.8175); C=40; C_2=2.*

Nachdem die Leistung der ATM-Netze unter reinem Videoverkehr betrachtet worden ist, wird nun der Einfluß eines gleichverteilten Hintergrundverkehrs auf eine Videoverbindung unter-

sucht. Es werden paketsynchrone 64 × 64 Netze betrachtet, die eine Netzlinkbandbreite von L=20Mbit/s aufweisen und aus 2 × 2 Koppelelementen aufgebaut sind. Um die Leistungsfähigkeiten der Netze vergleichen zu können, wird von einer Gesamtpufferkapazität von 40 Zellen pro Koppelelement ausgegangen. So werden in den ausgangsgepufferten und den partial buffer sharing Koppelelementen 20 Zellen pro FIFO-Puffer benutzt, während in den Prioritätskoppelelementen die FIFO-Puffer Platz für nur jeweils 17 Zellen haben (zusätzlich werden pro Eingangsport 2 Latches und pro Ausgangsport 1 Latch verwendet (siehe Abbildung 6.40)). Es wird eine Videoverbindung angenommen, bei der Videozellen von Quelle 0 zu Senke 0 übertragen wird, so daß eine Videorate von λ_v=0.1 resultiert. Zusätzlich produziert jede Quelle einen gleichverteilten Hintergrundverkehr mit der Hintergrundverkehrsrate λ. In Abbildung 6.51 ist die Durchlaufzeit und der Variationskoeffizient dieser Durchlaufzeit (ein Maß für den Durchlaufzeitjitter) der MPEG-Zellen gezeigt.

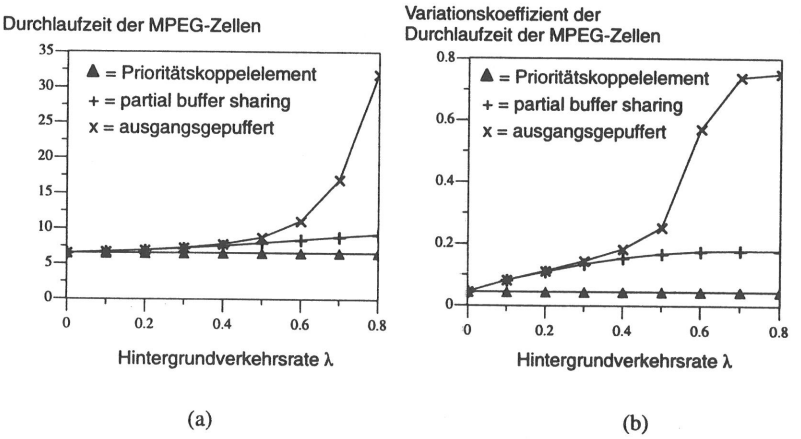


Abbildung 6.51: *Mittlere (a) Durchlaufzeit und (b) Variationskoeffizient der Durchlaufzeit der MPEG-Zellen in einem 64 × 64 ATM-Netz mit 2 × 2 Koppelelementen unter MPEG-codiertem Videoverkehr und gleichverteiltem Hintergrundverkehr (eine Videoquelle; L=20Mbit/s; Videorate 0.1; C=40; D_2=2).*

Wie zu erkennen ist, steigen die Durchlaufzeit und der Jitter der MPEG-Zellen mit steigender Hintergrundrate in den ausgangsgepufferten und partial buffer sharing Netzen an, während sie im Prioritätsnetz konstant bleiben (ein Grundjitter der MPEG-Zellendurchlaufzeit im Prioritätsnetz ist vorhanden, da der asynchrone Videostrom durch den Video-Paketierer an das synchrone ATM-Netz angepaßt werden muß). Da dem gleichverteilten Verkehr im partial buffer sharing Netz nur 2 Zellplätze pro Puffer zustehen, werden die MPEG-Zellen durch die gleichverteilten

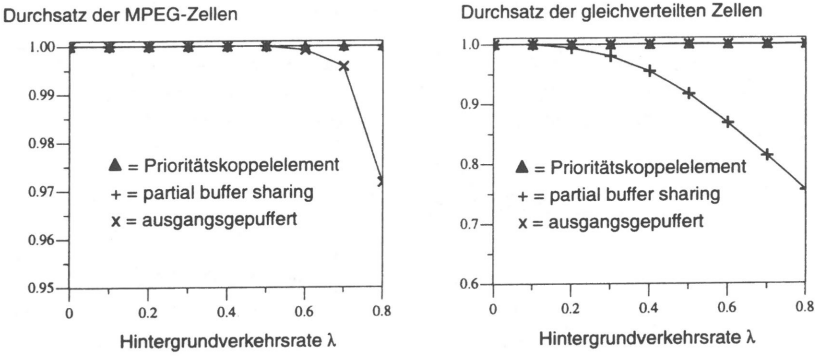


Abbildung 6.52: *Mittlerer relativer Durchsatz der (a) MPEG und (b) gleichverteilten Zellen in einem 64 × 64 ATM-Netz mit 2 × 2 Koppelelementen unter MPEG-codiertem Videoverkehr und gleichverteiltem Hintergrundverkehr (eine Videoquelle; L=20Mbit/s; Videorate 0.1; C=40; D_2=2).*

Zellen in diesem Netz geringer beeinflußt, als im ausgangsgepufferten Netz. Dies wird jedoch mit einem stark reduzierten Durchsatz der gleichverteilten Zellen bezahlt, wie in Abbildung 6.52 zu sehen ist. Hier sind die relativen Durchsätze der MPEG-Zellen (Abbildung 6.52a) und der gleichverteilten Zellen (Abbildung 6.52b) in den drei Netzen gezeigt. Auffallend ist auch, daß der Durchsatz der MPEG-Zellen im ausgangsgepufferten Netz unter einer hohen Verkehrsrate abfällt, während er sich in den anderen Netzen kaum verändert.

In Abbildung 6.53 ist die Durchlaufzeit und der Variationskoeffizient der Durchlaufzeit der gleichverteilten Zellen in den drei Netzen gezeigt. Bei einer Verkehrsrate bis 0.4 werden die gleichverteilten Zellen durch die Videoverbindung nur gering beeinflußt, so daß sich die Durchlaufzeiten und der Jitter der gleichverteilten Zellen in den Netzen gleichen. Bei höheren Verkehrsraten zeichnet sich das partial buffer sharing Netz durch die kleinste Durchlaufzeit und den kleinsten Jitter aus. Dies muß allerdings im Zusammenhang mit dem niedrigen Durchsatz der gleichverteilten Zellen betrachtet werden (siehe Abbildung 6.52b). Bei einer hohen Hintergrundrate beginnt sich der Prioritätsmechanismus im Prioritätsnetz auf den Hintergrundverkehr auszuwirken, so daß ein erhöhter Anstieg der Durchlaufzeit und des Jitters der gleichverteilten Zellen im Prioritätsnetz verglichen zu den ausgangsgepufferten und den partial buffer sharing Netzen resultiert.

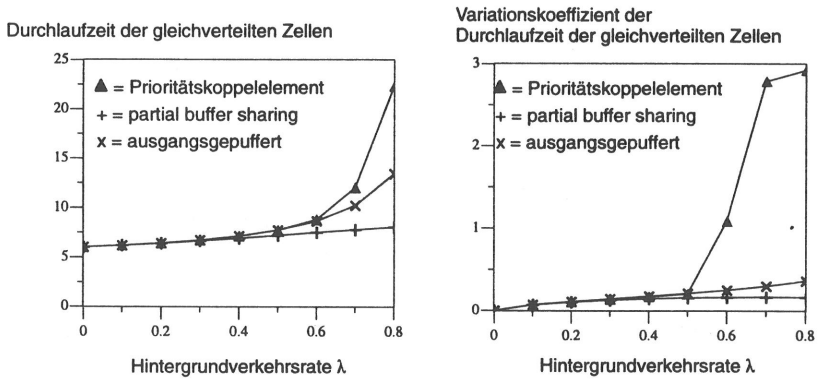


Abbildung 6.53: *Mittlere (a) Durchlaufzeit und (b) Variationskoeffizient der Durchlaufzeit der gleichverteilten Zellen in einem 64 × 64 ATM-Netz mit 2 × 2 Koppelelementen unter MPEG-codiertem Videoverkehr und gleichverteiltem Hintergrundverkehr (eine Videoquelle; L=20Mbit/s; Videorate 0.1; C=40; D_2=2).*

In den ausgangsgepufferten und den partial buffer sharing ATM-Netzen wird die Videoverbindung demnach durch den Hintergrundverkehr beeinflußt, während er im Prioritätsnetz keine Beeinflussung erleidet. Dies wird allerdings im Prioritätsnetz durch eine Verschlechterung der Durchlaufzeit und des Jitters der gleichverteilten Verkehre bei höheren Verkehrsraten erkauft. Im partial buffer sharing Netz kann eine Verringerung der Durchlaufzeit und des Jitters der MPEG-Zellen bei höheren Verkehrsraten nur durch einen massiven Verlust von gleichverteilten Zellen erreicht werden.

Mit der Prioritätskoppelelementarchitektur ist es also möglich, den Einfluß des Hintergrundverkehrs auf den Videostrom zu beseitigen, so daß die Qualitätsparameter einer Videoübertragung garantiert eingehalten werden können. Um eine vorgegebene Zellverlustrate einzuhalten, müssen zusätzliche Puffer an den Eingängen des Prioritätsnetzes vorhanden sein; diese Puffer können jedoch viel kürzer sein als z. B. die in den ausgangsgepufferten Koppelelementen. Das Prioritätsnetz eignet sich besonders gut zur priorisierten Weiterleitung von Realzeit-Videoströmen, da hierbei die Durchlaufzeit wie auch der Durchlaufzeitjitter minimal und die Zellverlustrate gering sein muß, was im Prioritätsnetz gewährleistet ist.

7 Zusammenfassung und Ausblick

In massiv parallelen Rechensystemen ist ein Grundproblem die schnelle und effiziente Kommunikation zwischen den einzelnen Knoten über das Verbindungsnetz. Hiervon wird die Gesamtleistung des Parallelrechners entscheidend beeinflußt. Entwurfsparameter, wie z. B. Netztopologie und Architektur, die leistungsbestimmend für das Netz sind, müssen in der Netzentwurfsphase durch Analyse und/oder Simulation festgelegt werden. Die Wahl der Parameter und das resultierende Netz hängen hierbei stark von den Annahmen über den Verkehrsfluß ab. Netze, die unter gleichverteilten Verkehren gute Leistungsparameter aufweisen, können durch Leistungseinbrüche unter ungleichverteilten Verkehren zum leistungsbegrenzenden Teil des Parallelrechnersystems werden. In dieser Arbeit sind deshalb die Effekte von ungleichverteilten Datenverkehren in indirekten Verbindungsnetzen und deren Auswirkungen auf die Netzleistung untersucht, ein Konzept zur Klassifikation der Verkehrsauswirkungen und der erreichbaren Netzleistung erarbeitet und Methoden zur Leistungsverbesserung von Netzen unter diesen Datenverkehren entwickelt worden.

Zur Bestimmung von Parametern und Leistungseigenschaften von Verbindungsnetzen können Analyse und Simulation herangezogen werden. Da die Analyse von vereinfachten mathematischen Modellen ausgeht, die die realen Netzeigenschaften nicht immer richtig modellieren können, ist für eine realistische Parameterbestimmung und Netzbeurteilung in vielen Fällen die Simulation der Analyse vorzuziehen. Zwei wichtige Aspekte bei einer Simulation sind die Genauigkeit der Ergebnisse und die Simulationszeit. Je genauer die Ergebnisse sein sollen oder je seltener zu simulierende Ereignisse auftreten, desto länger muß simuliert werden. Hier stoßen Simulatoren, die auf einer normalen Workstation implementiert sind, durch die hohen Simulationszeiten an ihre Grenzen. Deshalb wurde in dieser Arbeit als Grundlage zur simulativen Leistungsbestimmung von Verbindungsnetzen und Optimierung von Netzparametern, wie Topologie und Architektur, die parallele Simulation herangezogen, mit deren Leistungsfähigkeit die Bestimmung von Netzcharakteristiken mit vertretbarem Zeitaufwand in vielen Fällen erst ermöglicht wird.

Bei der parallelen Simulation von Verbindungsnetzen existieren viele Möglichkeiten, die Netztopologie und Architektur auf den Parallelrechner abzubilden. Ein Unterscheidungsmerkmal der verschiedenen Abbildungsmöglichkeiten ist hierbei die Parallelität der Abbildung, die in der Anzahl der benutzten Prozessoren pro Simulator gemessen werden kann. Um die Leistung des vorhandenen Parallelrechners vollständig auszunützen, können mehrere Simulatoren parallel auf der Maschine implementiert werden. Mit steigender Parallelität der Abbildung nimmt die Anzahl der parallel laufenden Simulatoren ab. Gleichzeitig nimmt jedoch die Zeit eines Simulationslaufes zu, da bei niedrigerer Parallelität mehr Simulationsschleifen bearbeitet werden müssen. Auch nimmt der Speicherplatzbedarf pro Parallelrechnerknoten ab, während die Kommunikationsanforderungen an den Parallelrechner steigen. Durch diese gegenläufigen Verhalten

läßt sich nicht ohne weiteres feststellen, welche Abbildungsstrategie zu dem schnellsten Simulator führt. Deshalb sind beispielhaft zwei Simulatoren implementiert und Leistungsmessungen vorgenommen worden, um quantitative Vergleiche zu erhalten. Als Netztypen wurden hierbei synchrone Delta-Netze gewählt, welche bis zu 1024 Quellen mit 1024 Senken verbinden können. Diese Netze sind auf den MasPar-Rechner MP-1 abgebildet worden, welcher 16K Prozessoren in SIMD-Architektur besitzt. Es wurden die Linkabbildung (die Zwischenleitungen des Netzes sind auf die Parallelrechnerknoten abgebildet) und die Schalterabbildung (die Koppelelemente sind auf die Rechnerknoten abgebildet) implementiert. Meßergebnisse zeigen, daß die Schalterabbildung deutlich leistungsfähiger als die Linkabbildung ist. Die Schalterabbildung stößt allerdings an ihre Grenzen, wenn z. B. komplexere Koppelelemente mit mehreren parallelen Puffern pro Ein- und/oder Ausgang simuliert werden sollen, da diese, abhängig von der Koppelelementgröße, durch den beschränkten Speicherplatz pro Prozessor des SIMD-Rechners nicht mehr implementiert werden können. Vergleiche von Messungen an einem äquivalenten seriellen Simulator zeigen, daß durch die parallele Simulation eine Leistungssteigerung um einen Faktor von bis zu 150 erreicht werden kann. So ist es möglich, 10^9 Pakete in 40 Minuten zu simulieren. Diese Ergebnisse verdeutlichen den Leistungsgewinn einer parallelen Simulation.

Unter Zuhilfenahme des parallelen Simulators sind Effekte von statisch bzw. temporär ungleichverteilten Datenverkehren in mehrstufigen Delta-Netzen untersucht worden. Unter ungleichverteilten Verkehren, wie z. B. Hot-Spot- und Bit-Reverse-Permutationsverkehre, können sich Sättigungsbäume von gefüllten Koppelelementpuffern in indirekten Netzen aufbauen, die zu signifikanten Leistungseinbrüchen in den Netzen führen. Während unter statisch ungleichverteilten Verkehren die maximal zu erreichende Netzauslastung und der Datendurchsatz abnimmt, und die Paketdurchlaufzeit ansteigt, führen transiente Sättigungsbäume unter temporär ungleichverteilten Verkehren zu temporären Netzüberlasten. Deshalb müssen zusätzliche Leistungsmerkmale in Verbindungsnetzen unter temporär ungleichverteilten Datenverkehren, wie die Längen der in dieser Arbeit eingeführten Hot-Spot- und Überlastphasen, untersucht werden, um die Dynamik dieser Verkehre und deren transiente Auswirkungen im Netz mit zu berücksichtigen.

Es sind verschiedene Methoden zur Verminderung von Sättigungsbaumeffekten in Verbindungsnetzen unter ungleichverteilten Datenverkehren in der Literatur vorgestellt worden. Eine Klasse von Methoden zeichnet sich durch erweiterte Koppelelementarchitekturen aus, wobei durch Verwendung von parallelen logischen Puffern im Netz versucht wird, Datenpakete am Sättigungsbaum vorbeizuführen, so daß weniger Pakete vom Baum beeinflußt werden können. Zur Klassifizierung und analytischen Leistungsbewertung, sowie für die Leistungsoptimierung dieser Netze kann das Modellierungskonzept der HOL-Blockierungen höherer Ordnung verwendet werden, welches in dieser Arbeit eingeführt worden ist. Dieses Konzept klassifiziert Netze nach der minimalen Länge von Wegen durch das Netz, die keine gemeinsamen Puffer besitzen. Die

HOL-Blockierungsordnung H einer Klasse ist hierbei proportional zu dieser minimalen Länge. Innerhalb jeder Klasse weisen Netze gleicher Topologie eine vergleichbare Leistung bei vergleichbarem Hardwareaufwand unter ungleichverteilten Datenverkehren auf, während die Leistung der Netze mit steigender HOL-Blockierungsordnung zunimmt. Diese Klassifikationseigenschaften wurden analytisch und simulativ für mehrstufige Delta-Netze nachgewiesen. In einem Fallbeispiel wurde gezeigt, wie dieses Konzept zur Leistungsoptimierung einer in der Literatur vorgestellten Koppelelementarchitektur benutzt werden kann. Diese Optimierung führt zu einer signifikanten Leistungssteigerung (einer Erhöhung der maximalen Netzauslastung um bis zu 20% mit einer damit verbundenen Verkleinerung der Paketdurchlaufzeiten) von Delta-Netzen aufgebaut aus diesen Koppelelementen unter ungleichverteilten Datenverkehren bei einer gleichzeitigen Verringerung des Hardware-Aufwands.

Leistungsuntersuchungen von Netzen aus verschiedenen HOL-Blockierungsklassen unter temporär ungleichverteilten Datenverkehren zeigen, daß die HOL-Blockierungscharakteristiken dieser Netze einen Einfluß auf die Paketdurchlaufzeit unter diesen Verkehren hat, während jedoch die Längen der Hot-Spot- und Überlastphasen nicht beeinflußt bzw. verändert werden können. Mit der in dieser Arbeit vorgestellten Prioritätskoppelelementarchitektur ist es möglich, das Auftreten von Netzüberlasten und die damit verbundenen Phasenlängen und Paketdurchlaufzeiten unter temporär ungleichverteilten Datenverkehren zu steuern. Diese Architektur zeichnet sich durch eine getrennte Pufferung von gleichverteilten und ungleichverteilten Datenpaketen aus. Ein parametrisierbarer, alternierender Transferprioritätsmechanismus sorgt für eine feste Bandbreitenzuteilung für die einzelnen Paketklassen innerhalb eines Koppelelements. Durch eine geeignete Wahl des Prioritätsparameters kann die gegenseitige Beeinflussung der gleich- und ungleichverteilten Pakete gesteuert werden. Das Setzen des Parameter zu 0 entspricht einer hohen Transferpriorität der ungleichverteilten Pakete gegenüber den gleichverteilten, so daß eine minimale Durchlaufzeit der ungleichverteilten Pakete und eine minimale Länge der Hot-Spot-Phase erreicht wird, während sich ein Sättigungsbaum im Netz aufbaut, der zu hohen Durchlaufzeiten der gleichverteilten Pakete und einer langen Netzüberlastphase führt. Wird ein großer Prioritätsparameter gewählt, so kehrt sich das Netzverhalten um. Das Auftreten von Sättigungsbäumen unter ungleichverteilten Datenverkehren kann so vollständig unterdrückt werden, so daß eine minimale Durchlaufzeit der gleichverteilten Pakete, verbunden mit einer Unterdrückung von Netzüberlasten, erreicht werden kann. Hierbei vergrößern sich jedoch die Durchlaufzeit der ungleichverteilten Pakete und die Länge der Hot-Spot-Phase. Durch eine adaptive Wahl des Prioritätsparameters, zentral durchgeführt durch den Parallelrechner, können die Leistungscharakteristiken des Verbindungsnetzes an die Leistungsanforderungen der auf dem Parallelrechner laufenden Algorithmen und Applikationen angepaßt werden. Untersuchungen haben gezeigt, daß das Prioritätskonzept auch in direkten Gitternetzen unter ungleichverteilten Verkehren erfolgreich eingesetzt werden kann [Kec94].

Das Prioritätskoppelelement kann auch in ATM-Netzen zur priorisierten Weiterleitung von zeitkritischen Datenströmen effizient eingesetzt werden. Dies wurde anhand von MPEG-codierten Videoströmen in ATM-Netzen nachgewiesen. Im konventionellen Netz wird ein Videostrom durch den Hintergrundverkehr verzögert. Mit der Prioritätskoppelelementarchitektur ist es möglich, den Einfluß des Hintergrundverkehrs auf den Videostrom zu beseitigen, so daß die Qualitätsparameter einer Videoübertragung garantiert eingehalten werden können. Dies zeigen durchgeführte Leistungsvergleiche mit ausgangsgepufferten und mit priorisierten ATM-Netzen. Das Prioritätsnetz eignet sich besonders gut zur priorisierten Weiterleitung von Realzeit-Videoströmen, da hierbei die Durchlaufzeit wie auch der Durchlaufzeitjitter minimal und die Zellverlustrate gering sein muß, was im Prioritätsnetz gewährleistet ist.

Zur Leistungsbestimmung von Parallelrechnernetzen werden bisher meist stark vereinfachende Verkehrsmuster gewählt (z. B. gleichförmiger Verkehr, Hot-Spot- und Permutationsverkehre); neue Rechnerorganisationen, Betriebssystemanforderungen (z. B. Multithreading, SPMD-Modus und dynamische Lastbalancierung) und Anwendungsparadigmen führen jedoch zu neuen, hochdynamischen Verkehrsmustern, deren Abstraktion und Modellierung erforderlich ist, um Netzarchitekturen realistisch beurteilen zu können. Deshalb ist es nötig, theoretische Betrachtungen der Auswirkungen von Anwendungsprogrammen und Steuermechanismen in Parallelrechnern auf den Netzverkehr durchzuführen und die resultierenden Verkehre zu abstrahieren und zu modellieren. Durch die Analyse und/oder Simulation von Netzarchitekturen unter diesen Verkehren können dann leistungsfähigere Netze unter den Anforderungen neuer Parallelrechnerarchitekturen, Anwendungsklassen und Betriebssysteme entwickelt werden.

Literaturverzeichnis

[AbP89] S. Abraham and K. Padmanabhan, "Performance of the direct binary n-cube network for multiprocessors," *IEEE Transactions on Computers*, Vol. C-38, No. 7, July 1989, pp. 1000-1011.

[AdS82] G. B. Adams III and H. J. Siegel, "The extra stage cube: A fault-tolerant interconnection network for supersystems," *IEEE Transactions on Computers*, Vol. C-31, No. 5, May 1982, pp. 443-454.

[AdS84a] G. B. Adams III and H. J. Siegel, "A survey of fault-tolerant multistage networks and comparison to the extra stage cube," *Seventeenth Annual Hawaii International Conference on System Sciences*, January 1984, pp. 268-277.

[AdS84b] G. B. Adams III and H. J. Siegel, "Modifications to improve the fault tolerance of the extra stage cube interconnection network," *1984 International Conference on Parallel Processing*, August 1984, pp. 169-173.

[Aga92] A. Agarwal, "Performance tradeoffs in multithreaded processors," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 5, September 1993, pp. 525-539.

[Agr83] D. P. Agrawal, "Graph theoretical analysis and design of multistage interconnection networks," *IEEE Transactions on Computers*, July 1983, pp. 637-648.

[AhD89] H. Ahmadi and W. E. Denzel, "A survey of modern high-performance switching techniques," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 7, September 1989, pp. 1091-1103.

[AkC93] I. F. Akyildiz, L. Chen, S. R. Das, R. M. Fujimoto, and R. F. Serfozo, "The effect of memory capacity on Time Warp performance," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 411-422.

[ArG89] E. Arthurs, M.S. Goodman, M.P. Vecchi, and H. Kobrinski, "HYPASS: An optoelectronic hybrid switching system," *IEEE Transactions on Communications*, Vol. 37, No. 6, June 1899, pp. 645-648.

[AtS88] W. C. Athas and C. L. Seitz, "Multicomputers: Message-passing concurrent computers," *IEEE Computer*, August 1988, pp. 9-24.

[AyB93] R. Ayani and B. Berkman, "Parallel discrete event simulation on SIMD computers," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 501-508.

[BaF91] W. Buchhuber, W. Fuerst, R. Knorr, and T. Wild, "Architektur, Technologie und Realisierung eines ATM-Koppelnetzes," *ITG/GI-Workshop Verbindungsnetzwerke fuer Parallelrechner und Breitband-Uebermittlungssysteme*, September 1991, pp. 48-52.

[BaG91] P. Barri and J.A.O. Goubert, "Implementation of a 16 by 16 switching element for ATM exchange," *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 5, June 1991, pp. 751-757.

[BaM91] H.F. Badran and H.T. Mouftah, "Head of line arbitration in ATM switches with input-output buffering and backpressure control," *IEEE GLOBECOM'91*, Vol. 1, December 1991, pp. 347-351.

[BaS93] H. Bauer and C. Sporrer, "Reducing rollback overhead in Time-Warp based distribted simulation with optimized incremental state saving," *26th Annual Simulation Symposium*, March 1993, pp. 12-20.

[Bat76] K. E. Batcher, "The flip network in STARAN," *1976 International Conference on Parallel Processing*, August 1976, pp. 65-71.

[Bat80] K. E. Batcher, "Design of a massively parallel processor," *IEEE Transactions on Computers*, Vol. C-29, No. 9, September 1980, pp. 836-844.

[Bat82] K. E. Batcher, "Bit serial parallel processing systems," *IEEE Transactions on Computers*, Vol. C-31, No. 5, May 1982, pp. 377-384.

[BeC89] M. A. Beunder, G. Chen, B. Hoefflinger, J. Kernhof, W. Haas, M. Schau, T. Schwederski, and Springer, R, "Advanced semicustom design and fabrication at the Institute for Microelectronics Stuttgart," *UGIM'89*, June 1989.

[BeD85] J. Beetem, M. Denneau, and D. Weingarten, "The GF11 supercomputer," *12th Annual International Symposium on Computer Architecture*, June 1985, pp. 108-115.

[BeH88] M. A. Beunder, B. Hoefflinger, and J. Kernhof, "New directions in semi-custom arrays," *IEEE Journal of Solid-State Circuits*, Vol. 23, No. 3, June 1988, pp. 728-735.

[BeH91] W. Beller, R. Hainbucher, and T. Schwederski, *IMS 1.2um GATE FOREST Zellenkatalog*, IMS-TB-3/91, Institut fuer Mikroelektronik Stuttgart, 1991.

[BeK88] M. A. Beunder, J. Kernhof, and B. Hoefflinger, "The CMOS Gate Forest: An efficient and flexible high-performance ASIC design environment," *IEEE Journal of Solid-State Circuits*, Vol. 23, No. 2, April 1988, pp. 387-399.

[BeS91] E. Bernath, T. Schwederski, M. Siegel, and J. Wenzel, "Effiziente VLSI-Strukturen fuer mehrstufige Verbindungsnetzwerke," *ITG/GI-Workshop Verbindungsnetzwerke fuer Parallelrechner und Breitband-Uebermittlungssysteme*, September 1991, pp. 53-56.

[Ben65] V. E. Beneš, *Mathematical Theory of Connecting Networks and Telephone Traffic*, Academic Press, New York, NY, 1965.

[Bhu85] L. N. Bhuyan, "An analysis of processor-memory interconnection networks," *IEEE Transactions on Computers*, Vol. C-34, No. 3, March 1985, pp. 279-283.

[Bla90] T. Blank, "The MasPar MP-1 architecture," *IEEE Compcon*, February 1990, pp. 20-24.

[BoB87] P. Boyer, J. Boyer, J.-R. Louvion, and L. Romoeuf, "Modelling the ATD transfer technique," *5th ITC Seminar: Traffic Engineering for ISDN Design and Planning*, May 1987, pp. 26.

[BrM85] W. C. Brantley, K. P. McAuliffe, and J. Weiss, "The RP3 processor/memory element," *1985 International Conference on Parallel Processing*, August 1985, pp. 782-789.

[Bro88] R. Brown, "Calendar Queues: A fast O(1) priority queue implementation for the simulation event set problem," *Communications of the ACM*, Vol. 31, No. 10, October 1988, pp. 1220-1225.

[BuT89] R.G. Bubenik and J.S. Turner, "Performance of a broadcast packet switch," *IEEE Transactions on Communications*, Vol. 37, No. 1, January 1989, pp. 60-69.

[ChH91] D.X. Chen and J.F. Hayes, "A shared buffer memory switch with maximum queue and minimum allocation," *Canadian Conference on Electrical and Computer Engineering*, September 1991, pp. 7.1.1-7.1.4.

[ChM79] K. M. Chandy and J. Misra, "Distributed simulation: A case study in design and verification of distributed programs," *IEEE Transactions on Software Engineering*, Vol. SE-5, No. 5, September 1979, pp. 440-452.

[ChM81] K. M. Chandy and J. Misra, "Asynchronous distributed simulation via a sequence of parallel computations," *Communications of the ACM*, Vol. 24, No. 11, April 1981, pp. 198-206.

[Clo53] C. Clos, "A study of non-blocking switching networks," *Bell System Technical Journal*, Vol. 32, No. 2, March 1953, pp. 406-424.

[Cve87] Z. Cvetanovic, "Performance analysis of the FFT algorithm on a shared-memory parallel architecture," *IBM Journal for Research and Development*, Vol. 31, No. 4, July 1987, pp. 435-451.

[DaA93] W. J. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 4, April 1993, pp. 466-480.

[Dai87] J. N. Daigle, "Message delays with prioritized HOLP and round-robin packet servicing," *IEEE Transactions on Communications*, Vol. com-35, No. 6, June 1987, pp. 609-619.

[Dal92] W. J. Dally, "Virtual-channel flow control," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 2, March 1992, pp. 194-205.

[Des91] E. Desmet, "Performance assessment of an ATM self-routing switching network using parallel programming techniques and a network of transputers," *International Computer and Communication Conference*, October 1991.

[DiB94] J. Ding and L. N. Bhuyan, "Finite buffer analysis of multistage interconnection networks," *IEEE Transactions on Computers*, Vol. 43, No. 2, February 1994, pp. 243-247.

[DiJ81] D. M. Dias and J. R. Jump, "Analysis and simulation of buffered delta networks," *IEEE Transactions on Computers*, Vol. C-30, April 1981, pp. 273-282.

[DiK89] D. M. Dias and M. Kumar, "Preventing congestion in multistage networks in the presence of hotspots," *1989 International Conference on Parallel Processing*, Vol. I, August 1989, pp. 9-13.

[Din89] A. Dinning, "A survey of synchronization methods for parallel computers," *IEEE Computer*, Vol. 22, No. 7, July 1989, pp. 66-77.

[DoR91] J. W. Dolter, P. Ramanathan, and K. G. Shin, "Performance analysis of virtual cut-through switching in HARTS: A hexagonal mesh multicomputer," *IEEE Transactions on Computers*, Vol. 40, No. 6, June 1991, pp. 669-680.

[DrG94] J. T. Draper and J. Ghosh, "A comprehensive analytical model for wormhole routing in multicomputer systems," *Journal of Parallel and Distributed Computing*, Vol. 23, No. 2, November 1994, pp. 202-214.

[Dun90] R. Duncan, "A survey of parallel computer architectures," *Computer*, Vol. 23, No. 2, February 1990, pp. 5-16.

[EdG85] J. Edler, A. Gottlieb, C. P. Kruskal, K. P. McAuliffe, L. Rudolph, M. Snir, P. J. Teller, and J. Wilson, "Issues related to MIMD shared-memory computers: The NYU Ultracomputer approach," *12th International Symposium on Computer Architecture*, June 1985, pp. 126-135.

[Ens94] J. Enssle, "Modelling and statistical multiplexing of VBR MPEG compressed video in ATM networks," *4th Open Workshop on High Speed Networks*, September 1994, pp. 59-67.

[Ens95] J. Enssle, "Modelling of short and long term properties of VBR MPEG compressed video in ATM networks," *1995 Silicon Valley Networking Conference*, April 1995, pp. 95-107.

[EsK93] R. Esser and R. Knecht, "Intel Paragon XP/S - Architecture and software environment" in *Supercomputer '93*, H. W. Meurer, ed., Springer-Verlag, 1993.

[Fen81] T. Y. Feng, "A survey of interconnection networks," *Computer*, Vol. 14, No. 12, December 1981, pp. 12-27.

[FiF91] W. Fischer, O. Fundneider, E.H. Goeldner, and K.A. Lutz, "A scalable ATM switching system architecture," *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 8, October 1991, pp. 1299-1307 .

[Fly66] M. J. Flynn, "Very high-speed computing systems," *Proceedings of the IEEE*, Vol. 54, No. 12, December 1966, pp. 1901-1909.

[FuT92] R. M. Fujimoto, J. J. Tsai, and G. C. Gopalakrishnan, "Design and evaluation of the rollback chip: Special purpose hardware for time warp," *IEEE Transactions on Computers*, Vol. 41, No. 1, January 1992, pp. 68-82.

[Fuj90] R. M. Fujimoto, "Parallel discrete event simulation," *Communications of the ACM*, Vol. 33, No. 10, October 1990, pp. 31-53.

[GaG91] H.R. Gail, G. Grover, R. Guerin, and S.L. Hantler, "Buffer size requirements under longest queue first," *International Conference on the Performance of Distributed Systems and Integrated Communication Networks*, September 1991, pp. 383-394.

[GaG93] B. Gaujal, A. G. Greenberg, and D. M. Nicol, "A sweep algorithm for massively parallel simulation of circuit-switched networks," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 484-500.

[GaR90] I. Gard and J. Rooth, "An ATM switch implementation - Technique and technology," *XIII International Switching Symposium*, Vol. IV, May 1990, pp. 23-27.

[GaV91] M. W. Garrett and M. Vetterli, "Congestion control strategies for packet video," *4th International Workshop on Packet Video*, August 1991.

[GeJ91] V. S. Getov and C. R. Jesshope, "Simulation facility of distributed memory system with 'Mad Postman' communication network" in *Distributed Memory Computing,*

Lecture Notes in Computer Science 487, A. Bode, ed., Springer-Verlag, Berlin Heidelberg New York London Paris Tokyo, April 1991, pp. 224-233.

[GhC94] K. Ghose and E.-D. Cheng, "Efficient barrier synchronization techniques and their applications in large-scale shared memory multiprocessors," *Journal of Microcomputer Applications*, Vol. 17, no. 2, April 1994, pp. 197-206.

[GlN92] C. J. Glass and L. M Ni, "Maximally fully adaptive routing in 2D meshes," *1992 International Conference on Parallel Processing*, Vol. I, August 1992, pp. 101-104.

[GlT93] D. W. Glazer and C. Tropper, "On process migration and load balancing in Time Warp," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 3, March 1993, pp. 318-327.

[GoG83] A. Gottlieb, R. Grishman, C. P. Kruskal, K. P. McAuliffe, L. Rudolph, and M. Snir, "The NYU Ultracomputer - Designing an MIMD shared-memory parallel computer," *IEEE Transactions on Computers*, Vol. C-32, No. 2, February 1983, pp. 175-189.

[GoL73] G. R. Goke and G. J. Lipovski, "Banyan networks for partitioning multiprocessor systems," *First Annual Symposium on Computer Architecture*, December 1973, pp. 21-28.

[GrH81] D. Gross and C. M. Harris, *Fundamentals of Queueing Theory*, John Wiley and Sons, New York, NY, 1981.

[GrO93] R. I. Greenberg and H.-C. Oh, "Universal wormhole routing," *5th IEEE Symposium on Parallel and Distributed Processing*, December 1993, pp. 56-63.

[Gre92] P. E. Green, "An all-optical computer network: Lesson learned," *IEEE Network*, Vol. 6, No. 2, March 1992, pp. 56-60.

[GuA91] A. Gupta, I. F. Akyildiz, and R. M. Fujimoto, "Performance analysis of Time Warp with multiple homogeneous processors," *IEEE Transactions on Software Engineering*, Vol. 17, No. 10, October 1991, pp. 1013-1027.

[HaL90] G. Haworth, S. Leunig, C. Hammer, and M. Reeve, "The european declarative system, database, and languages," *IEEE Micro*, December 1990, pp. 20-23 and 83-87.

[Hah91] E. L. Hahne, "Round-robin scheduling for max-min fairness in data networks," *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 7, September 1991, pp. 1024-1039.

[Har91] P. G. Harrison, "On nonuniform packet switched delta networks and the hot-spot effect," *IEE Proceedings-E*, Vol. 138, No. 3, May 1991, pp. 123-130.

[HeN93] P. Heidelberger and D. M. Nicol, "Conservative parallel simulation of continuous time Markov Chains using uniformization," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 8, August 1993, pp. 906-921.

[HeS90] M. A. Henrion, K. J. Schrodi, D. Boettle, M. De Somer, and M. Dieudonne, "Switching network architecture for ATM based broadband communications," *XIII International Switching Symposium*, Vol. V, May 1990, pp. 1-8.

[Hil85] W. D. Hillis, *The Connection Machine*, MIT Press, Cambridge, MA, 1985.

[HlK88] M.G. Hluchyj and M.J. Karol, "Queueing in high-performance packet switching," *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 9, December 1988, pp. 1587-1597.

[Hoe90] B. Hoefflinger, "ASICs zum Spartarif," *Elektronik*, Vol. 26, Dezember 1990, pp. 64-72.

[HwB84] K. Hwang and F. A. Briggs, *Computer Architecture and Parallel Processing*, McGraw-Hill, New York, NY, 1984.

[Hwa84] K. Hwang, *Tutorial on Supercomputers: Design and Applications*, IEEE Computer Society Press, Washington, D.C., 1984.

[Hwa93] K. Hwang, *Advanced Computer Architecture*, McGraw-Hill, New York, NY, 1993.

[Ili91] I. Iliadis, "Head of the line arbitration of packet switches with combined input and output queueing," *International Journal of Digital and Analog Communication Systems*, Vol. 4, No. 3, July 1991, pp. 181-190.

[Iso84] ISO 7498, *Information Processing Systems - Open Systems interconnection - Basic Reference Model*, International Standard, 1984.

[Iso92] Draft ISO/IEC DIS 11172, *Information technology - Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s*, International Standard, 1992.

[Ito91] A. Itoh et al., "Practical implementation and packaging technologies for a large-scale ATM switching system," *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 8, October 1991, pp. 1280-1288.

[JaM86] L. H. Jamieson, P. T. Mueller, and H. J. Siegel, "FFT algorithms for SIMD parallel processing systems," *Journal of Parallel and Distributed Computing*, Vol. 3, No. 1, March 1986, pp. 48-71.

[JeS88] M. Jeng and H. J. Siegel, "Design and analysis of dynamic redundancy networks," *IEEE Transactions on Computers*, Vol. C-37, No. 9, September 1988, pp. 1019-1029.

[Jef85] D. R. Jefferson, "Virtual Time," *ACM Transactions on Programming Languages and Systems*, Vol. 7, No. 3, July 1985, pp. 404-425.

[Jen83] Y. C. Jenq, "Peformance analysis of a packet switch based on single-buffered banyan network," *IEEE Journal on Selected Areas in Communications*, December 1983, pp. 1014-1021.

[JuS93] M. Jurczyk and T. Schwederski, "On partially dilated multistage interconnection networks with uniform traffic and nonuniform traffic spots," *5th IEEE Symposium on Parallel and Distributed Processing*, December 1993, pp. 788-795.

[JuS94] M. Jurczyk and T. Schwederski, "Higher order head-of-line blocking in multistage interconnection networks," *7th ISCA International Conference on Parallel and Distributed Computing Systems*, October 1994, pp. 571-580.

[JuS95a] M. Jurczyk and T. Schwederski, "Switch box architecture for saturation tree effect minimization in multistage interconnection networks," *1995 International Conference on Parallel Processing*, August 1995, pp. I/41-I/45.

[JuS95b] M. Jurczyk and T. Schwederski, "SIMD Processing: Concepts and Systems" in *Handbook of Parallel and Distributed Computing*, A. Y. Zomaya, ed., McGraw-Hill, New York, NY, October 1995.

[JuS96] M. Jurczyk, T. Schwederski, H. J. Siegel, S. Abraham, and R. Born, "Strategies for the implementation of interconnection network simulators on parallel computers," *International Journal in Computer Simulation*, to appear, 1996.

[KaH87] M.J. Karol, M.G. Hluchyj, and S.P. Morgan, "Input versus output queueing on a space-division packet switch," *IEEE Transactions on Communications*, Vol. COM-35, No. 12, December 1987, pp. 1347-1356.

[KaK80] F. Kamoun and L. Kleinrock, "Analysis of shared finite storage in a computer network node," *IEEE Transactions on Communications*, Vol. 28, No. 7, July 1980, pp. 992-1003.

[KaS88] Y. Kato, T. Shimoe, K. Hajikano, and K. Murakami, "Experimental broadband ATM switching system," *IEEE GLOBECOM'91*, December 1988, pp. 1288-1292.

[KaS92] D. D. Kandlur and K. G. Shin, "Traffic routing for multicomputer networks with virtual cut-through capability," *IEEE Transactions on Computers*, Vol. 41, No. 10, October 1992, pp. 1257-1270.

[KeK79] P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique," *Computer Networks*, Vol. 3, 1979, pp. 267.

[KeS90] J. Kernhof, M. Selzer, M. A. Beunder, B. Hoefflinger, B. Laquai, and I. Schindler, "Mixed static and domino logic on the CMOS Gate Forest," *IEEE Journal of Solid-State Circuits*, Vol. 25, No. 2, April 1990, pp. 396-402.

[Kec94] D. O. Keck, *Parallele Simulation von Wormhole-Verbindungsnetzen auf dem MIMD-Parallelrechner Intel Paragon*, Diplomarbeit, Stuttgart: Institut fuer Nachrichtenvermittlung und Datenverarbeitung, 1994.

[KiD91] J. Kim and C. R. Das, "Modeling wormhole routing in a hypercube," *11th International Conference on Distributed Computing Systems*, 1991, pp. 386-393.

[KiL90] H. S. Kim and A. Leon-Garcia, "Performance of buffered banyan networks under nonuniform traffic patterns," *IEEE Transactions on Communications*, Vol. 38, No. 5, May 1990, pp. 648-658.

[KiL93] Y. M. Kim and K. Y. Lee, "PR-Banyan: A packet switch with a pseudorandomizer for nonuniform traffic," *IEEE Transactions on Communications*, Vol. 41, No. 7, July 1993, pp. 1039-1042.

[KiM91] F. E. Kiamilve, P. Marchand, A. V. Krishnamoorthy, S. C. Esener, and S. H. Lee, "Performance comparison between optoelectronic and VLSI multistage interconnection networks," *Journal of Lightwave Technology*, Vol. 9, No. 12, December 1991, pp. 1674-1692.

[Kle75] L. Kleinrock, *Queueing Systems, Volume 1: Theory*, John Wiley and Sons, New York, NY, 1975.

[Kon92] S. Konstantinidou, "Priorities in nonminimal, adaptive routing," *1992 International Conference on Parallel Processing*, Vol. I, August 1992, pp. 67-70.

[KoE91] T. Kozaki, N. Endo, Y. Sakurai, O. Matsubara, M. Mizukami, and K. Asano, "32x32 shared buffer type ATM switch VLSI's for B-ISDN's," *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 8, October 1991, pp. 1239-1247.

[KrH91] H. Kroener, G. Hebuterne, P. Boyer, and A. Gravey, "Priority management in ATM switching nodes," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-9, No. 3, April 1991, pp. 418-427.

[KrS88] C. P. Kruskal, M. Snir, and A. Weiss, "The distribution of waiting times in clocked multistage interconnection networks," *IEEE Transactions on Computers*, Vol. C-37, No. 11, November 1988, pp. 1337-1352.

[KrT92] N. Krivossidis and C. Tropper, "A performance analysis of Time Warp," *1992 International Conference on Parallel Processing*, Vol. II, August 1992, pp. 153-156.

[KuB93] M. Kumar, Y. Baransky, and M. Denneau, "The GF11 parallel computer," *Parallel Computing*, Vol. 19, 1993, pp. 1393-1412.

[KuJ86] M. Kumar and J. R. Jump, "Performance of unbuffered shuffle-exchange networks," *IEEE Transactions on Computers*, Vol. C-35, No. 6, June 1986, pp. 573-578.

[KuP86] M. Kumar and G. Pfister, "The onset of hot spot contention," *1986 International Conference on Parallel Processing*, August 1986, pp. 28-34.

[KuT91] L. Kurian and M. J. Thazhuthaveetil, "Effect of hot spots on multiprocessor systems using circuit switched interconnection networks," *1991 International Conference on Parallel Processing*, Vol. I, August 1991, pp. 554-557.

[LaK88] T. Lang and L. Kurisaki, "Nonuniform traffic spots (NUTS) in multistage interconnection networks," *1988 International Conference on Parallel Processing*, August 1988, pp. 191-195.

[Lav83] S. S. Lavenberg, *Computer Performance Modeling Handbook*, Academic Press, New York, NY, 1983.

[Law75] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Transactions on Computers*, Vol. C-24, No. 12, December 1975, pp. 1145-1155.

[LeH94] B. Lee and A.R. Hurson, "Dataflow architectures and multithreading," *IEEE Computer*, August 1994, pp. 27-39.

[LeK90] H. Lee, K.H. Kook, C.S. Rim, K.P. Jun, and S.K. Lim, "A limited shared output buffer switch for ATM," *4th International Conference on Data Communication Systems and their Performance*, June 1990, pp. 163-179.

[Len78] J. Lenfant, "Parallel permutations of data: A Benes network control algorithm for frequently used permutations," *IEEE Transactions on Computers*, Vol. C-27, No. 7, July 1978, pp. 637-647.

[LiA93] J. M. Lin and S. G. Abraham, "Utilizing global simulation information in conservative parallel simulation on shared memory multiprocessors," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 516-523.

[LiD88] Y.-S. Liu and S. Dickey, "Simulation and analysis of enhanced switch architectures for interconnection networks in massively parallel shared memory machines,"

UNKNOWN, probably: 8th International Conference on Distributed Computing Systems, June 1988, pp. 487-490.

[LiE93] X. Lin, A.-H. Esfahanian, P. K. McKinley, and A. Burago, "Adaptive wormhole routing in hypercube multicomputers," *5th IEEE Symposium on Parallel and Distributed Processing*, December 1993, pp. 72-79.

[LiL89] S. C. Liew and K. W. Lu, "A three-stage architecture for very large packet switches," *International Journal of Digital and Analog Cabled Systems*, Vol. 2, No. 4, October-December 1989, pp. 303-316.

[LiL90] Y.-B. Lin and E. D. Lazowska, "Exploiting lookahead in parallel simulation," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 1, No. 4, October 1990, pp. 457-469.

[LiN93] X. Lin and L. M. Ni, "Multicast communication in multicomputer networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No. 10, October 1993, pp. 1105-1117.

[LiP93] Y.-B. Lin, B. R. Preiss, W. M. Loucks, and E. D. Lazowska, "Selecting the checkpoint interval in Time Warp simulation," *7th Workshop on Parallel and Distributed Simulation*, Vol. 23, No. 1, July 1993, pp. 3-10.

[Lin92] Y.-B. Lin, "Effects of waiting overheads on conservative parallel simulation," *25th Annual Simulation Symposium*, April 1992, pp. 61-70.

[Luk93] P. Luksch, "Evaluation of three approaches to parallel logic simulation on a distributed memory multiprocessor," *26th Annual Simulation Symposium*, March 1993, pp. 2-11.

[Mat93] F. Mattern, "Efficient Algorithms for Distributed Snapshots and Global Virtual Time Approximation," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 423-434.

[McS82] R. J. McMillen and H. J. Siegel, "A comparison of cube type and data manipulator type networks," *Third International Conference on Distributed Computing Systems*, October 1982, pp. 614-621.

[Mis86] J. Misra, "Distributed discrete-event simulation," *ACM Computing Surveys*, Vol. 18, No. 1, March 1986, pp. 39-65.

[MuS80] P. T. Mueller, Jr., L. J. Siegel, and H. J. Siegel, "Parallel algorithms for the two-dimensional FFT," *Fifth International Conference on Pattern Recognition*, December 1980, pp. 497-502.

[MuY94] Y. Mun and H. Y. Youn, "Performance analysis of finite buffered multistage interconnection networks," *IEEE Transactions on Computers*, Vol. 43, No. 2, February 1994, pp. 153-162.

[NaS82] D. Nassimi and S. Sahni, "Optimal BPC permutations on a cube connected SIMD computer," *IEEE Transactions on Computers*, Vol. C-31, April 1982, pp. 338-341.

[NaS93] W. G. Nation, G. Saghi, and H. J. Siegel, "Properties of interconnection networks for large-scale parallel processing systems," *International Summer Institute on Parallel Computer Architectures, Languages, and Algorithms (ISIPCALA)*, July 1993, pp. 51-82.

[NiH93] D. M. Nicol and P. Heidelberger, "Optimistic parallel simulation of continuous time Markov Chains using uniformization," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 395-410.

[NiM93] L. M. Ni and P. K. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Computer*, Vol. 26, No. 2, February 1993, pp. 62-76.

[Nic90] J. R. Nickolls, "The design of the MasPar MP-1: A cost effective massively parallel computer," *IEEE Compcon*, February 1990, pp. 25-28.

[Nic92] D. Nicol, "Conservative parallel simulation of priority class queueing networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 3, May 1992, pp. 294-303.

[NoF89] M. Nomura, T. Fujii, and N. Ohta, "Basic characteristics of variable rate video coding in ATM environment," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-7, No. 5, June 1989, pp. 752-760.

[PaA93] A. C. Palaniswamy, S. Aji, and P. A. Wilsey, "Performance measures for several optimizations to a distributed digital system simulator," *26th Annual Simulation Symposium*, March 1993, pp. 21-29.

[PaZ93] P. Pancha and M. El Zarki, "A look at the MPG video coding standard for variable bit rate video transmission," *IEEE INFOCOM'93*, May 1993, pp. 85-94.

[Pat81] J. H. Patel, "Performance of processor-memory interconnections for multiprocessors," *IEEE Transactions on Computers*, Vol. C-30, No. 10, October 1981, pp. 771-780.

[PeL93] J.-K. Peir and Y.-H. Lee, "Look-ahead routing switches for multistage interconnection networks," *Journal of Parallel and Distributed Computing*, Vol. 19, No. 1, September 1993, pp. 1-10.

[Pea68] M.C. Pease, "An adaption of the Fast Fourier Transform for parallel processing," *Journal of the ACM*, Vol. 15, April 1968, pp. 252-264.

[Pea77] M. C. Pease III, "The indirect binary n-cube microprocessor array," *IEEE Transactions on Computers*, Vol. C-26, No. 5, May 1977, pp. 458-473.

[Per87] R. H. Perrott, *Parallel Programming*, Addison-Wesley, Reading, MA, 1987.

[PfB85] G. F. Pfister, W. C. Brantley, D. A. George, S. L. Harvey, W. J. Kleinfelder, K. P. McAuliffe, E. A. Melton, V. A. Norton, and J. Weiss, "The IBM Research Parallel Processor Prototype (RP3): Introduction and architecture," *1985 International Conference on Parallel Processing*, August 1985, pp. 764-771.

[PfN85] G. F. Pfister and V. A. Norton, "'Hot spot' contention and combining in multistage interconnection networks," *IEEE Transactions on Computers*, Vol. C-34, No. 10, October 1985, pp. 933-938.

[PoT93] R. Ponnusamy, R. Thakur, A. Choudhary, K. Velamakanni, Z. Bozkus, and G. Fox, "Experimental performance evaluation of the CM-5," *Journal of Parallel and Distributed Computing*, Vol. 19, 1993, pp. 192-202.

[PrS87] M. De Prycker and M. De Somer, "Performance of a service independent switching network with distributed control," *IEEE Journal on Selected Areas in Communications*, Vol. 5, No. 8, October 1987, pp. 1293-1301.

[Pre90] B. R. Preiss, "Performance of discrete event simulation on a multiprocessor using optimistic and conservative synchronization," *1990 International Conference on Parallel Processing*, Vol. III, August 1990, pp. 218-222.

[RaF92] M. Rawling, R. Francis, and D. Abramson, "Potential performance of parallel conservative simulation of VLSI circuits and systems," *25th Annual Simulation Symposium*, April 1992, pp. 71-81.

[RaS90] G. Ramamurthy and B. Sengupta, "Modelling and analysis of a variable bit rate video multiplexer," *7th ITC Specialist Seminar*, October 1990.

[RaT89] E. P. Rathgeb, T. H. Theimer, and M. N. Huber, "ATM switches - Basic architectures and their performance," *International Journal of Digital and Analog Cabled Systems*, Vol. 2, 1989, pp. 227-236.

[Raj92] H. Rajaei, "SIMA: An environment for parallel discrete-event simulation," *25th Annual Simulation Symposium*, April 1992, pp. 147-155.

[Rat91] E. P. Rathgeb, *Verkehrsfluesse in ATM-Netzen - Modellierung und Analyse von Verkehrsquellen und Quellflusskontrollverfahren*, 51. Bericht ueber verkehrstheoretische Arbeiten, Stuttgart: Institut fuer Nachrichtenvermittlung und Datenverarbeitung, 1991.

[ReC90] R. D. Rettberg, W. R. Crowther, P. P. Carvey, and R. S. Tomlinson, "The Monarch parallel processor hardware design," *Computer*, Vol. 23, No. 4, April 1990, pp. 18-30.

[ReP93] P. F. Reynolds, Jr., C. M. Pancerella, and S. Srinivasan, "Design and performance analysis of hardware support for parallel simulations," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 435-453.

[RoA93] R. Roennegren, R. Ayani, R. M. Fujimoto, and S. R. Das, "Efficient implementation of event sets in Time Warp," *7th Workshop on Parallel and Distributed Simulation*, Vol. 23, No. 1, July 1993, pp. 101-108.

[RoS88] K. Rothermel and D. Seeger, "Traffic studies of switching network for asynchronous transfer mode (ATM)," *International Teletraffic Conference 12*, June 1988, pp. 1.3A.5.1-1.3A.5.7.

[SaI90] Y. Sakurai, N. Ido, S. Gohara, and N. Endo, "Large scale ATM multi-stage switching network with shared buffer memory switching," *XIII International Switching Symposium*, Vol. IV, May 1990, pp. 121-126.

[ScD87] T. Schwederski and H. G. Dietz, "Barrier synchronization in the PASM parallel processing system," *Third SIAM Conference on Parallel Processing for Scientific Computing*, December 1987.

[ScS90a] S. L. Scott and G. S. Sohi, "The use of feedback in multiprocessor and its application to tree saturation control," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 1, No. 4, October 1990, pp. 385-398.

[ScS90b] D.B. Schwartz and V.K. Samalam, "VLSI for ATM network management," *IEEE GLOBECOM'90*, December 1990, pp. 1387-1392.

[Sch92] A. Schwarz, *Untersuchung und Entwurf der Kontrolleinheit fuer ein Koppelelement mit Zentralspeicher*, Diplomarbeit, IND/IMS, 1992.

[SeT90] K. Sezaki, Y. Tanaka, and M. Akiyama, "The cascade Clos broadcast switching network - A new ATM switching network which is multiconnection non-blocking," *XIII International Switching Symposium*, Vol. IV, May 1990, pp. 143-147.

[SeU80] M. C. Sejnowski, E. T. Upchurch, R. N. Kapur, D. P. S. Charlu, and G. J. Lipovski, "An overview of the Texas Reconfigurable Array Computer," *AFIPS 1980 National Computer Conference*, June 1980, pp. 631-641.

[ShM90] K. Shiomoto, M. Murata, Y. Oie, and H. Miyahara, "Performance evaluation of cell bypass queueing discipline for buffered Banyan type ATM switches," *IEEE INFOCOM'90*, Vol. 2, June 1990, pp. 677-685.

[SiM90] M. Siegel, R. Medow, T. Theimer, and T. Schwederski, "Design of a single-chip ATM switching element," *International Conference on Computer Communication*, November 1990, pp. 804-811.

[SiN89] H. J. Siegel, W. G. Nation, C. P. Kruskal, and L. M. Napolitano, "Using the multistage cube network topology in parallel supercomputers," *Proceedings of the IEEE*, Vol. 77, No. 12, December 1989, pp. 1932-1953.

[SiS78] H. J. Siegel and S. D. Smith, "Study of multistage SIMD interconnection networks," *Fifth Annual Symposium on Computer Architecture*, April 1978, pp. 223-229.

[SiS81] H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, Jr., and S. D. Smith, "PASM: A partitionable SIMD/MIMD system for image processing and pattern recognition," *IEEE Transactions on Computers*, Vol. C-30, No. 12, December 1981, pp. 934-947.

[Sie79] H. J. Siegel, "Interconnection networks for SIMD machines," *Computer*, Vol. 12, No. 6, June 1979, pp. 57-65.

[Sie90] H. J. Siegel, *Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies, Second Edition*, McGraw-Hill, New York, NY, 1990.

[SkS93] P. Skelly, M. Schwartz, and S. Dizit, "A histogram-based model for video traffic behavior in an ATM multiplexer," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, August 1993, pp. 446-459.

[SoS93] T. K. Som and R. G. Sargent, "A new process to processor assignment criterion for reducing rollbacks in optimistic simulation," *Journal of Parallel and Distributed Computing*, Vol. 18, No. 4, August 1993, pp. 509-515.

[StZ90] M. Stumm and S. Zhou, "Algorithms implementing distributed shared memory," *Computer*, Vol. 23, No. 5, May 1990, pp. 54-64.

[Ste93] J. S. Steinmann, "Breathing Time Warp," *7th Workshop on Parallel and Distributed Simulation*, Vol. 23, No. 1, July 1993, pp. 109-118.

[Sto71] H. S. Stone, "Parallel processing with the perfect shuffle," *IEEE Transactions on Computers*, Vol. C-20, No. 2, February 1971, pp. 153-161.

[Sto87] H. S. Stone, *High Performance Computer Architecture*, Addison-Wesley Publishing, Co., Reading, PA, 1987.

[SuN89] H. Suzuki, H. Nagano, T. Suzuki, T. Takeuchi, and S. Iwasaki, "Output-buffer switch architecture for asynchronous transfer mode," *International Journal of Digital and Analog Cabled Systems*, Vol. 2, 1989, pp. 269-276.

[SuS93] C.-C. Su and K. G. Shin, "Adaptive deadlock-free routing in multicomputers using only one extra virtual channel," *1993 International Conference on Parallel Processing*, Vol. I, August 1993, pp. 227-231.

[SzH87] T. H. Szymanski and V. C. Hamacher, "On the permutation capability of multistage interconnection networks," *IEEE Transactions on Computers*, Vol. C-36, No. 7, July 1987, pp. 810-822.

[TaF88] Y. Tamir and G. L. Frazier, "High-performance multi-queue buffers for VLSI communication switches," *15th International Symposium on Computer Architecture*, 1988, pp. 343-354.

[TaF92] Y. Tamir and G. L. Frazier, "Hardware support for high-priority traffic in VLSI communication switches," *Journal of Parallel and Distributed Computing*, Vol. 14, No. 4, April 1992, pp. 402-416.

[Tan81] A. S. Tanenbaum, *Computer Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[ThR91] T. H. Theimer, E. P. Rathgeb, and M. N. Huber, "Performance analysis of buffered banyan networks," *IEEE Transactions on Communication*, Vol. 39, No. 2, February 1991, pp. 269-277.

[The94] T.H. Theimer, *Vergleichende Untersuchungen an ATM-Koppelnetzstrukturen*, Dissertation, Stuttgart: Institut fuer Nachrichtenvermittlung und Datenverarbeitung, 1994.

[ToG93] A. I. Tomlinson and V. K. Garg, "An algorithm for minimally latent global virtual time," *7th Workshop on Parallel and Distributed Simulation*, Vol. 23, No. 1, July 1993, pp. 35-42.

[TuR88] L. W. Tucker and G. G. Robertson, "Architecture and applications of the Connection Machine," *Computer*, Vol. 21, No. 8, August 1988, pp. 26-38.

[Tur93] J. S. Turner, "Queueing analysis of buffered switching networks," *IEEE Transactions on Communications*, Vol. 41, No. 2, February 1993, pp. 412-420.

[Tze91a] N.-F. Tzeng, "An approach to the performance improvement of multistage interconnection networks with nonuniform traffic spots," *1991 International Conference on Parallel Processing*, Vol. I, August 1991, pp. 542-545.

[Tze91b] N.-F. Tzeng, "Alleviating the impact of tree saturation on multistage interconnection network performance," *Journal of Parallel and Distributed Computing*, Vol. 12, No. 2, June 1991, pp. 107-117.

[ViV91] M. Villen-Altamirano and J. Villen-Altamiranoo, "RESTART: A method for accelerating rare event simulations," *13th International Teletraffic Congress, Workshop on Queueing, Performance, and Control in ATM*, June 1991, pp. 71-76.

[Vri90] R. C. De Vries, "Reducing null messages in Misra's distributed discrete event simulation method," *IEEE Transactions on Software Engineering*, Vol. 16, No. 1, January 1990, pp. 82-91.

[WaS95] M.-C. Wang, H. J. Siegel, M. A. Nichols, and S. Abraham, "Using a multipath network for reducing the effect of hot spots," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 6, No. 3, March 1995, pp. 252-268.

[Wen91] J. Wenzel, *Untersuchung und VLSI-Entwurf des Datenpfades fuer ein ATM-Koppelelement mit Zentralspeicher*, Diplomarbeit, IND/IMS, 1991.

[WiJ89] F. Wieland and D. Jefferson, "Case studies in serial and parallel simulation," *1989 International Conference on Parallel Processing*, Vol. III, August 1989, pp. 255-258.

[WuF80a] C.-L. Wu and T. Y. Feng, "On a class of multistage interconnection networks," *IEEE Transactions on Computers*, Vol. C-29, No. 8, August 1980, pp. 694-702.

[WuF80b] C.-L. Wu and T. Y. Feng, "The reverse-exchange interconnection network," *IEEE Transactions on Computers*, Vol. C-29, September 1980, pp. 801-811.

[YaM91] Y. Yang and G. M. Masson, "Nonblocking broadcast switching networks," *IEEE Transactions on Computers*, Vol. C-40, No. 9, September 1991, pp. 1005-1015.

[YeH87] Y.-S. Yeh, M. G. Hluchyj, and A. S. Acampora, "The Knockout switch: A simple, modular architecture for high-performance packet switching," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No. 8, October 1987, pp. 1274-1283.

[YeT87] P.-C. Yew, N.-F. Tzeng, and D. H. Lawrie, "Distributing hot-spot addressing in large-scale multiprocessors," *IEEE Transactions on Computers*, Vol. C-36, No. 4, April 1987, pp. 388-395.

[YoL90] H. Yoon, K. Y. Lee, and M. T. Liu, "Performance analysis of multibuffered packet-switching networks in multiprocessor systems," *IEEE Transactions on Computers*, Vol. C-39, No. 3, March 1990, pp. 319-327.

[Yu92] M.-L. Yu, "En route to more efficient conservative parallel event simulation," *25th Annual Simulation Symposium*, April 1992, pp. 54-60.

Danksagung

Herrn Prof. Dr. P. J. Kühn danke ich für die freundliche Übernahme des Hauptberichts. Seine Betreuung, seine Förderung und seine Ideen haben wesentlich zum Gelingen dieser Arbeit beigetragen.

Herrn Prof. Dr. B. Höfflinger danke ich für die freundliche Übernahme des Mitberichts und die sich daraus ergebenen wertvollen Diskussionen. Auch bedanke ich mich für die Bereitstellung eines Arbeitsplatzes am IMS.

Weiterhin möchte ich Herrn Dr. T. Schwederski für seine Unterstützung und die Anregungen, die er mir im Verlaufe dieser Arbeit zukommen ließ, meinen aufrichtigen Dank aussprechen.

Danken möchte ich allen Kollegen des IMS, die mich als Graduiertenkollegiaten akzeptiert und unterstützt haben. Außerdem möchte ich allen Studenten danken, die zur Anfertigung dieser Arbeit beigetragen haben.

Mein Dank für das besondere Interesse und die Unterstützung dieser Arbeit gilt Dipl.-Ing. J. Enssle, Dipl.-Ing. D. O. Keck und Dipl.-Ing. S. Neusser.