Parallel und Batch Scientific Computing
TR 1993–2
- Entwurf -

Wolfgang Ksoll
CompuNet Berlin
Mariendorfer Damm 1–3
D–12099 Berlin
E–Mail: wks@CNB.CompuNet.DE

6. Dezember 1993

Inhaltsverzeichnis

1 Einführung

In vielen Wissenschaftsgebieten wie

- Hochenergiephysik (HEP)

- Quantenphysik, Quantenchemie
- Moleküldynamik
- Umweltmodellierung
- Computational Fluid Dynamics (CFD) bzw. Strömungsmechanik
- Bilddatenverarbeitung (Medizin, Geodäsie, Geophysik)
- Seismik (Exploration)

sind die numerischen Methoden soweit fortgeschritten, daß eine enorme Rechenleistung nachgefragt wird. Glaubte man früher noch, daß diese Nachfrage durch einen Leistungszuwachs einzelner Prozessoren (z.B. durch Einsatz von GaAs–Technologie) zumindest eine zeitlang befriedigt werden könnte, ist heute eine breite Bewegung im Gange, in der Anwender ihre Programme parallelisieren bzw. auf mehrere oder eine Vielzahl von Prozessoren verteilen.

Hier soll in einem Überblick gezeigt werden, welche Hard- und Softwaresysteme vom „Main Stream" der Anwender entwickelt und benutzt werden, um die eigene Entscheidung zur Parallelisierung oder Verteilung einer Anwendung zu erleichtern.

Man kann die unterliegende Hardware für solche Lösungen nach der Bandbreite der Kopplung der Prozessoren ordnen, da sich aus der Kopplung unterschiedliche Möglichkeiten der Programmierung ergeben.

Mit der höchsten Bandbreite arbeiten **Shared Memory Systeme**. Diese Systeme haben ein zentrales Memory, auf das mehrere Prozessoren über einen Bus oder einen Cross–Bar–Switch zugreifen (Cray, Convex, etc.). Diese Systeme sind für den Programmierer sehr freundlich: Er kann seine Anwendung aufgrund der hohen Bandbreite nicht nur auf Haupt– oder Unterprogrammebene sondern auch auf Statementebene parallelisieren (feine Granularität). Im günstigsten Fall (wenn Superposition in seiner Anwendung erlaubt ist) braucht er sich nicht mal Gedanken über die Synchronisation seiner Prozesse machen, in dem er in allen Instanzen seines Programmes auf dieselbe Matrix im Speicher zugreift.

Bei einer grossen Anzahl von Prozessoren ist aber auch das Memory zu verteilen und man kommt nicht umhin, andere Kommunikationsmechanismen zwischen den Prozessen einzuführen. Alle Massiv Parallelen Systeme arbeiten dann mit **Message Passing** Prinzipien, deren Kommunikationsoverhead nicht mehr zu vernachlässigen ist. Bei 500 oder 1000 Prozessoren ist echtes Shared Memory nicht machbar und Virtual Shared Memory ist in diesen Grössenordnugen bisher nicht gezeigt worden.

Nimmt man dann lose gekoppelte Systeme dazu, wie Workstations mit gängigen Netzwerkverbindungen (Ethernet, FDDI etc.), lohnt sich eine Verteilung nur, wenn ein ausgewogenes Verhältnis zwischen lokaler Rechenzeit eines Prozesses und seinem Kommunikationsbedürfnis besteht. Zum Glück ist das bei sehr vielen numerisch intensiven Anwendungen der Fall, so daß an vielen Einrichtungen der Wunsch besteht, den Zuwachs an Rechenbedarf nicht nur durch große, teure Zentralsysteme, sondern auch durch viele, preiswertere Workstations in Clustern oder Farms zu befriedigen.

Eine andere Art der Parallelisierung lassen **Queueing Systeme** zu. Mit ihnen kann man mehrere Instanzen eines Hauptprogrammes gleichzeitig auf einem oder mehreren Prozessoren mit unterschiedlichen Parametern laufen lassen. Man kann für einen Prozessor Warteschlangen unterschiedlicher Eigenschaften einrichten, so daß z.B. immer nur ein Prozeß mit hoher Memoryanforderung und ein Prozeß mit intensiven I/O–Anforderungen aktiv ist. Damit kann man zu einer optimalen, gleichmäßigen Auslastung eines Rechners kommen.

In einem Rechnernetzwerk kann man dann eine Lastverteilung und einen Lastausgleich unter den Prozessoren herbeiführen. Dafür ist es dann notwendig, daß das Queueing System über eine solche zentrale Instanz verfügt. Hohe Schule ist es dann, wenn der Scheduler auch in der Lage ist, in die einzelnen Warteschlangen auch einen Batchjob einzufügen, der zur Laufzeit mehrere Prozessoren braucht.

In dieser Übersicht wird der aktuelle Stand bei den Message Passing und Queueing Systemen aufgezeigt. Nicht eingegangen wird auf Shared Memory Systeme. Dann wird berichtet, welche Fortschritte in der Numerik mit der Parallelisierung gemacht werden. Anschließend wird an einigen Beispielen gezeigt, wo Parallelisierung schon Eingang in Standard–Pakete gefunden hat. Zudem werden einige Programmierwerkzeuge aufgezeigt. An den aktuellen Massiv Parallelen Systemen und einigen ausgewählten Clustern

wird dann die Kombination von Hard- und Software angedeutet. Zur Leistungsbeurteilung dieser Gesamtsysteme werden die Entwicklungen im Benchmarkbereich dargestellt.

2 Message Passing Systeme

2.1 PVM — Parallel Virtual Machine

PVM (Parallel Virtual Machine) wurde entwickelt in einem Joint Venture des Oak Ridge National Laboratory, der Emory University, Atlanta, und der Universität von Tennessee. Gesponsort wird das Vorhaben vom U.S. Department of Energy, der National Science Foundation und dem Staat von Tennessee.

PVM ist eine verteilte Programmierumgebung für parallele Anwendungen in Netzwerken von Prozessoren. PVM arbeitet nach dem Message–Passing–Programmiermodell und ist designt für heterogene Sammlungen von Prozessoren. Es besteht aus einer Bibliothek von Unterprogrammen, die aus FORTRAN oder C aufgerufen werden können, und aus unterstützender Software (Dämonen). Die Kommunikation basiert auf IP/UDP–Protokollen [45, 46, 111, 112, 35, 78].

HeNCE (Heterogenous Network Computing Environment) ist eine parallele Programmierumgebung, die die Erstellung, Compilation, Ausführung, das Debuggen und die Analyse von parallelen Programmen erleichtert. Zu diesem Zweck ist HeNCE mit einer graphischen Oberfläche unter X–Windows ausgestattet. Seit September 1992 ist die Version 2.0 fertiggestellt [112].

Im Dezember 1992 wurde die Version 2.4.1 von PVM freigegeben. Seit Sommer 1993 ist die Version 3 freigegeben. Es gibt eine neue Namenskonvention für die Subroutinen. Dynamische Prozeß Gruppen und eine dynamische Konfiguration wurden eingeführt. Prozesse können unter dem Debugger `dbx` gestartet werden. Die Maschinenauswahl ist auf Basis von Geschwindigkeit und Last möglich. Zudem können sich die PVM–Prozesse über Signale verständigen [112].

Um zu einer höheren Betriebssicherheit insbesondere bei Langläufern zu gelangen, wurden an der Carnegie Mellon University Untersuchungen und Beispielimplementierungen zu einem Fail-safe PVM durchgeführt [74]. Der Einbau von Checkpoints und ein automatisches Recovery im Fehlerfall sind hier die Untersuchungsgegenstände.

Am NASA Lewis Research Center wurden sieben IBM RS/6000-550 Workstations und 2 CRAY-Y-MP verclustert und mit einer aerodynamischen Anwendung, die mit PVM paralellisiert worden war, durchgemessen. Je nach Konfiguration wurden vier bis acht RS/6000–Prozessoren benötigt, um die Leistung eines Y-MP–Prozessors zu erbringen [106].

Im Rahmen ihrer Forschung zu Parform, einem System mit dynamischen Lastausgleich in heterogen Netzwerken, haben Cap und Strumpen, Universität Zürich, am Beispiel eines Wärmeleitungsproblems (Partielle Differentialgleichungen) vergleichende Messungen zwischen Parform, Linda und PVM veröffentlicht [15].

PVM wird über die NETLIB distributiert. Dafür ist es notwendig, eine E–Mail an `netlib@ornl.gov` zu schicken, in der steht:

```
send pvm_shar from pvm
send index from pvm
```

Damit erhält man eine Reihe von Briefen, in denen dann die neueste Version von PVM als Source–Code enthalten ist. Um das Release von PVM 3.0 zu bekommen, sollte man sich zunächst einen Überblick verschaffen mit:

```
send index from pvm3.
```

PVM ist aber auch über anonymen FTP zu bekommen, z.B. auf `netlib2.cs.utk.edu` im Directory `pvm3`.

In den USENET–NEWS findet man aktuelle Informationen in der Newsgruppe `comp.parallel.pvm`.

Mittlerweile gibt es auch kommerzielle PVM-Implementierungen. Cray unterstützt die Version 3.0 auf ihrem T3D [93]. Convex unterstützt die Version 3.0 auf Convex und HP-UX. IBM vertreibt über das ECSEC (European Center for Scientific and Engineering Computing) in Rom PVM/6000. Hierbei werden für Workstation-Cluster spezielle Netzwerke unterstützt: SOCC, FDDI und der Allnode-Switch. Eine Portierung auf FCS (Fiber Channel Standard) ist in Arbeit. PVM/6000 Version 2 basiert auf PVM 3.2. An Leistungsdaten hat IBM veröffentlicht [63]:

	Latency [μs]	Bandwidth [MBytes/s]
SOCC (standard device driver)	600	6,2
SOCC (enhanced device driver)	450	10,1
FDDI	250	6,5
Allnode	110	3,7

2.2 P4

P4 (Portable Programs for Parallel Processors) ist eine Bibliothek von Makros und Unterprogrammen für FORTRAN- und C-Programme, die am Argonne National Laboratory, Illinois, entwickelt wurde. P4 basiert auch auf den Erfahrungen, die mit PARMACS (Parallel Macros) gemacht wurden. Für Maschinen mit Shared Memory werden portable Locks und Monitore unterstützt. Für Cluster mit verteiltem Memory werden Message-Passing-Prinzipien eingesetzt [77, 13, 14].

Eine Vielfalt von Kommunikationsmöglichkeiten ist gegeben: wo vorhanden, wird Shared Memory angesprochen. Hypercubes und HiPPI-Verbindungen können eingesetzt werden. Lose gekoppelte Maschinen kommunizieren über Sockets. Wahlweise kann im heterogen Umfeld das XDR-Protokoll (External Data Representation) genutzt werden. Mit P4 ist es möglich, Mehrprozessor- und Monoprozessormaschinen zu einem Ganzen zusammenzuclustern.

P4 wird in einer Vielzahl von Institutionen und Firmen (Boeing, Motorola, Exxon, etc.) in unterschiedlich NIC-Anwendungen [1] eingesetzt. Unter anderem wird es auch am Argonne National Laboratory in der theoretischen Chemie für Full-CI ab initio Rechnungen eingesetzt [77].

Zur dynamischen Laufzeitanalyse ist das Paket UPSHOT einsetzbar, das mit dynamischen Histogrammen und einfacher Animation die Programmentwicklung unterstützt. In Entwicklung befindet sich PADL (Program Animation Display Language), das eine graphische Analyse von Logfiles erlaubt [77, 14].

Die Entwickler von P4 beteiligen sich an den Bemühungen, einen Message Passing Standard zu schaffen.

Zur Zeit werden folgende Maschinen direkt unterstützt:

Intel Touchstone DELTA, Intel iPSC/860, Intel Paragon, CM-5, KSR, Alliant, Sequent Symmetry, Encore Multimax, Cray X/MP und C90, BBN TC-2000, Sun, NeXT, SGI, HP, RS/6000, Stardent Titan, nCUBE, IBM SP1 [14].

Auf dem Rechner `info.mcs.anl.gov` im Directory pub/p4 liegt die Datei `p4-1.3.tar.Z`.

2.3 PARMACS

PARMACS (PARallel MACroS) entstand 1987 während eines Besuches von Rolf Hempel von der GMD (Gesellschaft für Mathematik und Datenverarbeitung, Sankt Augustin) am Argonne National Laboratory. Ursprünglich waren es Macros, die in ein C-Programm eingefügt werden konnten, um Message Passing zwischen Prozessen zu erlauben. Während einer Zusammenarbeit von ANL und GMD im Jahre 1988 entstand ein FORTRAN-Interface. Ein Jahr später trennten sich die Wege. Während das ANL aus PARMACS P4 entwickelte, wurde bei der GMD PARMACS weiterentwickelt [16, 14].

In der aktuellen Version 6.0 liegen keine Macros mehr vor sondern eine Programmierbibliothek für FORTRAN77 und ANSI C. In FORTRAN beginnen alle Unterprogramm mit `PM` und in C mit `pm_`. PARMACS unterstützt Message Passing in einem Host/Worker-Modell. Das Message Passing kann synchron und asynchron sein, jeder Prozess kann mit jedem anderen Prozess kommunizieren. Eine logische Prozesstopologie findet Unterstützung. Es kann eine optimierte Prozess-zu-Prozessor-Zuordnung doer ein Load Levelling eingesetzt werden. Zur Synchronisation können Barrieren verwandt werden. Volle Kompatibilität zu Version 5.1 ist gegeben [16, 55, 94, 54].

Als Debugginghilfe kann eine instrumetierte PARMACS-Bibliothek eingesetzt werden. Tracefiles können mit Visualisierungstools aus dem GENESIS Projekt bearbeitet werden oder es kann nach einer Konvertierung ParaGraph eingestzt werden. Im PPPE Esprit Projekt wird ein portabler Debugger entwickelt [16].

[1] NIC = Numerically Intensive Computing

PARMACS wird eingesetzt in industriellen und forschungsorientierten Kontexten. Die GMD Communications Library (Comlib) nutzt PARMACS bei gitterorientierten Lösungsverfahren wie bespielsweise partielle Differentialgleichungen in zwei und drei Raumdimensionen.

Während bei der GMD weiterhin Forschung und Weiterentwicklung betrieben wird, werden die aktuellen Portierungen, der Vertrieb und der Support durch Firma Pallas GmbH, Hermuelheimer Str.10, D-50321 Bruehl, Tel: +49-2232-1896-0, FAX: +49-2232-1896-29, E-Mail: `info@pallas-gmbh.de`, durchgeführt.

Damit unterscheidet sich PARMACS von PVM und P4, so daß der industrielle Anwender sich darauf verlassen kann, daß er eine stabile Plattform für seine Anwendung findet und nicht Teil eines Forschungsprojektsfeldttest ist.

PARMACS ist für folgende Distributed Memory Systeme verfügbar: Cray T3D, IBM SP1, Intel iPSC/2, iPSC/860, Paragon, Meiko CS-1 und CS-2, nCUBE 2, Parsytec GCel, GC, SC, Xplorer, TMC CM-5, Transtech Pyramid.

Aber es werden auch Shared Memory Systeme unterstützt: Convex C2, C3; Cray Y-MP, C90, EL.

PARMACS 6.0 ist derzeit für folgende Workstations verfügbar: DEC Alpha und MIPS, HP9000/700, IBM RS/6000, SGI Iris, SUN SPARCstations.

Lizenzen für Workstations gibt es in drei Klassen: 1-4, bis 12 und ∞ zeitlich unbegrenzt. Für Kunden aus dem Bereich Lehre und Forschung gibt es einen Nachlass von 30 % .

2.4 TCGMSG

Robert J. Harrison (`harrison@tcg.anl.gov`) aus der Theoretical Chemistry Group des Argonne National Laboratory beschreibt TCGMSG als Paket einfacher Send–/Receive–Subroutinen [53]. Sie unterstützen Shared Memory Mechanismen, Sockets und auch XDR. Unter anderem werden sie eingesetzt in Chemie–Paketen wie GAMESS, COLUMBUS oder DISCO.

Das Paket ist frei verfügbar über anonymen Ftp vom Rechner `ftp.tcg.anl.gov` erhältlich.

2.5 Express

Am California Institute of Technology entstand 1982 der „Cosmic Cube" als einer der ersten praktischen Parallelrechner. Das Forschungsteam, das sich mit der Entwicklung von Software für die Maschine beschäftigte, gründete die Firma ParaSoft.

Express ist ein kommerzielles Produkt der ParaSoft Corporation, Pasadena, das neben Unterprogrammen zur Verteilung von Daten (implizites Message Passing) auch eine Reihe von Werkzeugen zur Programmentwicklung beinhaltet [95, 96].

- `INSIGHT` wird zur Laufzeitanalyse des sequentiellen und parallelen Programms eingesetzt. Es können Memorybereiche identifiziert werden, auf die zur Laufzeit zugegriffen wird. Es werden FORTRAN und C unterstützt.
- Zur Programmflußanalyse kann `FTOOL` herangezogen werden.
- ASPAR ist ein automatischer Parallelisierer für sequentielle FORTRAN und C Programme
- `NETF77` ist der Netzwerk–FORTRAN–Compiler.
- `NDB` heißt der Netzwerkdebugger, der mit einer `dbx`– ähnlichen Systax gesteuert wird. Er ist ein Sourcecode–Debugger für parallele Programme, der sowohl für Mehrprozessormaschinen als auch für Workstation–Cluster verfügbar ist.
- `PM` wird zur Performance Analyse verwendet.
- `Express Runtime` dient zur dynamischen Lastverteilung zur Laufzeit.

Express stützt auch einige spezielle Kommunikationshardware mit geringer Latenzzeit und hoher Übertragungsrate, z.B. den BiT-3 Switch oder den IBM V7 Switch (Allnodeswitch), sowie diverse HIPPI-interfaces. Zudem lassen sich die Express Driver leicht an eigene Hardware anpassen [96].

In Deutschland wird Express von der Genias Software GmbH, Erzgebirgsstr. 2 B, D–93073 Neutraubling, Tel: ++49 +9401 9200-0, FAX: ++49 +9401 9200-92, (mailbox@genias.de) vertrieben. Lizenzen können für C und FORTRAN getrennt oder gemeinsam erworben werden. Die Firma Genias führt auch Schulungen und Parallelisierungen durch.

2.6 IBM Message Passing Library (MPL)

Das IBM Parallel Environment bietet Unterstützung für die Entwicklung und die Analyse paralleler Programme in einem TCP/IP–Netzwerk. Es werden FORTRAN– und C–Programme unterstützt. Es besteht aus mehreren Komponenten:

- Paralleles API (Message Passing Library MPL)
- Paralleles Operating Environment
- Paralleler Debugger (PDBX)
- Paralleles Profiling (ähnlich wie `prof` und `gprof`)
- Performance Visualisierung und Monitoring

Das Parallel Environment ist seit September 1993 verfügbar [62].

Die Message Passing Library (MPL) wird weiterentwickelt. Ein Prototyp existiert unter dem Namen MPL/p, wobei hier ohne TCP/IP-Overhead auf den Switch der SP1 zugegriffen wird. Dadurch werden die Latenzzeiten von 100 μs auf 10 μs gedrückt. Im ersten Halbjahr 1994 sollte diese Version als Produkt zur Verfügung stehen [81].

2.7 Linda

Linda wurde Mitte der 80er Jahre von David Gelernter und Kollegen an der Yale University entwickelt.

Linda wird eher als koordinierende Sprache für Systeme mit verteiltem Memory beschrieben. Es gibt C–Linda, Eiffel–Linda, Linda für PVM, fehlertolerantes Linda, Linda für Shared Memory Systeme [4, 83, 97].

Für Shared Memory Systeme vertreibt die Firma Cogent Research (tim_moore@cogent.com) XTM (für Transputer) und YTM(für i860). CircL heisst ihre Version für Workstations.

Eine andere kommerzielle Version von Linda für vernetzte Workstations aber auch für IBM SP1 wird von der Firma SCA Scientific Computing Asociates, New Haven, Conneticut, (linda@sca.com) vertrieben.

Auch für Linda gibt es Informationen in der NETLIB. Eine E-Mail an `netlib@ornl.gov` sollte enthalten:

`send linda from faq`.

2.8 MPI - Message Passing Interface

Die Definition eines standardisierten Message Passing Interface (MPI) ist eine Initiative einer grossen Gruppe von MPP-Herstellern und -Benutzern. Mittlerweile gibt es einen Entwurf des Standards [86, 87] (siehe auch `info.mcs.anl.gov:/pub/mpi`). Die Forschungsarbeiten zur Schaffung des Standard werden von ARPA, NSF und der Europäischen Gemeinschaft (Projekt ESPRIT) gefördert. Die Firma ParaSoft hat die erste kommerzielle Implementierung des MPI-Standards angekündigt. In Argonne sind auch auf der IBM SP1 erste Tests mit einer Implementierung des MPI gemacht worden [50, 51]. Informationen über die MPI-Aktivitäten werden auch über die *netlib* distributiert (`send index from mpi`).

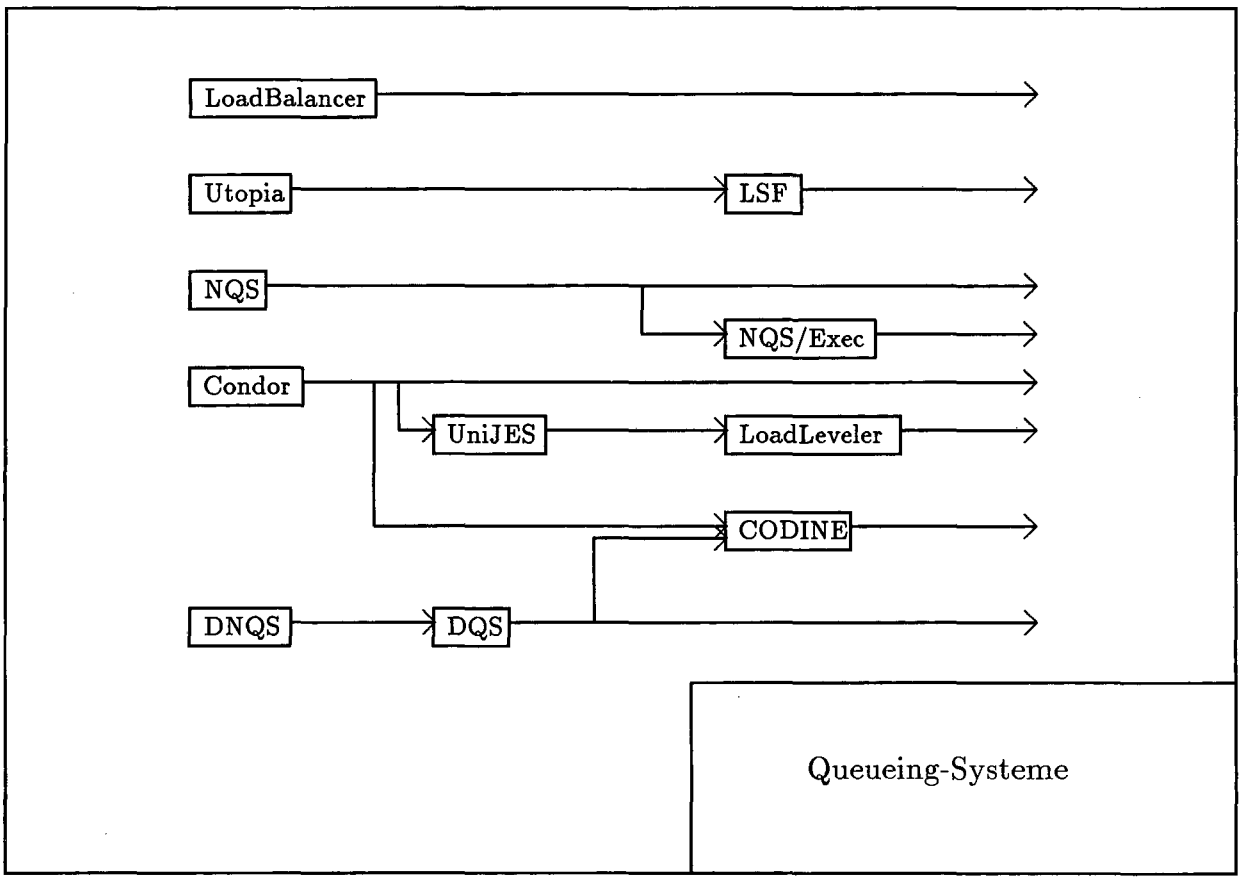
3 Queueingsysteme

In Mainframeumgebungen sind seit langer Zeit Batch–Systeme im Einsatz. Mit dem Aufkommen von Workstations und leistungsstarken Netzwerken in großen Stückzahlen möchte man heute freie Kapazitäten im Netz für rechenintensive Prozesse zu nutzen. Übersichten über Queueingsysteme werden in [116, 67] gegeben.

Was erwartet man von einem modernen Queueing System? Hier die wichtigsten Forderungen:

- Einfaches Einrichten und Betreiben von Queues für die sequentielle Abarbeitung eines Jobs auf einem beliebigen Rechner im Netzwerk
- Automatischer, dynamischer Lastausgleich im heterogen Netzwerk
- Unterstützung von parallelen Jobs
- Graphisches Interface für den Administrator, um die Gesamtauslastung eines Clusters zu beobachten
- Checkpoint/Recovery Mechanismen
- Jobmigration im Fehlerfall oder aus Lastgründen

Eine Übersicht über die aktuellen Queueingsysteme für heterogene Landschaften und ihre Verwandtschaft untereinander gibt die nachstehende Abbildung [67]:



3.1 NQS — Network Queueing System

Das Network Queueing System (NQS) wurde im Auftrag der NASA von der kalifornischen Firma Sterling entwickelt. Wie in den USA üblich wurden die öffentlich finanzierten Ergebnisse der Öffentlichkeit zur

freien Verfügung gestellt. Einerseits entwickelte Sterling daraus die kommerzielle Variante als Sterling NQS. Andererseits zirkuliert noch immer die Public Domain Version von NQS. Einige Computerhersteller entwickelten daraus ihre proprietäre Version (Convex mit CXBatch oder Cray im UNICOS) [12, 109, 110].

NQS kennt drei Arten von Queues:

- Device Queues
- Batch Queues
- Pipe Queues

Device Queues dienen zum reinen Serialisieren von Anforderungen an Geräte (z.B. Drucker, Plotter, Bandlaufwerke). Es können aber mehrere Devices pro Queue oder mehrere Queues für ein Device eingerichtet werden.

Batch Queues sind die eigentlichen Arbeitspferde. Es können je CPU unterschiedliche Queues eingerichtet werden (z.B. für kurze, mittlere oder lange Jobs [CPU-Zeit]). Kennt man die Memoryanforderungen der Jobs, kann man beispielsweise auch immer zwei Jobs gleichzeitig je CPU zulassen, wenn ein Job x MB RAM benötigt und die Maschine über 2 mal x MB RAM verfügt. Wesentliches Merkmal der Batch Queues ist, daß ihnen Quota Limits zugeordnet werden, die die sonstigen Einschränkungen der Benutzer außer Kraft setzen. So kann man auf einer Maschine interaktiv nur kurze und kleine Prozesse zulassen. Will der Benutzer grosse oder lange Jobs starten, muß er sie an das Queueing System abgeben, das dann für eine optimale und gleichmäßige Auslastung der Maschine sorgt.

Pipe Queues sind dagegen keine Senken wie Device und Batch Queues sondern Transitsysteme. Sie sind zum Transport von Requests zu Zielqueues notwendig. Damit können lokale und Queues in einem Netzwerk gefüttert werden. Allerdings ist im nackten NQS die Zuordnung statisch, so daß der Benutzer eine bestimmte Pipe Queue beschicken muß, um zu einer bestimmten Batch Queue zu gelangen. Ein Lastausgleich ist daher nur manuell auf Benutzerebene möglich.

Das Hinzufügen, Enablen, Disablen und Entfernen von Queues ist ebenfalls ein statischer Vorgang, den der Systemadministrator vornehmen muß [69].

Die PD–Version von NQS war früher auf dem Rechner `archive.ann.edu.au` als `pub/src/applications/nqs.tar.Z` zu finden. Uwe Untermarzoner, IBM, hat in Deutschland eine Version auf den Rechner `129.69.1.12` in `soft/aix/3.1` gelegt. Es kann auch über COSMIC, The University of Georgia, 382 East Broad Street, Athens, GA 30602–4272 bezogen werden.

Die kommerzielle Variante wird von in Deutschland von der Sterling Software GmbH, Vagendastraße 19, W–4000 Düsseldorf 30, vertrieben.

3.2 CERN/NQS

Christian Boissat (`boissat@vxcern.cern.ch`) vom CERN hat das NQS durch ein Load Balancing ergänzt zum CERN/NQS. Dies wurde erreicht durch die Einführung eines zentralen Dispatchers und Modifikationen des Pipeclient–Programms. Dedizierte Load–Balancing–Queues unterstützen den Vorgang.

Unter dem Namen NQS++ ist weiterhin ein Benutzerinterface geschaffen worden, daß es VM– und VMS–Benutzern ermöglicht, Jobs an ein UNIX–NQS–System zu schicken.

Das CERN/NQS unterstützt als Plattformen AEGIS, ULTRIX, HP–UX, AIX, MACH (Next), TC–IX, IRIX und SUN–OS. CERN/NQS ist frei verfügbar über anonymen Ftp vom Rechner `shift.cern.ch` im Directory `pub/NQS`.

3.3 NQS/Exec

NQS/Exec ist über das ursprüngliche NQS hinaus in der Lage, ein automatischen Lastausgleich herbeizuführen. Dies wird erreicht durch den Einsatz der Komponente „network computing executive"(NCE) der Firma The Cummnings Group (TCG). Auf jeder Maschine muß ein NCE–Client laufen und auf einer zusätzlich ein NCE–Server [109, 110].

Mit der graphischen Oberfläche `xnqs` können Jobs gestartet, beobachtet und abgebrochen werden. Zudem kann die Lastverteilung im gesamten Cluster angezeigt werden. Ärgerlich ist, daß Batchjobs nur

auf dem Rechner abgebrochen (`qdel`) werden können, auf dem die Batchqueue residiert, in der der Job läuft.

NQS/Exec ist für AIX, HP–UX, SunOS und SOLARIS verfügbar und wird auch von der Sterling Software vertrieben (`nqs@sterling.com`).

3.4 DQS — Distributed Queueing System

DQS (Distributed Queueing System) wurde am Supercomputer Computations Research Institute der Florida State University von Tom Green und Jeff Snyder entwickelt [103, 48, 49]. Einige Zeit hieß es auch DNQS. Um Verwechselungen zu vermeiden, wurde es DQS genannt. Neben der FSU wird es auch intensiv am NCSA (National Center for Supercomputer Applications) eingesetzt. Auch die Open Computing Facility (OCF) des Lawrence Livermore Laboratorys fährt auf einigen Knoten ihres RS/6000–Clusters DQS.

Als Design Ziel wurde der Einsatz in Workstation Farms oder Clustern ins Auge gefasst. Neben dem Betrieb von einzelnen Queues mit womöglich unterschiedlichen Prioritäten kennt DQS Gruppen, mit deren Hilfe mehrere Queues auf unterschiedlichen Prozessoren geschedult werden können. Damit ist es auch möglich, daß der Qmaster Resourcen freihält, um zum Beispiel PVM-Jobs optimal laufen zu lassen.

Mit dem Programm `qidle` ist es möglich, daß Queues suspendiert werden, wenn ein Benutzer interaktiv an einer Workstation arbeitet. Mithilfe einfacher X–Window–Programme ist es möglich, den Workstationcluster zu beobachten und zu verwalten.

Mittlerweile wird die Entwicklung von DQS vom Department of Energy (DoE) gefördert und es wird in der zweiten Hälfte 1993 mit einer neuen Version gerechnet. Darin soll dann neben PVM auch P4 unterstützt werden. Remote Cells werden eingeführt, um Jobs an andere administrative Domänen absetzen zu können. fertiggestellt werden.

Zur Zeit werden AIX, SunOS, IRIX, UNICOS, ConvexOS und OSF/1 direkt unterstützt. Neben NFS kann auch mit AFS gearbeitet werden.

DQS ist auf dem Rechner `ftp.fsu.edu` im Directory `pub/DQS` in der jeweils neuesten Fassung abholbereit. Für Anregungen und Probleme gibt es die E–Mailadresse `dqs@scri.fsu.edu`.

3.5 DNQS — Distributed Network Queueing System

Martin–Daniel Lacasse (`isaac@physics.mcgill.ca`) von der McGill Universität in Montreal, Kanada, hat das alte DNQS von Peter Green übernommen und völlig neu implementiert [71].

Lee Busby aus der X–Division des Lawrence Livermore National Laboratory berichtet, daß er DNQS auf einem Cluster von 11 HP Workstations laufen hat. Die Entscheidung fiel auf DNQS nachdem QBATCH, HP Task Broker, NQS, SCRI–DNQS und McGill–DNQS evaluiert worden waren [12]. Allerdings ist DQS nicht mit in die Evaluierung einbezogen gewesen.

DNQS ist über anonymen Ftp auf dem Rechner `ftp.physics.mcgill.ca` verfügbar.

3.6 VQS — Vienna Queueing System

Das Rechenzentrum der Universität Wien betreibt seit Februar 1992 einen Cluster mit sechs RS/6000–Workstations, die über Ethernet miteinander verbunden sind (später dann FDDI). Über einen der Rechner wird auf die anderen zugegriffen. Alle sind als klassische Batchrechner konfiguriert. Mit Hilfe des Vienna Queueing Systems (VQS) werden die Batch Jobs auf die einzelnen Rechner verteilt und ein Lastausgleich herbeigeführt (Graeff in [1], [79, 80]).

3.7 CONDOR

CONDOR wurde ursprünglich an der University of Wisconsin, Madison, entwickelt und von der IBM an Workstationgegebenheiten anpaßt (Pederson in [1]).

CONDOR ist eine verteiltes, netzwerkbasiertes Job–Scheduling–System für Workstations. Es können Batch Jobs, parallele Batch Jobs und interaktive Jobs geschedult werden. Dabei kann jede Workstation als Submitter, als Compute Server oder beides über ein einfaches Xwindows Interface vom Benutzer oder vom CONDOR–Verwalter konfiguriert werden.

Miron Livny berichtet [1], daß CONDOR an der University of Wisconsin seit mehr als vier Jahren im Einsatz ist und zur Zeit mehr als 250 Workstations kontrolliert, mit denen über 30 Forscher unterschiedlichster Disziplinen arbeiten.

Carolyn Gard von der Universität von Georgia zählt zu den Vorteilen von CONDOR das Einrichten von Job–Klassen nach Memory, CPU–Zeit und temporärem Plattenplatzbedarf. Ein Load–Balancing ist dadurch möglich. Sie weist auch darauf hin, daß der Vorteil von relativ preiswerten CPU–Zyklen erkauft wird durch einen Anstieg der Personalkosten auf der Systemverwalterseite, wobei sehr gut ausgebildete System–Manager gebraucht werden. Die Arbeiten werden in einer gemeinsamen Studie der Universität von Georgia und der IBM durchgeführt auf einem RS/6000–Cluster, der von NQS zu CONDOR migriert ist [42].

3.8 IBM LoadLeveler

IBM hat aus CONDOR ein eigenes Produkt entwickelt, das unter dem Namen UniJES an einigen Stellen evaluiert wurde [124]. Unter der Produktbezeichnung IBM LoadLeveler vertreibt die IBM ein Job Scheduling System für verteilte Ressourcen. Zunächst werden IBM 9076 Systeme (Scalable POWERparallel System), RS/6000–Cluster und Workstations unterstützt [61].

Mithilfe einer graphischen Oberfläche kann der Systemadministrator flexibel Jobklassen und Queues definieren nach Speicher, CPU oder Zeit. Er kann die Schedulingregeln definieren und die Queues steuern und überwachen. Das gesamte System kann dynamisch rekonfiguriert werden.

Für den Benutzer gibt es ebenfalls eine graphische Oberfläche, mittels derer er Jobs beschreiben und starten kann.

Der IBM LoadLeveler hat Mechanismen für Checkpoints/Restarts und Job Migration implementiert. Daneben gibt es noch folgende Features:

- Versionen für die Plattformen anderer Hersteller (SUN, SGI)
- Neben seriellen Batch Jobs können auch parallele Jobs geschedulet werden
- An NQS–Knoten können Jobs submitted, abgefragt und gecancelt werden.

3.9 Load Balancer

Ein weiteres Queueingsystem ist Load Balancer [67, 41]. Ein graphisches Interface auf X-Windows-Basis existiert noch nicht. Parallele Jobs werden nicht unterstützt.

Load Balancer ist verfügbar für SunOS, SGI IRIX, HP-UX, IBM AIX, Ultrix und OSF/1. Es wird vertrieben durch Freedman Sharp and Associates, Inc., 508-1011- 1st Street SW, Calgary Alberta, Canada T2R 1J2, Tel: (403) 264-4822, FAX: (403) 264-0873, E-mail: `lb@fsa.ca`.

3.10 Utopia LSF (Load Sharing Facility)

Aus Utopia [125], einem Forschungsprojekt der Universität Toronto, ist eine kommerzielle Version entstanden: LSF (Load Sharing Facility)[67, 125, 126].

LSF unterstützt an Parallelsystemen PVM und TCGMSG sowie einige im industriellen Umfeld entstandene. Es ist verfügbar für ConvexOS, DEC OSF/1, Ultrix, HP-UX, IBM AIX, SGI IRIX, SunOS und Solaris.

Utopia LSF wird von der Firma Platform Computing Corporation, 203 College Strret, Suite 303, Toronto, Ontario M5T 1P9, Kanada, Tel: 416-978-0458, FAX: 416-978-0878, E-Mail: `info@platform.com` vertrieben.

3.11 CODINE

CODINE (COmputing in DIstributed Networked Environments) ist aus DQS und Condor entstanden. Checkpointing und Jobmigration sind integriert [67].

Als kommerzielles Produkt ist es verfügbar für ConvexOS, DEC OSF/1, Ultrix, HP-UX, IBM AIX, SGI IRIX, SunOS und Solaris. CODINE wird von der Genias Software GmbH, Erzgebirgsstr. 2 B,

D–93073 Neutraubling, Tel: ++49 +9401 9200-0, FAX: ++49 +9401 9200-92, (mailbox@genias.de) vertrieben.

4 Numerik

[23, 25, 8, 35, 75, 123, 56, 5]

4.1 Lineare Algebra

4.1.1 Lineare Gleichungssysteme - Direkte Methoden

4.1.1.1 LU Faktorisierung

Am Beispiel einer LU-Faktorisierung zeigen Filippone und Sales [35], daß durch Tunen ein hohes Maß an paralleler Effizienz möglich ist. In einem Cluster von RS/6000-580, die mit einem Serial Optical Channel Converter (SOCC, 240 MBit/s) verbunden sind, wird bei 8 Prozessoren eine Effizienz von 90 % und mit 4 Prozessoren sogar 96 % erreicht.

4.1.1.2 Choleski Faktorisierung

In einer komplexen hermitischen Cholesky–Faktorisierung werden 95 % (bei 8 Prozessoren) und 99 % (bei 2 Prozessoren) von Filippone und Sales [35] gemessen. Damit wird praktisch kein Overhead durch die Netzwerkkommunikation verursacht. Allerdings beruhen die Ergebnisse auf einer getunten Version PVMe. PVMe ist eine Erweiterung des Standard–PVMs, die am European Center for Scientific and Engineering Computing (ECSEC) der IBM in Rom, Italien, entwickelt wurde und schnelle Kommunikationsverbindungen wie den SOCC (Serial Optical Channel Converter) unterstützt. PVMe war von der IBM frei erhältlich.

4.1.2 Lineare Gleichungssysteme - Iterative Methoden

[5]

4.1.2.1 Conjugierte Gradienten Methode - CG

Jones und Plassmann berichten, daß sie einen general-purpose parallelen, iterativen Solver für große, dünnbesetzte Lineare Gleichungssysteme entwickelt haben. In zwei Anwendungen (piezoelektrische Kristallschwingungen und einem Supraleitermodell) zeigen sie die Skalierbarkeit auf einem Intel DELTA. Auf 512 Knoten wurden über 480.000 Gleichungen mit 161.150.990 Nonzeros gelöst, wobei eine Leistung von ungefähr zwei Gigaflops gemessen wurde. Der Solver benutzt eine unvollständige Matrixfaktorisierung als Präkonditionierer für den Conjugierten Gradienten Algorithmus [66].

[75]
[16]
[6]
[117]

4.1.2.2 SVD - Singular Value Decomposition

[8]

4.1.2.3 Conjugate Gradient Squared - CGS

[5]
[68]

4.2 FFT - Fast Fourier Transformation

4.3 Partielle Differentialgleichungen

4.4 Finite Element Methoden

4.5 Standardpakete

In den siebziger Jahren kam der Wunsch auf, in numerisch intensiven Anwendungen einerseits Portabilität durch den Einsatz von Fortran beizubehalten, andererseits für ständig wiederkehrende numerische Probleme, Standardbibliotheken zu entwickeln, die in optimaler Weise auf den Zielprozessor optimiert sind, aber dem Programmierer immer dieselbe Schnittstelle bieten, so daß seine Portabilität nicht gefährdet ist. Zudem soll durch den Einsatz von Standardbibliotheken die Stabilität und Robustheit sowie die Effizienz eines Programms sichergestellt werden.

Für Vektor-Vektor-, Matrix-Vektor- und Matrix-Matrix-Operationen wurde die BLAS (Basic Linear Algebra Subroutines) eingeführt [73]. Später kamen LINPACK (zur Analyse und Lösung von linearen Gleichungssystemen und Least Square Problemen) und EISPACK (zur Berechnung von Eigenwerten und Eigenvektoren) hinzu [25]. LINPACK und EISPACK wurden in LAPACK (Linear Algebra Package) zusammengefasst und modernisiert [25].

Allen Paketen ist gemein, daß sie nur für sequentielle Rechner existieren. Mit dem Aufkommen von Parallelrechnern ist daher auch der Wunsch entstanden, die weitverbreiteten Pakete auch dort zur Verfügung zu haben. Daraus entstanden die Bemühungen, mit Hilfe von BLACS (Basic Linear Algebra Communication Subroutines) [26, 27] und ScaLAPACK [25] auch auf den neuen Rechenern, die alte Software weiterverwenden zu können bzw. neu zu erstellende portabel zu halten.

4.5.1 BLAS - Basic Linear Algebra Subroutines

Die BLAS Routinen sind in drei Leveln organisiert:

- Vektor–Vektor Operationen
- Matrix–Vektor Operationen
- Matrix–Matrix Operationen.

[52]

4.5.2 BLACS - Basic Linear Algebra Communication Subprogramms

[25, 26, 27]

4.5.3 LINPACK und EISPACK

LINPACK ist eine Sammlung von Fortran Unterprogrammen zur Analyse und Lösung von linearen Gleichungssystemen und Least Square Problemen. Das Paket löst lineare Systeme mit allgemeinen, gebänderten, symmetrisch indefiniten, symmetrisch positiv definiten, triangulären und tridiagonal quadratischen Matrizen. Linpack ist organisiert um vier Matrixfaktorisierungen herum: LU Faktorisierung, pivotisierte Cholesky Faktorisierung, QR-Faktorisierung und Single-Value-Dekomposition[25].

EISPACK ist eine Sammlung von Fortran Unterprogrammen zur Berechnung von Eigenwerten und Eigenvektoren in neun Klassen von Matrizen: allgemeine komplexe, komplex hermitische, allgemeine reelle, reell symmetrische, gebänderte reell symmetrische, reell symmetrisch tridiagonale, spezielle reelle tridiagonale, generalisierte reelle und generalisierte reell symmetrische Matrizen. EISPACK wurde 1972 freigegeben [25].

LINPACK und EISPACK sind designt, um BLAS Level 1 Routinen zu benutzen.

4.5.4 LAPACK

[25, 22]

LAPACK ist designt, um BLAS Level 3 Routinen zu benutzen.

4.5.5 BLACS - Basic Linear Algebra Communication Subprogramms

[25, 26, 27]

4.5.6 ScaLAPACK

ScaLAPACK wird für Ende 1994 erwartet, eine Vorabversion soll schon Ende 1993 über die *netlib* zur Verfügung gestellt werden [25].

4.6 Kommerzielle Numerikbibliotheken

Mit der wachsenden Verbreitung von Parallelplattformen gehen auch die Hersteller von kommerziellen Numerikbibliotheken dazu über, ihre Produkte auf diesen Plattformen anzubieten.

4.6.1 IMSL

IMSL hat 1991 begonnen, einzelne Unterprogramme zu parallelisieren. Zunächst war eine Version für SGI–Shared–Memory–Systeme verfügbar.

4.6.2 BLADE

BLADE (Basic Linear Algebra in Distributed Environments) ist ein Produkt des IBM European Center for Scientific and Engineering Computing (ECSEC) in Rom, Italien. Es ist eine FORTRAN–Unterprogrammbibliothek für Probleme der Linearen Algebra (reelle und komplexe Matrix Multiplikation, reelle und komplexe LU Faktorisierung und Lösung, reelle symmetrische und komplexe hermitische Cholesky Faktorisierung, reell–symmetrische Band– und Skyline–Cholesky Faktorisierung).

Voraussetzung sind AIX 3.2, XLF, ESSL und PVM. Auch getunte PVMe–Versionen können eingesetzt werden (auf Ethernet, Token Ring, FDDI und SOCC). Auch eine SP1 wird unterstützt.

5 Parallellisierte Anwendungspakete

5.1 Strukturmechanik

5.1.1 DPAM: Distributed Parallel Application Manager

MSC/NASTRAN

5.1.2 PAM-CRASH

Engineering System International Group (ESI) hat angekündigt, daß sie ihr PAM-CRASH, das hauptsächlich in der Automobil- und Luftfahrtindustrie zur Crashanalyse eingesetzt wird, auch auf die IBM SP1 portieren wollen [64].

5.1.3 RADIOSS

RADIOSS ist ein weiteres Paket zur Crashanalyse. Es wird von MECALOG SARL hergestellt und auf die IBM SP1 portiert[64].

5.1.4 MARC

MARC Analysis Research Corporation hat ihr FEM-Paket MARC für die IBM SP1 verfügbar [64].

5.1.5 LS-DYNA3D

Livermore Software Technology Corporation.

Für Cray T3D [18] und IBM SP1 [81] verfügbar.

5.1.6 PVSOLVE

5.2 Computational Fluid Mechanics (CFD)

[43, 44, 120, 98, 30]

5.2.1 FIRE

FIRE ist ein CFD-Paket, das beispielsweise in der Automobilindustrie zur Auslegung von Motoren und Karosserien verwendet wird. AVL List GmbH hat angekündigt auf der Supercomputing 93, daß sie FIRE auf die IBM SP1 protieren wollen [64].

5.2.2 FLO67

T3D

5.3 Computational Chemistry

5.3.1 AMBER

Jim Vincent (vincent@retina.chem.psu.edu) hat eine Zusammenfassung gepostet, welche Arbeitsgruppen an einer Parallelversion von AMBER arbeiten (mit einer Ergänzung von Sophie Creuzet (`sophiec@frmop11.cnusc.fr`)):

Plattform	Personen	Institutionen
SIMD Board für PC	D. Evans	Visionary Systems
SGI R3000 (unconfirmed)	M. Berger, R. Gomperts	SGI
Indigo und RS/6000 (PVM)	T. Lybrand	U of Washington, Seattle
nCube	S. DeBolt, P. Kollmann	UCSF
IBM SP1	S. Paternello	`paternel@vnet.ibm.com`
Cray T3D		Cray

In [19] wird angegeben, daß das Pittsburgh Super Computer Center Mitte 1993 damit begonnen hat, AMBER auf die T3D zu portieren. Zunächst wurde der T3D-Emulator auf der Cray C916 benutzt, um die 20.000 Zeilen Code (von insgesamt 160.000), die 90-95 % der Rechenzeit verbrauchen, zu parallelisieren. Es wird damit gerechnet, daß Anfang bis Mitte 1994 Vergleichszahlen für die T3D vorliegen [19].

Martin [81] gibt für die IBM SP1 mit High Performance Switch folgende Werte für AMBER an:

N	Zeit	Speedup
1	7092	1,00
4	2130	3,33
6	1587	4,47
8	1308	5,42
10	1206	5,88

Diese Zahlen beziehen sich auf einen AMBER-MINMD-Lauf mit D. Vulgaris Rubredoxin in einer Wasserbox mit periodischen Randbedingungen. Das System besteht aus 7749 Atomen. Der Cutoff ist auf 10 Å gesetzt, was zu ca. 1,6 M non-bon Paaren führt. Der Job wurde für 400 Minimierungszyklen gerechnet gefolgt von 800 Dynamik-Schritten [81].

5.3.2 CHARMM

S. Fleischman (`sfleisch@hydra.convex.com`), Convex Computer Corporation, berichtet [1], daß er eine auf PVM–basierende Parallel–Version des Molekül–Dynamik–Paketes CHARMM (Chemistry at HARvard Macromulecular Mechanics) auf einem Cluster aus HP–Workstations durchgemessen hat, wobei er auch vergleichende Performance–Messungen zwischen Ethernet und FDDI vorgenommen hat.

Milan Hodoscek (milan@helix.nih.gov) weist darauf hin, daß er CHARMM23 auf verschiedene parallele Plattformen portiert hat, worüber er und B.R. Brooks in Chemical Design Automation News, Volume 7, Number 12, December 1992, unter dem Titel „Parallelization of CHARMM for MIMD Machines" zu lesen war (siehe auch [84]). Er gibt in CCL folgende Zahlen an:

Maschine	Anz. Proz.	Speedup	Bemerkungen
Intel Gamma	32	28	
Intel Gamma	128	80	
HP/730	2	1.9	
HP/730	4	3.7	FDDI
IBM RS/6000-320	4	3.1	Ethernet

Auch für Cray T3D [18] ist CHARMM verfügbar.

Martin [81] gibt für die IBM SP1 mit High Performance Switch für CHARMM (MbCO) folgende Werte an:

N	Zeit	Speedup
1	20196	1,00
4	6402	3,15
6	5232	3,86
8	4214	4,79
16	2475	8,16

5.3.3 COLUMBUS

COLUMBUS ist ein MRSDCI–Paket (Multireference single– and double–excitation configuration interaction) [108, 76]. Die paralellisierte Fassung bedient sich der TCGMSG–Subroutines aus Argonne. Eine frei verfügliche Fassung ist auf dem Rechner ftp.tcg.anl.gov im Directory pub/columbus bzw. pub/tars über anonymen Ftp erhältlich.

5.3.4 DISCO

DISCO wurde an der ETH Zürich von Lüthi grob granular parallelisiert für Cluster von Cray-Maschinen und Workstationcluster. Bei letzteren wurde ein Speedup über 10 bei 18 CPU's erreicht. Eine Portierung auf PVM existiert.

Pettersson (lgm@vand.physto.se) von der Universität Stockholm und Faxen (faxen@alliant.com) von der Firma Alliant berichten, daß sie große Metall–Cluster in einer parallelisierten Fassung des SCF–Paketes DISCO gerechnet haben haben [99]. Es kam die tcgmsg–Bibliothek auf einer Alliant Campus/800 MPP mit 200 Prozessoren zum Einsatz.

IBM gibt an, daß eine parallelisierte Version von DISCO (NCSAdisco) im vierten Quartal 1994 für die SP1 verfügbar ist [82].

5.3.5 DISCOVER

DISCOVER ist ein moleküldynamisches Paket der Fa. BIOSYSM und liegt auch in einer parallelisierten Version u.a. auch für IBM SP1 [82], nCUBE2 und KSR1 vor.

5.3.6 DGEOM

DGEOM ist ein Programm, daß herausfindet, ob molekulare Strukturen konsistent mit eingebenen Entfernungsrandbedingungen sind. Als Monte-Carlo-Simulation gehört es zu den „Embarrassingly Parallel" Programmen.

Mattson [84, 85] berichtet, daß es als pDGEOM parallelisiert worden ist mit C-Linda und tcgmsg. Der Aufwand für Linda war recht gering. Die rechenintensiven Module waren in Fortran und wurden

mit einem C-Wrapper an Linda angebunden. In [84] werden einige Messergebnisse mit Cyclosporin A auf Clustern von IBM RS/6000 und SGI gezeigt.

Auch für die IBM SP1 ist DGEOM verfügbar [82].

5.3.7 DMOL

DMOL ist ein Dichte-Funktional-Code. Bei BIOSYM wir an einer MPP Version gearbeitet. Auch an der Northwestern University wird DMOL parallelisiert.

5.3.8 GAMESS-US

Das Department of Chemistry der Iowa State University (Mark Gordon und Mike Schmidt `mike@si.fi.ameslab.gov`) pflegt das quantenchemische Programmpaket GAMESS (General Atomic and Molecular Electronic Structure System). Dabei können die SCF–Wellenfunktion und Energiegradienten parallel berechnet werden. Dazu kommen die TCGMSG–Routinen aus Argonne zu Einsatz. Theresa Windus (`theresa@si.fi.ameslab.gov`) weist aber darauf hin, daß auch eine PVM-Version existiert.

Das Programmers Reference Manual vom 17.6.92 weist für Beispielrechnungen auf 5 DECstations einen parallelen Speedup von 3,48 (70 %) und 3,98 (80 %) aus.

Für Intel Delta, iPSC, CM-5, nCUBE, Cray T3D [18] und IBM SP1 liegen [81, 82] Portierungen vor.

GAMESS-US ist frei verfüglich und kann durch eine E-Mail an `mike@si.fi.ameslab.gov` bestellt werden.

5.3.9 GAMESS-UK

GAMESS-UK hat sich aus der amerikanischen Version (NRCC, 1981) entwickelt und wird vom Science and Engineering Council (SERC) Daresbury Laboratory gepflegt. An der Weiterentwicklung arbeitet ein internationales Konsortium (Guest, van Lenthe, Kendrick und Schoeffel [`ksc@fsitu1.hre.hydro.com`]).

Die GAMESS-UK-Funktionalität ist verteilt auf mehrere Module: Hartree-Fock, Elektronen-Korrelation, Dierect-SCF und Direct-MP2, Molekulare Eigenschaften, Pseudopotentiale, Visualisierung, Density Functional Theory, Hybrid QM + MM, und semiempirische Methoden. Davon sind die konventionellen und die direkten SCF-Rechnungen parallelisiert.

Parallele Versionen sind verfügbar für Intel iPSC/860, Meiko-Rechner und Workstation Cluster (HP und IBM).

Vertrieben wir GAMESS-UK durch Computing for Science Ltd., SERC Daresbury Laboratory, Warrington, WA4 4AD UK, Tel: +44 925 603 240, FAX: +44 925 603 634.

5.3.10 GAUSSIAN

An einer parallelisierten Version vom GAUSSIAN-Code wird an der Wayne State University von Schlegel gearbeitet. Welchen Einfluß das auf die kommerzielle Version hat, ist nicht bekannt.

5.3.11 GROMOS

Ganesan Ravishanker (`ravishan@swan.wesleyan.edu`) von der Wesleyan University in Middletown, Conneticut, schreibt in CCL, daß er unter dem Namen Wesdyn eine Parallelprotierung von GROMOS86 auf Network Linda basierend vorgenommen hat. Er gibt Speedups von 30 % für 2 Knoten und 50 % für 3 Knoten an.

5.3.12 HONDO

HONDO liegt in einer parallelisierten Fassung von der IBM für die SP1 vor und wird zum Beispiel am Cornell Theory Center in der Version 8.4 basierend auf PVM 2.4.2 eingesetzt.

5.3.13 MOPAC

Kim Baldridge vom San Diego Supercomputer Center arbeitet an einer parallelisierten Fassung von MOPAC.

5.3.14 SPARTAN

ParNet ist eine Programmierbibliothek (FORTRAN und C), die auf PVM 2.4 aufsetzt und für SGI–Maschinen erstellt wurde. Als Beispiel für die Möglichkeiten wurden Messungen an einer parallelisierten SPARTAN–Version vorgenommen.

Bei R4000–Uniprozessormaschinen wurden parallele Effizienz von 95 % (5,74 Speedup bei 6 Prozessoren, Single Point Direct SCF) gemessen und bei SGI–Mehrprozessormaschinen 68 % (13,55 Speedup bei 20 Prozessoren, 6 Systeme, Gradient Direct SCF) [47]. Die Autoren sind über `parnet@boston.sgi.com` erreichbar.

Martin [81] gibt für einen SPARTAN-Lauf mit 21 Atomen, 211 Basisfunktionen in einer direkten SCF-Rechnung folgende Zahlen (Zeiten in Sekunden) für eine IBM SP1 und eine SGI 150 MHz Challenge an:

N	SP1 Zeit	SP1 Speedup	SGI Zeit	SGI Speedup
1	3062	1,00	5077	1,0
4	821	3,73	1293	3,93
8	458	6,69	706	7,19
12	350	8,75	496	10,2
16	306	10,00	415	12,2
20			367	13,8
24			315	16,1

5.3.15 SUPERMOLECULE

Jan Almlof an der University of Minnesota hat eine Parallelversion von SUPERMOLECULE erstellt. Zumindest für Cray T3D ist sie verfügbar [18].

5.3.16 TURBOMOLE

TURBOMOLE ist ein Programmpaket für konventionelle und direkte SCF–Methoden, das vor allem an der Universität (TH) Karlsruhe von Reinhart Ahlrichs entwickelt wurde. Mit Unterstützung der Firmen BASF, HOECHST und BAYER wurde eine Parallelversion für Workstationcluster (TCP/IP–basierend) von Stefan Brode (BASF), Michael Ehrig und Hans Horn (beide TH Karlsruhe) erarbeitet [10, 11].

Ahlrichs führt aus, daß er an einem RS/6000–Cluster mit einer SCF/Gradienten-Rechnung an einem sehr großen Molekül (164 Atome, 1576 Basisfunktionen) eine Effizienz von 75 % erzielt hat (18 Workstations, paralleler Speedup 14). Die jetzige Implementierung hält er einsetzbar für bis zu 30 Workstations [3]. Zur Zeit wird an einer Portierung auf PVM gearbeitet.

TURBOMOLE wird vertrieben von der Fa. BIOSYM Technologies GmbH, München.

5.3.17 XPLOR

Im dritten Quartal 1993 soll XPLOR in einer parallelisierten Fassung für die IBM SP1 vorliegen [82]. Auch an der Yale University wird an einer Parallelisierung gearbeitet.

5.4 Hochenergiephysik

5.5 Meteorologie und Klimatologie

5.5.1 PCCM2 - Parallel Community Climate Model

PCMM2 (Parallel Community Climate Model) ist eine auf Message Passing parallelisierte Version des Community Climate Models 2, das am National Center for Atmospheric Research (NCAR) und in Argonne entwickelt wurde. Im Rahmen der Department of Energy Initiative CHAMPP (Computer Hardware, Advanced Mathematics and Model Physics) wird das Paket zur Klimavoraussage verwendet. PCMM2 basiert auf PICL [50].

5.5.2 MPMM - Massively Parallel Mesoscale Model

In einer Prototyp-Studie wurde das MM4 (Mesoscale Model) grobgranular parallelisiert auf einem Intel Touchstone Delta (Caltech). Mit 12 i860-Prozessoren wurde etwa die halbe Leistung einer Cray Y-MP erreicht [39].

In einem zweiten Schritt wurde das MPMM (Massively Parallel Mesoscale Model) als feingranulare Dekomposition des an der Pennsylvania State University und am NCAR erstellten Mesoscale Model Version 5 entwickelt. Mit dem Programm wird eine Echtzeitvorhersage und Klimavoraussage durchgeführt [39]. Der numerische Kern besteht aus einem Finiten Differenzen Schema vierter Ordnung. Die Parallelisierung basiert auf PCN [51], wobei einerseits ein 2-D-Gitteri des Modells auf ein 2-D-Prozessor-Gitter gemappt wird und andererseits durch PCN eine dynamische Lastverteilung auf die Prozessoren ermöglicht wird [39]. Das Paket wurde auf der Supercomputing 93 auf IBM SP1 gezeigt.

6 Programmierwerkzeuge

6.1 Codeanalyse und Parallelisierung

6.1.1 Forge 90

Forge ist eine interaktive Umgebung für FORTRAN77–Programmierer, die es ermöglicht, existierende FORTRAN–Programme zu analysieren (Baseline System) und für verschiedene Archietekturen zu optimieren (Vektorrechner, Shared Memory Systeme und Systeme mit verteiltem Memory).

Mit dem Baseline System [36] können Variable getracet werden, Konstanten und COMMON Blöcke beobachtet und eine Datenflußanalyse gemacht werden.

Für die unterschiedlichen Architekturen gibt es dann spezielle Module, zum Beispiel für vernetzte Workstations den Distributed Memory Parallelizer. Als Message Passing Bibliothek wird hier dann PVM unterstützt [37].

Forge 90 wurde von der kalifornischen Firma Applied Parallel Research entwickelt. In Deutschland wird es vertrieben von Genias Software GmbH, Erzgebirgsstr. 2 B, D–93073 Neutraubling, Tel: ++49 +9401 9200-0, FAX: ++49 +9401 9200-92, (`mailbox@genias.de`).

6.1.2 MAGIC

Auf der Supercomputing '93 in Portland wurde der automatische Parallelisierer MAGIC von der Firma Applied Parallel Research vorgestellt. Auf der Basis einer statischen Analyse werden Datenfelder partitioniert und Loops verteilt. MAGIC ist für Shared und Distributed Memory Systeme im Zusammenhang mit FORGE verfügbar.

6.2 Compiler

6.2.1 High Performance Fortran – HPF

Im High Performance Fortran Forum haben sich Anwender und Anbieter zusammengeschlossen, um einen High Performance Fortran Standard zu definieren. Vertreter US–amerikanischer Hochschulen, Großforschungseinrichtungen, Computer– und Softwarehersteller und numerisch intensive Anwender haben sich getroffen, um den bestehenden Fortran 90 Standard um notwendige Sprachmittel zu ergänzen. Aber auch einige europäische Hochschulen (Stuttgart, Wien, Delft) und Institutionen (GMD, ZIB) sind vetreten.

Ziel ist es „data parallel programming" und Höchstleistung auf MIMD und SIMD Maschinen durch Erweiterungen des Sprachstandards zu erreichen [60].

Es wird erwartet, daß noch 1993 HPF–Compiler auf den Markt kommen, die zumindest HPF–Programme präprozessieren und zu FORTRAN 77 zerlegen mit entsprechenden Laufzeitbibliotheken.

Für die IBM SP1 hat die Fa. Applied Parallel Research, Inc. (APR) die Verfügbarkeit ihres Produktes Forge 90 sowie einen „xHPF77 Batch Parallelizer" angekündigt.

6.2.2 Fortran M

6.2.3 PCN

PCN (Program Composition Notation) [38, 50, 51] ist ein System zur Entwicklung und Ausführung paralleler Programme. In einer C-ähnlichen Syntax werden parallele Konstrukte unterstützt. Es sind Fortran- und C-Schnittstellen vorhanden, um bestehende Programme einbinden zu können. Die Kommunikation basiert auf TCP/IP-Sockets und rsh-Mechanismen zum Starten von Programmen. Zur Unterstützung wird der parallele Debugger PDB herangezogen. Zur Laufzeitanalyse wird Gauge und Upshot eingesetzt. PCN ist ein gemeinsames Forschungsprojekt von Caltech und Argonne.

6.3 Parallele Debugger

6.3.1 TotalView

Bolt, Baranek an Newman

6.4 Laufzeitanalyse

6.4.1 ParaGraph

Oak Ridge (Trace Informationen visualisieren)

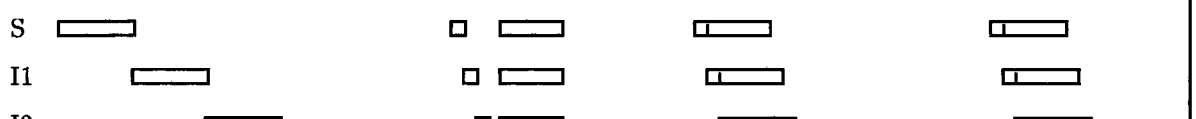
7 Koppelnetzwerke

Die Leistungsfähigkeit von Parallelrechnern hängt wesentlich von der Leistungsfähigkeit des Koppelnetzwerkes zwischen den Prozessoren ab. Man möchte Netzwerke mit niedriger Latenzzeit und hoher Bandbreite. Um dies zu erreichen, werden verschiedene Switching-Technologien, Topologien, Routing-Politiken und auch unterschiedliche Beaufschlagungen des Netzwerkes eingesetzt.

Zu den heute verwendeten Switching-Technologien gehören [91, 92]:

- Store-and-forward Switching
- Circuit-Switching
- Virtual-cut-through
- Wormhole Routing

Durch die verschiedenen Technologien ergeben sich bei sonst freiem Netz unterschiedliche Latenzzeiten, wie aus der folgenden Abbildung ersichtlich ist [91, 92]:



	Store and Forward	Circuit Switching	Virtual Cut Through	Wormhole Routing
Latency:	L/B*D	(LC/B)*D+L/B	(LH/B)*D+L/B	(LF/B)*D+L/B

L Packetlength
B Bandwidth
D Pathlength
LC Length of Control Packet
LH Length of HeaderField
LF Length of Flit

iPSC-2
iPSC/860

NCube2
Paragon
J-Machine MIT
T3D
SP1

Die verbreitetsten Topologien sind heute [91, 92, 34]:

- Ring (KSR1, Convex MPP-1)
- Hypercubes (nCUBE-2)
- N-Cubes
- 2D-Gitter (Parsystec GCel [T805], Paragon XP/s
- 3D-Gitter (Parsytec GC [T9000], Cray T3D)
- Multistage Networks (CM-5, CS-2, SP1)
- Crossbarswitch (VPP 500)

8 Koppelnetzwerke für Workstation-Cluster

8.1 IBM Vulcan Switch

Der IBM Vulcan Switch ist ein Wormhole Router [91, 92], das heißt daß eine Wegeentscheidung für ein einkommendes Paket schon nach wenigen Bits getroffen wird und das Paket ohne Zwischespeicherung weitergeleitet wird. Über Microchannelkarten können bis zu 16 CPU's gekoppelt werden. Der Vulcan Switch findet primär in der IBM SP1 Verwendung. In den USA sind aber schon erste Standalone-Versionen zur Koppelung von Workstations im Einsatz.

8.2 IBM Allnode Switch

Der IBM Allnode Switch ist ein Circuit-Switch-Router, das heißt das erste Paket wird dazu genutzt eine Verbindung aufzubauen, auf der dann alle folgenden Pakete transportiert werden können. Damit ist der Switch langsamer als ein Wormhole-Router. Mit dem IBM Allnode Switch können bis zu 16

Workstaions über Microchannel-Karten angeschlossen werden. Der Switch wird als OEM-Produkt auch als CompuNet-Allnode-Switch angeboten. Der Switch kann über eine eigene Bibliothek angesprochen werden oder über TCP/IP. Auch für PVM/6000 [63] und Express existieren Driver. Als Prototyp wurde er Ende 1992 vorgestellt [17].

8.3 BIT3 Switch

Der BIT3 Switch von der gleichnamigen Firma ermöglicht es, bis zu 4 RS/6000 über Shared Memory zu koppeln. Intern ist er mit einem VME-Bus-Backplane ausgebaut, dass Karten zum Anschluß der RS/6000 (dort im Microchannel) aufnimmt sowie eine Logik zum Zugriff auf das Memory hat. Es existieren Driver für TCP/IP und Express.

Adresse: BIT3 Computer Corp., 8120 Penn Avenue South, Bloomington, MN 55431-1393 USA, Tel.: (612) 881-6955, FAX: (612) 881-9674.

8.4 DEC Gigaswitch

Der DEC Gigaswitch ist ein Gerät zum Anschluss von FDDI-Knoten [21]. In seinem Innern befindet sich ein Crossbar-Switch, so daß mehrere Punkt-zu-Punkt-Verbindungen gleichzeitig betrieben werden können. Er wird z.B am am Pittsburgh Super Computer Center eingesetzt, um 14 Alpha Systeme zu koppeln. White et al. [122] haben mit PVM auf SGI-Maschinen Messwerte in der Nähe der theoretischen FDDI-Bandbreite erzielt.

8.5 IBM SOCC Serial Optical Channel Converter

Mit 240 Mbit/s arbeitet der Serial Optical Channel Converter (SOCC) der IBM. Mit ihm können drei Workstations direkt miteinander verbunden werden. Nimmt man den Network Systems DX Router hinzu, können 16 Workstations verbunden werden.

9 MPP–Systeme

9.1 MasPar

Die Firma MasPar stellte 1990 ihren Parallelrechner MP-1 (MP-1216) vor. Es handelt sich um einen SIMD-Rechner (Single Instruction Multiple Data) mit 16384 4-bit-Prozessoren. 1992 folgte der MP-2 (MP-2216) mit ebenfalls 16384 Prozessoren. Als Preise werden 260.000 US-$ bis 1,6 Millionen US-$ angegeben. Als Wirtsrechner wird eine DECstation 5000 eingesetzt [101, 102].

Das Koppelnetzwerk ist als Multistage Crossbar Interconnection mit Routern ausgelegt, wobei als Topologie ein 2D-Gitter gebildet wird. Jeder Knoten ist mit seinen acht nächsten Nachbarn direkt verbunden. MasPar bezeichnet diese Topologie als X-Net. Prechelt gibt folgende Leistungszahlen an [102]:

	Unit	MP-1	MP-2
Raw computation			
32 bit Integer	MIPS	26000	68000
32 bit Floating Point	MFLOPS	1200	6300
64 bit Floating Point	MFLOPS	550	2400
Memory bandwidth			
direct addressing	MB/s	11000	20000
indirect addressing	MB/s	4000	7800
Communication			
xnet bandwidth	MB/s	23000	20000
router bandwith	MB/s	1300	1300

Der Benutzer sieht als Betriebssystem nur das Ultrix des Vorrechners. Fortran und C sind mit datenparallelen Erweiterungen vorhanden. Auch kann die MPL (MasPar Application Language) benutzt werden.

Installationen: University of Bergen, Norway; Melbourne University; Ames Lab, Iowa State University; The Institute for Genomic Research; University of California, Irvine; University of Tennessee; Harvard University; Universität Karlsruhe (MP-1).

9.2 Meiko Computing Surface 2

PE mit VP, Sun SPARC und 2 Fujitsu microVP, 32-128 MB RAM 64-1024 Knoten Elan Network Interface Fat Tree

9.3 Convex MPP

9.4 Fujitsu VPP500

9.5 NEC Cenju-3

VR4400sc Prozessoren, 32-64 MB RAM 8-256 Prozessoren NEC Multistage crossbar network

9.6 Kendall Square Research KSR1

Seit 1991 bietet Kendall Square Research ihren Rechner KSR1 an (in Deutschland auch über SIEMENS). In den Processing Elements finden sich proprietäre Prozessoren (20 MHz), die mit 32 MB RAM ausgerüstet sind. Der Prozessor ist mit einem 64-bit Floating-Point-Rechenwerk ausgestattet. An einen Knoten kann ein I/O-Adapter (30 MByte/s) angeschlossen werden [9, 31, 104].

Das Koppelnetzwerk besteht aus einem Ring mit einer Übertragungsleistung von 1 GByte/s. Bis zu 32 Prozessoren können an einen Ring angeschlossen werden. Mehrere Ringe können über einen weiteren Ring zusammengefasst werden. Bis zu 1088 Prozessoren können in einem System vorhanden sein (34 Ringe mit jeweils 32 Prozessoren). Das Koppelnetzwerk unterstützt Virtual Shared Memory in Hardware. Durch dieses Design ergibt sich eine Memory-Hierarchie mit unterschiedlichen Zugriffszeiten [9, 31, 104]:

	[9] Zyklen	[31] Zyklen	[104] Zyklen
lokaler Subcache	2	2	2
lokaler Cache	20	18	18
gleicher Ring	130	126	175
entfernter Ring	570	600	600

Zur Vermeidung von Bottlenecks wird in der Memoryverwaltung Prefetch und Poststore angeboten [31, 104].

Als Betriebssystem wird KSR OS eingesetzt, einer Variante von OSF/1. Fortran77, C und C++ stehen zur Verfügung. Als Paralleliserungshilfsmittel wird der Präprozessor KAP eingesetzt. PRESTO ist eine parallele Laufzeitumgebung, die über Compilerdirektiven im Sourcecode angesteuert wird. Als Message Passing Bibliotheken werden TCGMSG und PARMACS unterstützt.

Installationen: Cornell Theory Center; Leibniz Rechenzentrum München (32); Universität Mannheim; GMD Birlinghoven.

9.7 Intel Paragon XP/S

Seit vielen Jahren bietet die Firma Intel über ihre Suprecomputer Systems Division Parallelrechner an. Ältere Modelle wie der iPSC/2, iPSC/860 und Touchstone Delta haben eine gewisse Verbreitung, so daß an vielen Stellen hiermit Betriebserfahrungen vorliegen [7, 33].

Das aktuelle Modell ist der Intel Paragon XP/S. Es basiert auf dem Intel i860 XP (50 MHz). In jedem Processing Element werden zwei i860 eingesetzt, einer als Application Processor und einer als Message Processor. Jedes Processing Element kann mit 16 oder 32 MB RAM ausgerüstet sein. Es gibt

drei Type von Knoten: Compute Knoten, Service Knoten und I/O Knoten, wobei die Knoten durch einfaches Zustecken von Adaptern ihre Funktion ändern können. Bis zu 1024 Knoten können zu einem Gesamtsystem integriert werden [33].

Das Koppelnetzwerk basiert auf dem Message Routing Chip iMRC mit fünf Eingängen und fünf Ausgängen. Der Router arbeitet als Wormhole-Router [91, 92] mit einem FLIT-Buffer von 16 Bit. Die Kanäle sind 16 Bit breit und haben eine Bandbreite von 200 MByte/s. Mit diesem Chip wird ein 2D-Gitter als Topologie realisiert (im Gegensatz zum Hypercube des iPSC/860). Es wird ein deterministisches Routing eingesetzt: zunächst werden Nachrichten in horizontaler Richtung, dann in vertikaler Richtung geroutet. Neben dem Datennetzwerk gibt es noch ein Diagnose-Netzwerk. Aus einer Programmiersprache heraus sind Latenzzeiten von 30 μs realisierbar [33].

Als Betriebssystem wird OSF/1 eingesetzt. Wie schon beim iPSC/860 kommt als Message Passing Library NX zum Einsatz. Die Processing Elements werden partitioniert, wobei auch Partitionen für interaktives Arbeiten vorgesehen werden. Die Belegung der Partitionen wird durch *Gang Scheduling* geregelt. Um System Ressourcen zu managen, wird das *Multi-User Accounting, Control and Scheduling (MACS)* System des San Diego Supercomputer Centers verwendet. Für Batch-Jobs steht NQS bereit [33, 9].

Neben Message Passing werden auch High Performance Fortran (HPF) und Shared Virtual Memory (SVM) unterstützt. Darüber hinaus sind auch PVM, EXPRESS, PARMACS und TCGMSG einsetzbar. Als Programmierhilfsmittel stehen (neben den Compilern für Fortran 77, HPF, C, C++ und Ada) Forge90, der Interactive Parallel Debugger (IPD), prof860, das Performance Visualisation System (PVS) und ParaGraph zur Verfügung [33, 9].

Als mathematische Bibliothek stellt Intel die CLASSPACK von Kuck and Associates bereit. In ihr sind BLAS Routinen (Level 1-3), FFT Routinen und Löser für tridiagonale und pentadiagonale lineare Systeme. SEGlib ist eine Signal Processing Library von Intel. Als parallele Bibliothek stellt Intel ProSolver bereit für ausgewählte Probleme aus der Linearen Algebra [33, 9].

Installationen: KFA Jülich.

9.8 TMC CM-5

[57, 88, 9]

Sparc-Prozessoren mit vier Vector Units von Texas Instruments

Fat Tree Topologie

9.9 Parsytec GigaCube

Neben den kleineren Parallelrechnern MC [68] und SC [16], die auf dem Inmos T800-Prozessor basieren, bietet die Firma Parsytec aus Aachen einen GigaCube (GC) an, der aus Inmos T9000 Prozessoren gebaut wird. Jeder T9000 bietet 25 MFLOPS Numerik Leistung. Jeweils 17 Prozessoren werden zu einem Prozessorcluster zusammengefasst. Vier Prozessorcluster bilden einen Cube und bis zu 256 Cubes können zusammengeschlossen werden [57, 65].

Für das Koppelnetzwerk kommen Inmos C104 Kommunikations Chip mit Wormhole Routing [91, 65] zum Einsatz. In jedem Cluster werden vier C104-Chips zur Kommunikation eingesetzt.

9.10 Cray T3D

Von der Firma Cray Research Inc. wurde Ende 1993 der Parallelrechner T3D vorgestellt. Er basiert auf DEC-Alpha-Chips (150 MHz). Jeder Knoten kann mit 16 oder 64 MB RAM ausgestattet werden. 32 bis 2048 Processing Elements können zu einem System zusammengefasst werden [93, 100].

In folgenden Stufen wird das System angeboten [18]:

Procs	Peak GFLOPS	Price Mio $
32	4,8	2,2
64		
128		
256		
512		
1024	153,6	31,0
2048	307,2	

Bis 128 Prozessoren kann das System luft- oder flüssigkeitsgekühlt betrieben werden, darüber muß mit Flüssigkeit gekühlt werden.

Das Koppelnetzwerk basiert auf Motorola Kommunikations Chips, die ein Virtual-Cut-Through bzw. Wormhole-Routing ermöglichen. Das Netzwerk arbeitet mit der gleichen Taktfrequenz wie die Prozessoren. Als Latenzen für das Netzwerk werden Zeiten $< 1\mu s$ angegeben. Latency Hiding und Virtual Shared Memory sollen unterstützt werden. Das Koppelnetzwerk ist als toroidales 3D-Gitter ausgeprägt. Das Routing im Koppelnetzwerk geschieht dimensionorientiert, das heißt, zunächst wird ein Weg in x-Richtung, dann in y-Richtung und zuletzt in z-Richtung gesucht. Durch die Verwendung von Softwareladbaren Routingtabellen, können fehlerhafte Knoten isoliert werden [93].

Das Gesamtsystem braucht einen Cray-Vorrechner (Y-MP oder C90), über den die PE'S mit Betriebssystem, Benutzerprogrammen und Benutzerdaten versorgt werden. Letztere werden zusätzlich zum Interprozessverkehr auf das Koppelnetzwerk gegeben [93].

Auf den PE's wird ein abemagertes UNIX gefahren (UNICOS MAX, eine MACH Variante). I/O wird über den Wirtsrechner gesteuert. High Performance Fortran (HPF) wird von Cray derzeit nicht angeboten. Stattdessen wird ein proprietäres MPP Fortran eingesetzt. Zur Programmentwicklung wird der Debugger TotalView von der Firma Bolt, Baranek and Newman (BBN) eingesetzt. Zur Performanceanalyse steht das Werkzeug MPP Apprentice zur Verfügung. An Message Passing Bibliotheken unterstützt Cray PVM und PARMACS. Als Queueing-System wir ein angepasstes NQS verwendet [93].

Anwendungen: CFD (FLO67); Chemie (AMBER, CHARMM, GAMESS-US, SUPERMOLECULE); Combustion (FIRE); Environment (POP); Ingenieur (LS-DYNA3D).

Installationen: Pittsburgh Supercomputer Center (32 Proz., 512 Prozessoren Anfang 1994).

Bestellungen: NASA Jet Propulsion Laboratory/Caltech; Ecole Polytechnique de Lausanne; Arctic Region Supercomputer Center; ECMWF, European Center for Midrange Weather Forcast, Reading, GB (64 Proz. Mitte 94 an bestehende Y-MP 2E); Minnesota Supercomputer Center (128); CEA Commissariat a l'Energie Atomique, Division des Apllications Militaires (128 an bestehende M92).

9.11 IBM SP1

Das Gesamtsystem kann aus 1-4 Frames und 8-64 Prozessoren bestehen. Damit werden dann 1-8 GFLOPS Peak Performance eingebaut. An Memory können 0.5-16 GB vorhanden sein, während an Plattenplatz im System bis zu 128 GB integriert sein können (ohne Fileserver) [64].

Die Prozessor Knoten (maximal 64) bestehen aus RISC System/6000 Prozessoren (62.5 MHz) mit einer Peakleistung von 125 MFLOPS. Jeder Knoten fährt eine volle AIX Version. Es können 64-256 MB Memory eingebaut sein sowie 0, 1 oder 2 GB Festplatte vorhanden sein. Die Knoten können diskless oder dataless betrieben werden. Jeder Knoten ist mit zwei Ethernet-Adptern ausgestattet. Optional kann ein High Performance Switch eingesetzt werden. An jedem Knoten können Adapter für FDDI, S/390 BMCA (Block Multiplexor Channel Adaptor) oder SCSI angeschlossen werden. An Bandbreiten zum Fileserver gibt IBM an [64]:

Ethernet	1.25 MB/sec
FDDI	12.50 MB/sec
FCS	25.00 MB/sec
PCA	4.50 MB/sec

Für den High Performance Switch (Multi-Stage mit Wormhole Routing) werden 40 MB/s Peakbandbreite bidirektional pro Verbindung angegeben. 500 ns beträgt die Hardware-Latenz [64].

An jedem SP1 System müssen zusätzlich eine Kontrollworkstation und ein Fileserver angeschlossen sein. Die Kontrollworkstation dient zum Systemmonitoring und zur Administartion. Von ihr werden RS232-Leitungen zu den Frames geführt (neben Ethernet). Der Fileserver wird über NFS und BOOTP betrieben und dient je nach Konfiguration als Bootdevice (diskless und dataless), Fileserver für Anwendungen und Benutzer [64].

Ein Einstiegmodell mit 8 Prozessoren und High-Performance Switch, 1 GB Festplatte je Knoten, 64 MB RAM je Knoten, AIX, IBM LoadLeveler, AIXwindows, Systemmanagement und Systemmonitorsoftware kostet 329.200 US-$ Listenpreis [64].

Anwendungen: Math. Bibliotheken (BLADE); Chemie (AMBER, BATCHMIN, CHARMM, DISCOVER, DGEOM, DMOL, GAMESS, HONDO, NCASdisco, PROLSQ/PROTIN, SPARTAN, XPLOR); Ingenieur (LS-DYNA, DPAM, PVSOLVE); Elektronik (S-PISCES, S-SUPREM4, THUNDER, VWF, TFT, MINSIM, CHIP WIRING); Seismik (MIGPACK).

Datenbanken (DB2/6000, Oracle, Sybase) und Transaktionsmonitore (CICS/6000, Tuxedo, Encina) werden zur Zeit portiert sowie SAP und Quantum Leap [64].

Installationen: GMD Birlinghoven (8); Rensellaer Polytechnic Institute; Cornell Theory Center (64); Argonne National Laboratory (128); Universität von Hong Kong; Samsung, Korea; Universität von Mexico; IBM T.J. Watson Research Center (256); Maui High Performance Computing Center, Hawaii (64).

Das Cornell Theory Center betreibt seit Frühjahr 1993 eine SP1, die im Laufe des Jahres 1993 auf 64 Prozessoren aufgerüstet wurde. Von der National Science Foundation sind 13 Mio. US-$ bereitgestellt worden, um in 1994 eine 512-Prozessor-Version zu evaluieren.

Zollweg [127] vom Cornell Theory Center hat einige erste Erfahrungen berichtet. Zum einen ist der Übergang von PVM zu MPL recht leicht gefallen. Das Parallel Environment sei einfach zu benutzen und es ermutigt (wie MPI auch) das SPMD-Modell (Single Program Multiple Data). Er erwähnt auch das EUIH (External User Interface), daß die Kommunikation zwischen den Prozessoren ohne IP-Overhead vollständig im Benutzeradressraum abhandelt. Zum Vergleich gibt er für eine Anwendung folgende Zahlenwerte (Zeiten in Sekunden) [127]:

PE IP auf Ethernet	136,60
PVM Ethernet	128,10
PVM IP auf Switch	75,44
PE lsp	49,06
PVMe	48,86
EUIH	48,47
EUIH interrupt	45,02

Die US Air Force wird in Hawai das Maui High Performance Computing Center errichten. Betrieben wird das Center von der Universität von New Mexico, Albuquerque. Das Projekt hat einen Gesamtumfang von 26 Mio. US-$, wobei beginnend im Frühjahr 1994 der Ausbau des Parallelrechners in drei Stufen erfolgt. Eine Reihe anderer Supercomputer Center und Hochschulen sind ebenfalls beteiligt.

Das Argonne National Laboratory, Illinois, und IBM haben einen 3,6 Mio. US-$ Forschungs- und Entwicklungsvertrag geschlossen, um das massiv-parallele Supercomuting in kommerziellen und wissenschaftlichen Anwendungen voranzutreiben. Zum Einsatz wird eine 128-Prozessorversion kommen, an die eine 6,4-TB-Tape-Library angeschlossen wird, die IBM mit Ampex entwickelt hat. Weitere Partner hier sind im Umweltbereich AlliedSignal, Amoco, DuPont, Pacific Nortwest Laboratory und Phillips Petroleum Co. Das NASA Lewis Research Center wird hier Antriebsprobleme rechnen. Das Lawrence Berkeley Laboratory, die Universität von Illinois in Chicago, die Universität von Maryland und Argonne werden Probleme aus dem Bereich der Hochenergiephysik studieren. Das Illinois Institute of Technology wird sich Anwendungen aus den Bereichen Imaging und Turbulenzmodellen widmen. Die Universität von Chicago möchte sich mit der Bilddatenverarbeitung in der Medizin, geophysikalischen und astrophysikalischen Problemstellungen beschäftigen. Unter der Schirmherrschaft des U.S. Councils for Automotive Research (USCAR) werden Chrysler, Ford und General Motors Verbrennungsstudien und Materialforschung durchführen.

Erste Erfahrungen mit der IBM SP1 in Argonne werden in [50, 51] berichtet. Dabei kamen an parallelen Tools BlockSolve, Chamaeleon, Fortran M, MPI, PCN, PETSc, PRISM und P4 zum Einsatz. In den Feldern Elektromagnetik, Wettervorhersage, Nukleare Strukturen, Klimamodellierung, Phylogenetik und Supraleitung wurde der Einsatz des Systems erprobt. Als besonders positiv hervorgehoben wurde, daß an jedem Knoten ein vollen UNIX (AIX 3.2.4) zur Verfügung steht, daß an jedem Knoten relativ viel Memory (128 MB) ist und daß alle Anwendungen leicht portierbar waren.

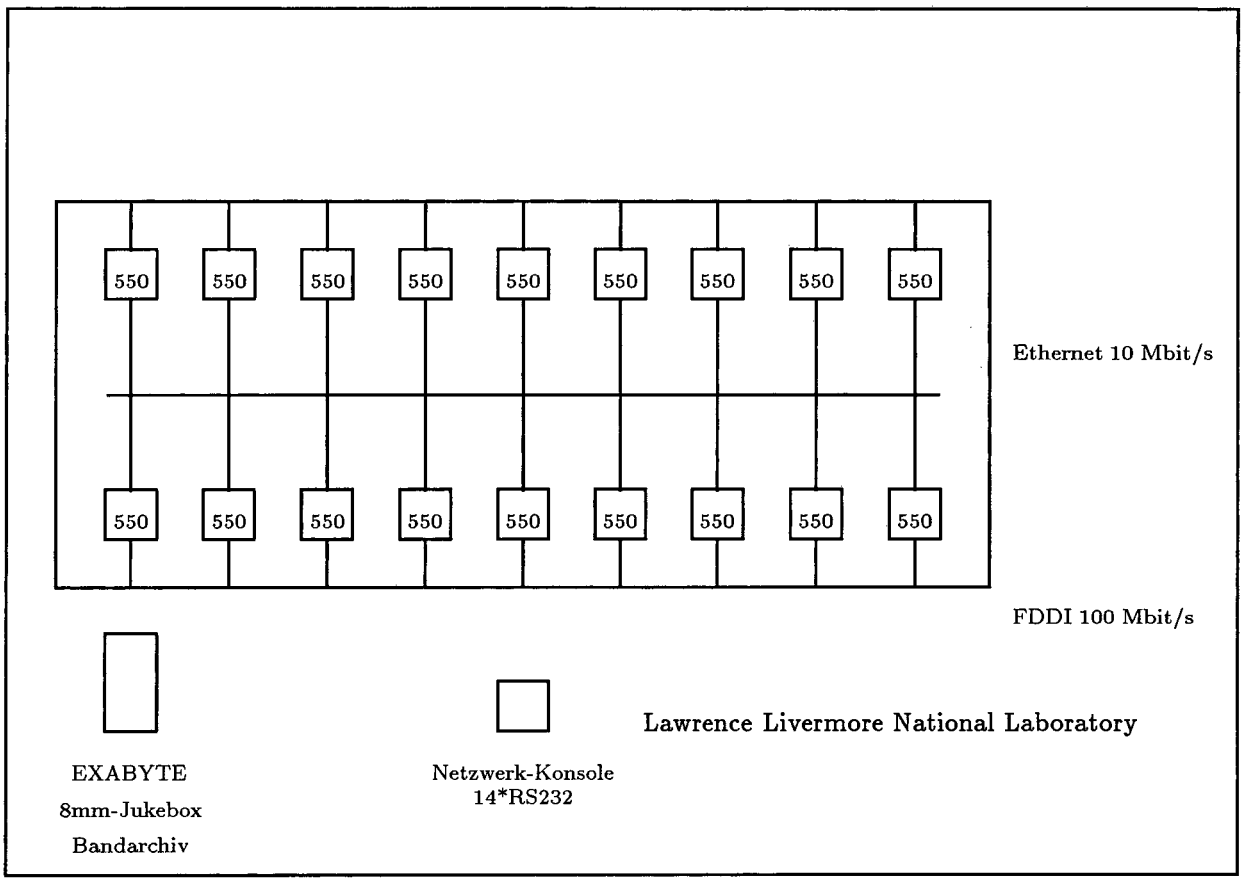
Die zukünftige Entwicklung der SP-Linie sieht für 1994 vor [82]:

- Integration der POWER2-Architektur
- Erweiterung auf 512 Knoten
- Shared Common Memory
- verbesserter Switch
- verbesserte Switch-Adapter
- vergrößerte Flexibilität durch fette und dünne Knoten
- FCS (Fiber Channel Standard) I/O
- Parallel I/O
- Debugger und Visualisierer für HPF
- Unterstützung von UNITREE

10 Beispiele ausgewählter Cluster

10.1 Lawrence Livermore Lab

Am Lawrence Livermore National Laboraty in Livermore, Kalifornien, wird seit 1991 ein Cluster von 18 IBM RS/6000 Modell 550 in der Open Computing Facility eingesetzt. In Summe ergibt sich damit eine numerische Peakleistung von 1,8 GFlops. Alle Maschinen sind mit 128 MB RAM und 1,8 GB Festplatten ausgestattet. Für besondere Anforderungen sind zwei Maschinen mit 512 MB RAM ausgerüstet. Der Cluster ist sowohl mit Ethernet als auch mit FDDI vernetzt. Als Konsole dient eine Workstation, die mit 18 RS232–Anschlüssen versehen ist. Als großer Massenspeicher dient ein EXABYTE–Bandarchiv, das als 8mm–Jukebox ausgeprägt ist [113].



Um den Cluster im Batchbetrieb zu nutzen, wird NQS von Sterling eingesetzt. Auf einzelnen Knoten ist auch DQS getestet worden. Der LoadLeveler von IBM wird ebenfalls evaluiert. Zur Parallelverarbeitung wird PVM eingesetzt.

Das Bandarchiv wurde zeitweise mit dem Legato Networker zu Backup–Zwecken betrieben.

10.2 Los Alamos National Lab

Im Los Alamos National Lab sind 16 IBM RS/6000 Modell 560 zu einem Cluster verknüpft. Netzwerke werden mit unterschiedlichen Bandbreiten eingesetzt: Ethernet (10 Mb/s), FDDI (100 Mb/s) und SOCC (Serial Optical Channel Converter mit 240 Mb/s). An Message–Passing–Routinen wird PVM eingesetzt. Unter anderem werden auf diesem Cluster SCF–Rechnungen durchgeführt.

10.3 Universität Oslo

Die Universität Oslo hat sich im Jahre 1992 einen Cluster mit IBM RS/6000 Maschinen zugelegt. Als Server sind zwei Modelle 980 im Einsatz und als weitere Knoten 14 Modelle 580. Die Server sind mit 128 MB RAM und 10 seriellen Plattenlaufwerken 9333 ausgestattet. Unter anderem werden im Rahmen von Industriekooperationen auch quantenchemische und moleküldynamische Probleme gerechnet.

10.4 Cornell University

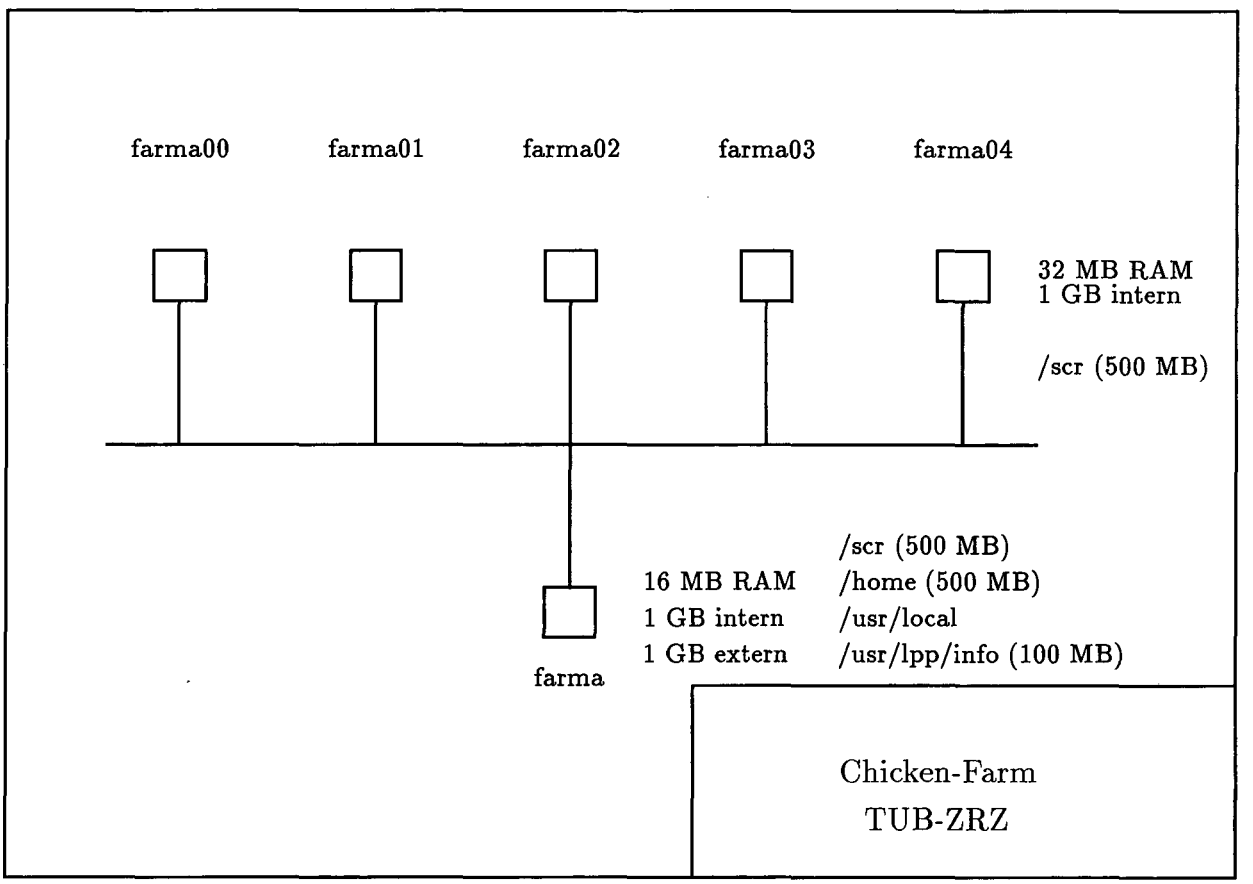
An der Cornell University, Ithaca, New York, sind vielfältige Parallel– und Batch–Aktivitäten im Gange. So betreibt das Cornell Material Science Center einen Cluster von 30 IBM RS/6000 Maschinen (früher mit NQS, jetzt mit DQS).

Das Cornell Theory Center veranstaltete im Frühjahr 1993 einen Workshop „Introduction to Distributed Memory Parallel Computing using Parallel Virtual Machine (PVM)". Eine der ersten SP1–Modelle von IBM wurde im Frühjahr 1993 an das Cornell Theory Center ausgeliefert.

10.5 Technische Universität Berlin, ZRZ

In der Zentraleinrichtung Rechenzentrum der Technischen Universität Berlin ist ein Workstationcluster eingerichtet, der für numerisch intensive Aufgaben als reiner Batchcluster betrieben wird.

Der Cluster besteht aus fünf IBM RS/6000 Modellen 350 mit jeweils 32 MB RAM und 1 GB Festplatte und einem Modell 220 mit 16 MB und 2 GB Festplatte als Vorrechner. Die Rechner sind über Ethernet verbunden. Die Maschinen sind alle in einem 19–Zoll–Schrank untergebracht.



Als Queueing System kommt CERN/NQS zum Einsatz, das einen automatischen Lastausgleich ermöglicht. Die geplante Arbeitsweise sieht vor, daß Benutzer interaktiv auf dem Vorrechner ihre Jobs vorbereiten und Ergebnisse sichten, während auf den fünf Rechenelementen nur serieller Batchjobverkehr stattfindet.

Die Rechenelemente sind als selbständige Individuen konfiguriert. Das bedeutet, daß jedes Element eine eigene Kopie des Betriebssystems AIX hat sowie wie C– und FORTRAN–Compiler ausgerüstet ist. Auf jeder Maschine ist auch eine Kopie von NQS/Exec. Jedes Element besitzt auf der lokalen Festplatte einen 500 MB großen Scratch–Bereich (`/scr`), auf dem zur Laufzeit der Nutzerprogramme Zwischendateien abgelegt werden können. Die Benutzerhomedirectories stehen auf allen Elementen über NFS zur Verfügung.

10.6 Universität Wien, Rechenzentrum

Im Rechenzentrum der Universität Wien wird ein RS/6000-Cluster als interaktiver und Batch-Cluster ohne Parallelverarbeitung betrieben. Der Cluster besteht aus sechs Modellen 550 (64 MB RAM und 1,6 GB Platte) als Compute-Elementen und einem Modell 340 mit 8 GB Platten und zwei 8-mm-Bandlaufwerken als Fileserver. Die Knoten sind über Ethernet vernetzt. Als Batchsystem wird das Vienna Queueing System (VQS) eingesetzt [79, 80].

10.7 CERN, PIAF

PIAF ist eine Farm von fünf HP 755 Workstations. Jede Workstation hat 128 MB Memory und 8 GB RAID–Platten. Die Plattensubsyteme lassen eine Bandbreite von 5 MByte/s zu. Die fünf Workstations sind untereinander und mit zwei Fileservern über FDDI verbunden.

Über Ethernet wird auf den Cluster von Benutzern zugegriffen. Das hauseigene Visualisierungssystem ist mithilfe von PVM parallelisiert worden [20].

11 Benchmarks

[72]

11.1 LINPACK

Einer der wichtigsten Indikatoren für die Leistung eines Einprozessorsystems ist seit Jahren der LINPACK-Benchmark von Jack Dongarra [24]. Dabei wird ein lineares Gleichungssystem gelöst mit einer dicht besetzten Matrix. Der Standard–Benchmark wir mit einer 100*100 Matrix durchgeführt. Bei diesem Benchmark darf der Quellcode nicht verändert werden. Seit einiger Zeit wird zusätzlich unter der Rubrik TPP (Towards Peak Performance) auch die Leistung für eine 1000*1000 Matrix angegeben, wobei hier der Quellcode optimal an das ausführende System angepaßt werden darf.

Da aber diese Benchmark nicht geeignet sind, die Leistung von Parallelsystemen zu beurteilen, wurden zwei weitere Rubriken eingeführt. Zum einen wird unter "A Look at Parallel Processing" die 1000*1000 Leistung für Parallelrechner angegeben und zum anderen unter "Highly Parallel Computing" die gemessene Spitzenleistung aufgeführt, wobei neben der Anzahl von Prozessoren auch die Grösse des Problems angegeben wird.

Die jeweils aktuelle Version des Berichts [24] kann über E-Mail von `netlib@ornl.gov` angefordert werden: `send performance from benchmark`.

11.2 NAS (Numerical Aerodynamic Simulation) Benchmarks

Vom NASA Ames Research Center wurde ein Benchmark entwickelt, der besser die Bedürfnisse der Anwender in der Lösung der "Grand Challenges" wiederspiegelt. Unter dem Namen Numerical Aerodynamic Simulation (NAS) werden acht Benchmarks in zwei Gruppen durchgeführt [70, 81].

Die erste Gruppe wir als "Kernel Benchmarks" bezeichnet und besteht aus den Modulen:

- EP (Embarrassingly Parallel)
  2^{28} Iterationen werden in einem Loop gemacht, in dem zwei Zufallszahlen generiert und getestet werden.
- MG (Multi-Grid)
  Auf einem $256 \times 256 \times 256$ Gitter wird in vier Iterationen eine Lösung des diskreten Poisson Problems $\nabla^2 u = v$ approximiert.
- CG (Conjugate Gradient)
  Aus einer symmetrischen, positiv definiten und dünn besetzten Matrix der Ordnung 14.000 wird der kleinste Eigenwert berechnet. [75]
- FT (3D Fast Fourier Transfrom Partial Differential Equation)
  Um eine dreidimensionale partielle Differentialgleichung zu lösen, wird eine Fast Fourier Transformation auf einem $256 \times 256 \times 256$ komplexen Feld durchgeführt.
- IS (Integer Sort)
  2^{23} Integer Schlüssel werden sortiert.

Die zweite Gruppe heißt "Simulated Application Benchmark" mit den Modulen:

- LU (LU Solver)
  Regular-sparse, block lower/upper triangular system

- SP (Scalar Pentadiagonal)
  Mehrere unabhängige Systeme von nicht-diagonal-dominierten, scalaren, pentadiagonalen Gleichungen
- BT (Block Tridiagonal) wie oben für block-tridiagonale Gleichungen

White, Ålund und Sunderam [122] haben die Kernel Benchmarks auf PVM portiert und Messungen auf Workstation Clustern durchgeführt. Sie kommen zu dem Ergebnis, daß die Leistung einer Cray Y/MP-1 (die als Vergleichszahl herangezogen wird) schon mit wenigen zehn Workstations erreicht wird. Einige Ergebnisse [70, 122]:

Maschine	Proz.	EP	MG	CG	FT	IS	LU	SP	BT
CM-5	512	1,4							
CM-5	256	2,7							
CM-5	128	5,4	6,10	6,20	6,60	12,00	171,0	119,00	119,0
CM-5	64	10,9	10,90	10,60	7,90	24,20	272,0	170,00	175,0
CM-5	32	21,5	19,50	20,70	14,90	43,10	418,0	289,00	283,0
CM-5	16	42,4							
C-90	16	3,2	0,96	0,58	0,91	0,30	17,6	13,06	28,4
C-90	4	12,4	2,42	1,51	2,58	0,95	43,9	49,74	
C-90	1	47,6	8,65	4,56	10,28	3,70	157,6	184,70	356,9
Y-MP	8	15,9	2,96	2,38	4,19	1,85	49,4	64,60	114,0
Y-MP	1	126,2	22,22	11,90	28,77	11,46	333,5	471,50	792,4
Y-MP EL	8	72,7	22,30	13,70	18,50	12,69	351,6	488,40	764,1
Y-MP EL	4	141,2	27,94	19,00	27,90	18,51	522,3	601,90	1090,2
Y-MP EL	1	505,5	89,19	64,50	105,10	59,90	1449,0	2026,00	3832,8
T3D	124	4,8	1,75	3,74	2,33	1,75	55,4	82,80	78,9
T3D	64	9,7	3,03	6,14	4,53	3,42	102,0	159,00	153,7
T3D	32						196,2	314,30	330,2
SP1	64	6,1			7,40	6,63	102,5	121,50	152,6
SP1	32	11,9			13,30	6,74	170,8	200,50	273,0
SP1	16	23,6			25,40	7,96	295,2	329,60	524,0
SP1	8	46,9			48,10	11,00	516,1	563,00	884,4
Paragon	512	5,2							
Paragon	256	9,9							
Paragon	128	19,8	5,80	6,70	6,40	13,30	378,0	221,20	185,1
Paragon	64	40,4		8,40	12,80	15,10	523,8	277,40	240,1
Paragon	32					21,50			
iPSC/860	128	26,0	8,61	7,00	10,00	13,60	442,5	449,50	414,3
iPSC/860	64	51,0			21,00	17,00			
iPSC/860	32	102,0				26,00			
MP2	4k							615,00	789,0
MP1	16k			14,60		23,60			
MP1	4k			64,50			463,5	1772,00	2396,0
KSR1	128	18,1						150,00	134,5
KSR1	96	23,4						170,20	167,9
KSR1	64	34,9			8,40			228,80	239,4
KSR1	32	69,8	20,60	21,70	13,60	40,20	1041,3	377,70	439,0
SGI Gsw.	8		168,00		228,00	258,00			
RS6K FDDI	8		110,00		412,00	318,00			
SS1+ Enet	16			605,00					
SGI Gsw.	8	446,0	264,00		1070,00	674,00			
RS6K FDDI	8	442,0	229,00		645,00	770,00			
SS1+ Enet	16	1603,0		701,00					

Anmerkungen: Die Zahlenangaben für die einzelnen Benchmarks sind in Sekunden. IBM SP1 mit MPL/p. Workstations mit normalem PVM 3.2 und getunetem PVM. [64, 93, 70, 122]

11.3 NCAR Shallow Water Benchmark

Am National Center for Athmospheric Research (NCAR) ist der Shallow Water Benchmark entwickelt worden. Dabei werden im Rahmen der CHAMMP-Iniatiative die Methoden für die Klimatologie und Meteorologie verbessert. Beim Shallow Water Benchmark geht es um eine Finite-Differenzen-Lösung von zeitabhängigen 2D shallow-water Gleichungen, die bei ozeanischen und athmosphärischen Zirkulationsproblemen eingesetzt werden. Durch eine Ring-Dekomposition wird intensiv das Kommunikationsnetz gemessen, wobei im wesentlichen Nearest-Neighbor-Kommunikation vorliegt [70, 81, 29].

11.4 Genesis Distributed Memory Benchmarks

Diese Benchmark Suite besteht aus 12 Benchmarks in drei Kategorien [70]:

- Synthetische Code Fragmente, die Anwendungskerne repräsentieren oder Hardwareeigenschaften messen
  - COMMS1 Unidirektionaler Nachrichtentransfer (Pinpong)
  - COMMS2 Bidirektionaler Nachrichtentransfer (Pingping)
  - TRANS1 Transposition
  - SYNCH1 Barrieren Synchronisationskosten
- Anwendungskerne oder häufig benutzte Bibliotheksfunktionen
  - FFT1 Eindimensionale FFT
  - PDE1 3D Poisson Solver mit Red-Black Relaxation
  - PDE2 2D Multigrid Poisson Solver mit einer Spanne feiner und grober Gitter
  - QCD2 Konjugierter Gradienten Kernel aus QCD
- Anwendungen
  - QCD1 Quantenchromodynamik (Monte Carlo)
  - MD1 Molekulardynamik (Teilchen-Teilchen)
  - GR1 Allgemeine Relativität
  - LPM1 Lokale Teilchen-Gitter Simulation (elektro-magnetisch)

11.5 RAPS Real Applications on Parallel Systems

Das RAPS (Real Applications on Parallel Systems) Konsortium besteht aus Benutzern und Entwickelrn von großen Anwendungsprogrammen auf Supercomputern. Es wird unterstützt von einem beratenden Konsortium von Herstellern (Convex, Cray, Fujitsu, IBM, Intel and Meiko).

Zum RAPS-Benchmark gehören folgende Programme:

- PAM-CRASH
  Ein FEM-Code, der hauptsächlich in der KFZ-Crashanalyse benutzt wird
- IFS/ARPEGE
  ein globaler Atmosphären-Simulationscode aus der Meteorologie und Klimatologie
- FIRE
  Ein strömungsmechanischer Code für Simulationen im Kraftfahrzeugbereich
- GEANT
  wird am CERN benutzt, um die Interaktion von hochenergetischen Teilchenschauern mit den Detektoren zu simulieren

11.6 PARKBENCH PARallel Kernels and BENCHmarks

Auf der Supercomputing'92 in Minneapolis haben sich ca. 50 interessierte Leute zusammengetan aus Universitäten, Forschungseinrichtungen und Industrie und gründeten das PARKBENCH (PARallel Kernels and BENCHmarks) Komitee. In ihm sind Anwender und Hersteller von beiden Seiten des Atlantiks vertreten [58].

Der anfängliche Fokus liegt auf Benchmarks für skalierbare Distributed-Memory Message-Passing Architekturen. So werden zunächst die Benchmarks in FORTRAN77 mit PVM aus Portabilitätsgründen gemacht, und wenn das MPI definiert und verfügbar ist, wird dieses herangezogen [58, 59].

Die Benchmarks bestehen aus mehreren Teilbenchmarks[58]:

- Low-Level Benchmarks

Benchmark	Messungen	Herkunft
Single Processor		
TICK1	Timer resolution	Genesis
TICK2	Timer value	Genesis
RINF1	Basic Arith. ops.	Genesis, MOD1AC EuroBen
POLY1	Cache-bottleneck	MOD1G EuroBen
POLY2	Memory-bottleneck	Hockney
Multi Processor		
COMMS1	Basic Message perf.	Genesis
COMMS2	Message exch. perf.	Genesis
COMMS3	Saturation Bandwidth	PARKBENCH
POLY3	Comms. Bottleneck	PARKBENCH
SYNCH1	Barrier time and rate	Genesis

- Kernel Benchmarks
  - Matrix benchmarks
    - Dense matrix multiply
    - Transpose
    - Dense LU factorisation with partial pivoting
    - QR Decomposition
    - Matrix tridiagonalisation (eigenvalues for symmetric matrices)
  - Fast Fourier Tranforms
    - 1-D FFT
    - 3-D FFT (3-D FFT PDE aus NAS Parallel Benchmark)
  - PDE Kernels
    - Successive Over-Relaxation (SOR) (PDE1 aus Genesis)
    - Multigrid Kernel (MG aus NAS Parallel Benchmark)
  - Others
    - Embarrassingly Parallel (aus NAS Parallel Benchmark)
    - Large Integer Sort (aus NAS Parallel Benchmark)
    - Input/Output
- Compact Applications
  Zunächst zehn Applikationen aus den Bereichen:
  - Climate and meteorological modeling
  - Computational fluid dynamics (CFD)
  - Finance, e.g., portfolio optimization
  - Molecular dynamics

- Plasma physics
- Quantum chemistry
- Quantum chromodynamics (QCD)
- Reservoir modeling

- HPF Compiler Benchmarks
  - FORALL staement - kernel FL
  - Explicit template - kernel TL
  - Communication detection in array assignments - kernels AA, SH and IR
  - INDEPENDENT assertion - kernel EP
  - Non-elemental intrinsic functions - kernel RD
  - Passing distributed arrays as subprograms' arguments - kernels AS, IT, IM and EI

12 Informationsquellen

Eine der aktuellsten Informationsquellen sind die USENET–NEWS. Dort werden in den Newsgruppen `comp.parallel` und `comp.parallel.pvm` die neuesten Nachrichten ausgetauscht.

Für Cluster von Rechnern hat sich eine Mailing–Liste gebildet, bei der man sich durch einen Brief an `Info-Clusters-Request@LaRC.NASA.GOV` anmelden kann.

Für den Bereich Computational Chemistry wird am Ohio Supercomputer Center eine E–Mail–Verteilerliste gepflegt. Mit einem Brief an `MAILSERV@osc.edu` mit der Nachricht `help chemistry` kann man erfahren, wie man sich in die Liste eintragen kann und welche Dienstleistungen noch geboten werden (anonymer ftp, Archiv etc.).

An der Universität von Kent in Canterbury wird ein anonymer FTP betrieben, auf dem viele im Internet angekündigte Informationen gespeichert werden. Der Rechner hat den Namen `unix.hensa.ac.uk`. Im Directory `/parallel` sind die Informationen nach unterschiedlichen Kriterien abgelegt.

13 Zukunftsaussichten

Die Entwicklung auf dem Gebiet geht mit einer erheblichen Geschwindigkeit weiter. Für den Anwender ist es sehr wichtig, daß immer mehr mehr fertige Anwendungspakete auf dem Markt erscheinen. Aber auch der Entwickler von Software wird durch das Erscheinen von parallelisierenden Compilern bzw. Programmiersprachen (wie zum Beispiel HPF) unterstützt. Die Debugmöglichkeiten paralleler Programme müssen noch verbessert werden.

Die personalintensiven Administratorarbeiten werden bei weiterer Verbreitung von Clustern eine Reduzierung erfahren durch Ausreifung der Systeme. Im Bereich der Massiv Parallelen Systeme und der Shared Memory Systeme sind einige spannende Entwicklungen von den verschiedenen Herstellern angekündigt.

Erfreulich ist, daß auch schon Standardisierungsbemühungen wie POSIX 1003.15 (Queueing Systems) und das Message Passing Interface (MPI) in der Entwicklung sind.

Literatur

[1] Abstracts. Cluster Computing Workshop. Florida State University. Dezember 1992. `[ftp.scri.fsu.edu:/pub/cluster-workshop.92/abstracts]` [2] [3]

[2] Die Angaben in eckigen Klammern beziehen sich auf die Quellen der Artikel, wenn sie in elektronischer Form vorliegen. Sie sind dann über anonymen Ftp zu bekommen. Es ist immer Rechnername und Dateiname angegeben

[3] Einige der genannten Artikel sind auch über anonymen Ftp von den Rechnern `ftp.CNB.CompuNet.DE` und `ftp.zrz.TU-Berlin.DE` zu erhalten.

[2] Abstracts. Cluster Computing Workshop. Florida State University. Dezember 1993.
[ftp.scri.fsu.edu:/pub/cluster-workshop.93/abstracts]

[3] Ahlrichs, R.:
Quantenchemie — ein Phönix aus der Asche. IBM Nachrichten 42 (1992) Special II, S. 42–44.

[4] Ahuja, S., Carriero, N., Gelernter:
Linda and Friends. IEEE Computer 19(8) 1986, 26–34.

[5] Barret, R., Berry, M., Chan, T., Demmel, J., Donald, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., van der Vorst, H.:
Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. Knoxville, October 1993.
[netlib2.cs.utk.edu:linalg/templates.ps.Z]

[6] Basermann, A.:
Conjugate Gradients Parallelized on the Hypercube. KFA-ZAM-IB-9309. September 1993.
[ftp.zam.kfa-juelich.de:/pub/zamdoc/ib-9309.ps]

[7] Berrendorf, R., Detert, U., Docter, J., Ehrhart, U., Gerndt, M., Gutheil, I., Knecht, R.:
Massively Parallel Computing in a Production Environment iPSC/860 Installation at KFA Jülich. KFA-ZAM-IB-9304. Februar 1993.
[ftp.zam.kfa-juelich.de:/pub/zamdoc/ib-9304.ps]

[8] Berry, M., Do, T., O'Brien, G., Krishna, V., Varadhan, S.:
SVDPACKC (Version 1.0) — User's Guide. April 1993.
[netlib2.cs.utk.edu:tennessee/ut-cs-93-194.ps]

[9] Bönniger, T., Esser, R., Krekel, D.:
CM-5, KSR1, Paragon XP/S: a comparative description of massively parallel computers on the basis of a catalog of classifying characteristics. KFA-ZAM-IB-9320. Oktober 1993.
[ftp.zam.kfa-juelich.de:/pub/zamdoc/ib-9320.ps]

[10] Brode, S., Horn, H., Ehrig, M., Moldrup, D., Rice, J., Ahlrichs, R.:
A Parallel Direct SCF and Gradient Program for Workstation Clusters.

[11] Brode, S.:
Entwicklung eines parallelen Quantenchemie-Programms für Workstation-Cluster. Workshop Supercomputer'93. Mannheim.

[12] Busby, L.:
A Review of the Lasnet Batchjob System. Cluster Computing Workshop. Florida State University. Dezember 1992. [ftp.scri.fsu.edu:/pub/cluster-workshop.92/leebusby.tar.Z]

[13] Butler, R., Lusk, E.:
User's Guide to the p4 Programming System. Argonne National Laboratory. October 1992.

[14] Butler, R., Lusk, E.:
Monitors, Messages, and Clusters: the p4 Parallel Programming System. U of North Florida, Argonne National Laboratory. 1993.
[info.mcs.anl.gov:/pub/p4/p4-paper.ps.Z]

[15] Cap. C., Strumpen, V.:
THE PARFORM — A High Performance Platform for Parallel Computing in a Distributed Workstation Environment. Institut für Informatik, Universität Zürich. 23.6.1992.

[16] Calkin, R., Hempel, R., Hoppe, H.-C., Wypior, P.:
Portable Programming with PARMACS Message Passing Library. GMD, October 13, 1993. In Press.

[32] Esser, R., Knecht, R.:
Intel Paragon XP/S — Architecture and Software Environment. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[33] Esser, R., Knecht, R.:
Intel Paragon XP/S — Architecture and Software Environment. KFA-ZAM-IB-9305. April 1993.
[ftp.zam.kfa-juelich.de:pub/zamdoc/ib-9305.ps]

[34] Emmen, A.:
An overview of HPC-systems. SHARE Europe. October 27, 1993.

[35] Filippone, S., Sales, M.L.:
Parallel Linear System Solvers on IBM RISC/System 6000 Clusters. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/rs6knum.ps.Z]

[36] Applied Parallel Research:
Forge 90, Version 8.0, Baseline System. Preliminary Quickstart Tutorial. September 1992.

[37] Applied Parallel Research:
Forge 90, Distributed Memory Parallelizer. Preliminary Release Notes. Version 8.0. September 1992.

[38] Foster, I., Olson, R., Tuecke, S.:
Productive Parallel Programming: The PCN Approach. Scientific Progr., 1(1): 51-66, Fall 1992.
[info.mcs.anl.gov:pub/pcn/sci_prog.ps.Z]

[39] Foster, I., Michalakes, J.:
Massively Parallel Implementation of the Penn State/NCAR Mesoscale Model.
[info.mcs.anl.gov:pub/tech_reports/P324.ps.Z]

[40] Frank, S., Burkhardt, H., Rothnie, J.:
The KSR1: High Performance and Ease of Programming, No Longer an Oxymoron. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[41] Freedman Sharp and Associates Inc.:
Load Balancer v3.3.5. Automatic job queueing and load distribution over heterogeneous UNIX networks. 1993.

[42] Gard, C.H.:
University of Georgia's Experiences with Clustering RS/6000s and ES/9000 720. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/gard.talk.asc]

[43] Geiger, A.:
PARIS – Das Parallel RISC Projekt der IBM, DLR und Universität Stuttgart. In Supercomputer'93, Mannheim.

[44] Geiger, A.:
Workstation–Cluster für die Parallelverarbeitung. In Supercomputer'93, Mannheim. Workshop.

[45] Geist, A., Sunderam, V.S.:
Network Based Concurrent Computing on the PVM System.

[46] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., Sunderam, V.:
PVM 3 User's Guide and Reference Manual. ORNL/TM-12187. May, 1993.
[netlib2.cs.utk.edu:pvm3/ug.ps]

[47] Gomperts, R., Caltabiano, R., SGI:
A Simple Approach to Network Parallelism — ParNet. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/ParNet.SGI.ps.Z]

[48] Green, T.P., Snyder, J.:
DQS, A Distributed Queueing System. 15.3.1992.

[49] Green, T.P., Hudgens, J.:
DQS, A Distributed Queueing System. 4.12.1992. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/dqs-hudgens.tar.Z]

[50] Gropp, W.:
Early Experiences with the IBM SP1 and the High-Performance Switch. ANL-93/41. November 1993.
[info.mcs.anl.gov:pub/pdetools/TR93-41.ps.Z]

[51] Gropp, W.:
Early Experiences with the IBM SP-1. ANL/MCS-TM-177. May 1993.
[info.mcs.anl.gov:pub/pdetools/TM177.ps.Z]

[52] Gutheil, I., Krotz-Vogel, W.:
Performance of a Parallel Matrix Multiplication Routine on an Intel iPSC/860. KFA-ZAM-IB-9308. Mai 1993.
[ftp.zam.kfa-juelich.de:/pub/zamdoc/ib-9308.ps]

[53] Harrison, R.J.:
README des tcgmsg-Paketes. Argonne National Laboratory. Dezember 1991.

[54] Hempel, R.:
The ABL/GMD Macros (PARMACS) in FORTRAN for Portable Parallel Programming using the Message Passing Programming Model. User's Guide Reference Model. GMD, November 27, 1991.
[gmdzi.gmd.de:/gmd/numsoft/PARMACS/PM-5.1.ps.Z]

[55] Hempel, R., Hoppe, H.-C., Supalow, A.:
PARMACS 6.0 Library Interface Specification. GMD, December 17, 1992.
[gmdzi.gmd.de:/gmd/numsoft/PARMACS/PM-6.0.ps.Z]

[56] Henry, G., Hoisie, A.:
Blockfactorizations on a Cluster of RS/6000s. January 23, 1993.
[info.tc.cornell.edu:tr088.tex]

[57] Hertweck, F.:
A Comparison of Some Current Parallel Computer Architectures. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[58] Hockney, R., Berry, M.:
Public International Benchmarks for Parallel Computers. PARKBENCH Committee: Report-1. University of Tennessee. November 17, 1993.
[netlib2.cs.utk.edu:/tennessee/ut-cs-93-213.ps]

[59] Hockney, R.:
Supercomputer Benchmark and Results. In: SUP'EUR 93. Vienna, Austria. September 23-29, 1993.

[60] High Performance Fortran Forum:
High Performance Fortran, Language Specification. January 25, 1993. Version 1.0 DRAFT.

[61] IBM–Announcement 293-041 vom 2.2.93:
IBM LoadLeveler Version 1.1.0

[62] IBM–Announcement 293-040 vom 2.2.93:
IBM AIX Parallel Environment.

[63] IBM:
PVM/6000 version 2 - Parallel Virtual Machine for RISC/6000 workstations. October 1993.

[64] IBM Press Release PR931117:
IBM High Performance Parallel Computer Outperforms Competitors on Parallel Computing Benchmarks, Demonstrating Industry Leadership. November 17, 1993.

[65] Inmos; May, M.D., Thompson, P.W., Welch, P.H.:
Networks, Routers and Transputers. Inmos, February 1993.
[unix.hensa.ac.uk:/parallel/books/ios/nrat/*]

[66] Jones, M., Plassmann, P.:
Solution of Large, Sparse Systems of Linear Equations in Massively Parallel Apllications. Argonne National Laboratory, 1992.
[info.mcs.anl.gov:pub/techreports/P13.dvi.Z]

[67] Kaplan, J., Nelson, M.:
NASA Technical Memorandum 109025. A Comparison of Queueing, Cluster and Distributed Computing Systems. NASA Langley Research Center, October 1993.
[techreports.larc.nasa.gov:pub/techreports/larc/93/tm109025.ps.Z]

[68] Kapteyn, C., Xue, L., Thiele, F.:
Parallelisierung des CGS-Verfahrens auf dem Parsytec MultiCluster-1. Hermann-Föttinger-Institut für Thermo- und Fluiddynamik, TU Berlin. 29. Juni 1993.

[69] Kingsbury, B.A.:
The Network Queueing System. Preliminary Draft. Sterling Software. November 1990.

[70] Kuzela, J.M.:
IBM Scalable POWERparallel System SP1. Performance Measurements. September 21, 1993. Unveröffentlicht.

[71] Lacasse, M.-D.:
DNQS — A Dynamical Network Queueing System. User Manual. Preliminary Copy. Version 1.02, April 1992.

[72] LaRose, B.:
The Development and Implementation of a Performance Database Server. Knoxville, August 1993.
[netlib2.cs.utk.edu:/tennessee/ut-cs-93-195.ps]

[73] Lawson, C., Hanson, R., Kincaid, D., Krogh, F.:
Basic Linear Algebra Subprograms for Fortran Usage. ACM Transactions on Mathematical Software, Vol. 5, No. 3, September 1979, Pages 308-323.

[74] Leon, J., Fisher, A., Steenkiste, P.:
Fail-safe PVM. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/fsafe.pvm.jl.ps.Z]

[75] Lewis, J., van de Geijn, R.:
Distributed Memory Matrix–Vector Multiplication and Conjugate Gradient Algorithms.
[cs.utexas.edu:pub/SC93.ps]

[76] Lischka, H., Karpfen, A.:
Computersimualtion der Struktur und Dynamik von Molekl̈en, Polymeren und Festkörpern.
In: Marksteiner, P., Schnable, W.: Zwei Jahre Supercomputing. Bericht über das EASI-Kooperationsprojekt 1989/90 an der Universität Wien. Wien. 1992.

[77] Lusk, R., Butler, R.:
Portable Parallel Programming with P4. Cluster Computing Workshop. Florida State University. Dezember 1992. [ftp.scri.fsu.edu:/pub/cluster-workshop.92/p4.lusk.ps.Z]

[78] Manchek, R.:
An Introduction to PVM (Parallel Virtual Machine). Knoxville. August, 1993.
[netlib2.cs.utk.edu:pvm3/pvm_slides.tar.Z]

[79] Marksteiner, P.:
VQS – The Vienna Queueing System. In: SUP'EUR 93. Vienna, Austria. September 23-29, 1993.

[80] Marksteiner, P.:
Using the RS/6000 Cluster at Vienna University Computer Center. 15 July 1993.
[ftp.univie.ac.at:/at.local/cluster/vdoc.ps.Z]

[81] Martin, J.:
SP1 - Performance Benchmarks. SHARE Europe. October 27, 1993.

[82] Martin, J.:
Status and Directions. Scalable POWERparallel Systems. SHARE Europe. October 27, 1993.

[83] Mattson, T.G., Bjornson, R., Kaminsky, D.:
The C-Linda Language for Networks of Workstations. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/linda.ps.Z]

[84] Mattson, T.:
Programming Parallel and Distributed Computers. Cluster Computing Workshop. Florida State University. Dezember 1993.
[ftp.scri.fsu.edu:/pub/cluster-workshop.93/Tutorial/par_prog.ps.Z]

[85] Mattson, T., Judsen, R.:
pDGEOM: a Parallel Dinstance Geometry Program. Yale University Research Report, May 1993. Unveröffentlicht.

[86] Message Passing Interface Forum:
DRAFT Document for a Standard Message-Passing Interface. May 28, 1993.
[netlib2.CS.UTK.EDU:mpi/draft528.ps]

[87] Message Passing Interface Forum:
MPI: A Message Passing Interface.
[netlib2.CS.UTK.EDU:mpi/sc93.ps]

[88] Minser, K.:
Parallel Map Analysis on the CM-5 for Landscape Ecology Models. Koxville. August 1993.
[netlib2.cs.utk.edu:/tennessee/ut-cs-197.ps]

[89] Miura, K.:
VPP500 Supercomputing System. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[90] Nelson, S.
Designing MPP Systems to Optimize Time-To-Solution Performance. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[91] Ni, L.M., McKinley, P.:
A Survey of Wormhole Routing Techniques in Direct Networks. IEEE COMPUTER. February 1993, p.62-76.

[92] Ni, L.M., McKinley, P.:
A Survey of Routing Techniques in Wormhole Networks. Technical report MSU-CPS-ACS-46, October 17, 1991.
[ftp.cps.msu.edu:pub/crg/PAPERS/msu-cps-acs-46.ps]

[93] Oed, W.:
The Cray Research Massively Parallel Processor System CRAY T3D. Cray Research Inc. November 15, 1993. [ftp.cray.com:pub/product-info/mpp/T3D_overview.ps]

[94] Pallas GmbH:
PARMACS versus PVM - a Position Paper. Pallas GmbH. September 21, 1993.

[95] ParaSoft:
Programming with Express by Example. Cluster Computing Workshop. Florida State University. Dezember 1992. [ftp.scri.fsu.edu:/pub/cluster-workshop.92/express.ps]

[96] ParaSoft:
Writing Programs for Parallel and Distributed Computers. Level I (Theory). Tutorial. ParaSoft Corporation, 1993.
[ftp.parasoft.com:/express/classes/theory.ps.Z]

[97] Patterson, L., Turner, R.S., Hyatt, R.M., Reilly, K.D.:
Construction of a Fault–Tolerant Distributed Tuple–Space. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/tuplespace.tex]

[98] Perić, M., Schäfer, M., Schreck, E.:
Numerical Simulation of Complex Fluid Flows on MIMD Computers.

[99] Pettersson, L., Faxen, T.:
Massively Parallel Direct SCF Calculations on Large Metal Clusters: Ni_5 – Ni_{481}.

[100] Pleier, C.:
Massiv–Paralleler Supercomputer T3D von Cray. iX 10/1993, S.144-151.

[101] Prechelt, L.:
Measurements of MasPar MP-1216A Communication Operations. Technical Report 1/93. Universität Karlsruhe. January 6, 1993.
[sanfrancisco.ira.uka.de:/pub/maspar/maspar_measure_report.ps.Z]

[102] Prechelt, L.:
Comparison of MasPAr MP-1 and MP-2 Communication Operations. Technical Report 16/93. Universität Karlsruhe. April 23, 1993.
[sanfrancisco.ira.uka.de:/pub/maspar/maspar_measure_report2.ps.Z]

[103] Revor, L.S.:
DQS Users Guide. Argonne National Laboratory, 15.9.1992.

[104] Rosti, E., Smirni, E., Wagner, T., Apon, A., Dowdy, L.:
The KSR1: Experimentation and Modeling with Poststore. Vanderbuildt University, Nashville. 1992.
[hopscotch.ksr.com:pub/mikep/UT-prefetch.ps.Z]

[105] Roweth, D.:
Computing Surface 2. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[106] Sanz, J., Pischel, K., Hubler, D.:
The Engine Design Engine. Cluster Computing Workshop. Florida Sate University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/engine.design.engine.ps]

[107] Sept, D.:
The Design, Implementation and Performance of a Queue Manager for PVM. Knoxville, August 1993.
[netlib2.cs.utk.edu:tennessee/ut-cs-93-196.ps]

[108] Shephard, R., Shavitt, I., Pitzer, Comeau, D., Pepper, M.,
Lischka, H., Szalay, P., Ahlrichs, R., Brown, F., Zhao, J.-G.: A Progress Report on the Status of the COLUMBUS MRCI Program System. International Journal of Quantum Chemistry: Quantum Chemistry Symposium 22, 149–165 (1988).

[109] Sterling Software:
Sterling NQS and NQS/Exec. User's Guide. 1993.

[110] Sterling Software:
Sterling NQS and NQS/Exec. System Administrator Guide. Revision 2.3, 1993.

[111] Sunderam, V.S.:
PVM: A Framework for Parallel Distributed Computing.

[112] Sunderam, V.S.:
Concurrent Computing with PVM. Cluster Computing Workshop. Florida State University. Dezember 1992. [ftp.scri.fsu.edu:/pub/cluster-workshop.92/pvm.tss.tar.Z]

[113] Szelenyi, Ferenc:
Workstation Cluster und Massiv-Parallele Systeme. In: IBM Hochschulkongreß, Dresden, 30.9.92 – 2.10.92. Dokumentation.

[114] Szelenyi, Ferenc:
Scalable POWERparallel Systems IBM 9076 SP1. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[115] Szelenyi, Ferenc:
Vom Workstation Cluster zu massiv-parallelen Systemen. In: Offene Systeme. GUUG-Jahrestagung 1993. Wiesbaden.

[116] Turcotte, L.:
A Survey of Software Environments for Exploiting Networked Computing Resources. June 11, 1993. [bulldog.wes.army.mil:pub/report.ps.Z]

[117] Vereecken, H., Lindenmayer, G., Kuhr, A., Welte, D.H., Basermann, A.:
Numerical Modelling of Field Scale Transport in Heterogeneous Variably Saturated Porous Medi. KFA-ZAM-IB-9301. Januar 1993.
[ftp.zam.kfa-juelich.de:pub/zamdoc/ib-9301.ps]

[118] Wacker, H.-M.:
Evolution und Perspektiven der Supercomputer. Workshop. Supercomputer'93. Mannheim.

[119] Wallach, S.:
SPP-1. In: Meurer, H.-W.: Supercomputer'93. Mannheim.

[120] Wanie, K.M., Schmatz, M.A.:
Verification and Application of the NSFLEX Method for Hypersonic Conditions. Messerschmidt–Bölkow–Blohm.

[121] Wehinger, W.:
Workstation Cluster in der Nachfolge des klassischen Universalrechners. Workshop. Supercomputer'93. Mannheim.

[122] White, S., Ålund, A., Sunderam, V.:
The NAS Parallel Benchmarks on Virtual Parallel Machines. Atlanta, Fall 1993.
[mathcs.emory.edu:pub/vsspvm/naspvm.ps.Z]

[123] Windheiser, D., Boyd, E., Hao, E., Abraham, S., Davidson, E.:
KSR1 Multiprocessor: Analysis of Latency Hiding Techniques in a Sparse Solver.
[hopscotch.ksr.com:pub/mikep/UofM.latency.ps.Z]

[124] Wolbers, S.:
Parallel and Clustered Computing at Fermilab. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/fermilab.cps.ps]

[125] Zhou, S., Wang, J., Zheng, X., Delisle, P.:
UTOPIA: A Load Sharing Facility for Large, Heterogenous Distributed Computer Systems. Technical Report CSRI–257. April 1992.

[126] Zhou, S.:
LSF: Load Sharing in Large-Scale Heterogenous Distributed Systems. Cluster Computing Workshop. Florida State University. Dezember 1992.
[ftp.scri.fsu.edu:/pub/cluster-workshop.92/lsf.talk.ps.Z]

[127] Zollweg, J.:
SP1 vs. Cluster. SHARE Europe. October 27, 1993.