

# Cluster Computing

Proseminar „Paralleles und verteiltes Rechnen“

Bernd Wurst

Sommersemester 2003

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>3</b>
1.1	Härter arbeiten . . . . .	3
1.2	Geschickter arbeiten . . . . .	4
1.3	sich Hilfe von anderen holen . . . . .	4
<b>2</b>	<b>SMP oder Cluster?</b>	<b>5</b>
2.1	Vorteile von Cluster-Systemen . . . . .	7
2.2	Nachteile von Clustern . . . . .	7
<b>3</b>	<b>Kommunikationstechniken</b>	<b>8</b>
3.1	Exposed-Kommunikation . . . . .	9
3.2	Enclosed-Kommunikation . . . . .	10
<b>4</b>	<b>Aufstellungsprinzipien</b>	<b>11</b>
4.1	Glass-House . . . . .	11
4.2	Campus-Wide . . . . .	12
<b>5</b>	<b>Software-Strategien</b>	<b>13</b>
5.1	Condor . . . . .	13
5.2	Piranha . . . . .	13
5.3	Sprite/Locus . . . . .	14
<b>6</b>	<b>Programmiermodelle</b>	<b>14</b>
6.1	Small-N . . . . .	15
6.2	Large-N . . . . .	15
<b>7</b>	<b>Single-System-Image</b>	<b>15</b>
<b>8</b>	<b>Zusammenfassung und Ausblick</b>	<b>16</b>

# 1 Einleitung

Um eine Leistungssteigerung zu erlangen gibt es drei grundsätzlich unterschiedliche verschiedene Vorgehensweisen:

1. härter arbeiten.
2. geschickter arbeiten.
3. sich Hilfe von anderen holen.

Die erste dieser Methoden ist wohl die naheliegendste, denn dies kann jeder selber machen. Zumindest bis zu einem gewissen Grad, denn irgendwann ist eine natürlich gegebene Grenze erreicht. Diese Grenze ist immer abhängig von der Leistung, die der Einzelne im Stande ist zu verrichten. Dieser zusätzliche Aufwand kann aber nicht konserviert werden und wenn das nächste Mal eine gleiche Arbeit getan werden muss, steht man wieder vor dem selben Problem und muss, sofern man sich wieder für diese Form der Leistungssteigerung entscheidet, noch einmal mehr Einzel-Leistung bringen.

Die zweite Taktik, intelligenter zu arbeiten, kann auch jeder versuchen. Nur wird es selten jemanden geben, der die optimale Methode findet um eine gegebene Arbeit zu erledigen. Es ist für einen einzelnen auch damit nur bis zu einer gewissen Grenze möglich, den Arbeitsablauf zu optimieren. Der Vorteil dieser Vorgehensweise ist, dass man selbst und auch andere von der Optimierung profitieren können, sobald einmal eine ähnliche Arbeit gemacht werden muss.

Erst bei der dritten Vorgehensweise, der Hilfe von anderen, gibt es nahezu unbegrenzt viele Möglichkeiten eine Steigerung der Leistung zu erzielen. Hier hängt es stärker von der Aufgabe ab, wie viel man von zusätzlichen Helfern profitieren kann. Diese zusätzlichen Helfer können jedes Mal neu gesucht werden oder sich einfach zu einem starken Team zusammenschließen, das zukünftige Aufgaben sehr flott erledigen kann.

Wenn man diese verschiedenen Ansätze auf moderne Computersysteme überträgt, finden sich dort sehr genaue Entsprechungen für jede dieser Methoden.

## 1.1 Härter arbeiten

Einen Computer kann man sehr einfach härter arbeiten lassen, in dem man die Taktfrequenz des Systems erhöht. Dies funktioniert meist so, dass man

in ein bestehendes System eine schnellere CPU einbaut oder eine bestehende CPU auf höheren Takt einstellt. Entgegen der oben beschriebenen Problematik bei natürlichen Arbeitskräften, wird hier natürlich die Leistung dauerhaft gesteigert, der Aufwand ist also nicht unbedingt nur für eine Aufgabe sondern bleibt erhalten. Allerdings wird auch hier irgendwann die Grenze eines Systems erreicht, sodass eine neue CPU auf einem alten Mainboard ihre Leistung nicht mehr voll ausschöpfen kann oder vor allem bei modernen Rechnern erst gar nicht mehr dort eingebaut werden kann.

Im Zuge einer solchen Aufrüstung ist also auch oft der Austausch des ganzen Computers nötig um von den aktuellsten Neuerungen der Prozessorhersteller zu profitieren. Und auch wenn man diesen Schritt durchführt, ist die neue Grenze der Belastbarkeit des Systems doch meist viel zu schnell erreicht.

## 1.2 Geschickter arbeiten

Diese Form der Leistungssteigerung betrifft in der Datenverarbeitung in der Regel die Software. Wenn dort eine oft benutzte Routine mit einem effizienteren Algorithmus bearbeitet werden kann, verbessert das die Geschwindigkeit meist sehr deutlich. Dieser effizientere Maschinencode kann hier in unterschiedlichen Arbeitsschritten gewonnen werden, nämlich entweder durch eine Implementierung in effizienteren Algorithmen oder durch eine bessere Anpassung des vom Compiler generierten Codes an den Prozessor, auf dem das Programm seine Arbeit verrichten soll.

Jedoch gilt auch hier die Erkenntnis, dass diese Form der Leistungssteigerung zwar kein definierbares Ende hat, denn kein Algorithmus ist optimal. Aber die Arbeit den ein Programmierer einsetzen muss um ein Programm zu optimieren nimmt untolerierbare Ausmaße an und deshalb kann man auch hier von einem frühen Ende der Beschleunigung ausgehen.

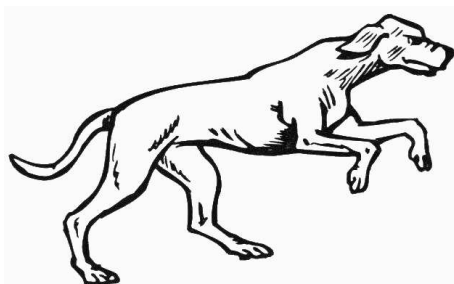
## 1.3 sich Hilfe von anderen holen

Diese Methode zur schnelleren Bearbeitung von Aufgaben wird Thema dieses restlichen Textes sein. Denn auch dabei gibt es in der EDV unterschiedliche Ansätze, diese Methode umzusetzen. Ähnlich wie oben für den Fall von natürlichen Arbeitskräften, können solche Rechengemeinschaften mit verschiedenen Bindungsstärken realisiert werden. Dazu kommen Unterschiede in der Fehlerbehandlung und zum Beispiel der Datenspeicherung.

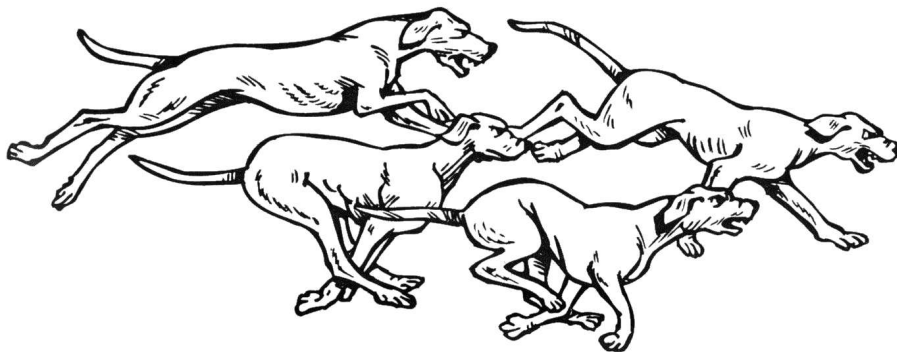
## 2 SMP oder Cluster?

Diese beiden Begriffe werden im Text noch häufiger vorkommen, denn so kann man eine grobe Einteilung vornehmen zwischen Rechnern, die sehr eng zusammenarbeiten, bei denen nur die Hauptkomponente in Form des Prozessors mehrfach vorkommt und anderen Systemen, die an sich autonome Rechner miteinander verbinden.

Der Autor des Buches „In Search Of Clusters“ [Pfi95] beschreibt eine sehr anschauliche Analogie. Betrachten wir nun einmal einen Computer sowie dessen Rechenleistung als einen Hund und dessen Jagd-Fähigkeiten.



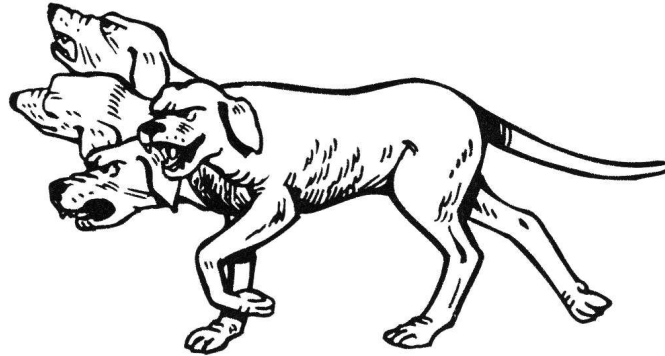
Hier sehen wir einen einzelnen, kräftig gebauten Jagdhund. Dieser Hund hat nahezu ideale Voraussetzungen für die Jagd. Allerdings hat er einfach ein natürlich gesetztes Limit seiner Möglichkeiten. Um mehr Leistung zu erreichen, schließen sich Hunde in der freien Wildbahn auch gerne zu Rudeln zusammen. Dies würde dann ungefähr so aussehen.



Dieses Rudel besteht aus einigen Hunden des beschriebenen Körperbaus, alle sind für sich genommen sehr agil und geschickt, nur leider gibt es ab und zu Unstimmigkeiten in der Rangordnung, was sich vor allem darin bemerkbar macht, das es für das Jagdergebnis unnötige Kämpfe innerhalb des Rudels

gibt, bei denen Leistung verschwendet wird.

Um diesen Nachteil zu kompensieren, erfindet der Autor eine dritte Jagdstrategie, den sogenannten *Savage Multiheaded Pooch*, kurz auch SMP genannt.



Dieser speziell für die Steigerung der Leistung gezüchtete Hund lässt sich dank nur eines einzelnen Körpers sehr einfach zur Kooperation anregen. Gewiss gibt es auch hier gewisse Nachteile, denn wenn einer der Köpfe plötzlich abgelenkt ist, beeinträchtigt das die Funktionalität des ganzen „Rudels“, denn der Körper könnte damit verschiedene Befehle bekommen. Problematisch wird es ebenfalls wenn ein Teil (außer dem Kopf) des SMP verletzt wird.

Es ist treffend, sich mit Hilfe dieser Analogie die Methoden zu veranschaulichen, wie man einen Computer zu höherer Leistung bringen kann. Denn die beschriebenen Probleme lassen sich auch recht gut auf Computersysteme übertragen. Dabei steht der Begriff „SMP“ für *Symetric Multiprocessors* und damit werden die letztgenannten Systeme bezeichnet, die nur den Kern symmetrisch mehrfach eingebaut haben. Die andere Bauform „Cluster“ bezeichnet die Verbindung grundsätzlich unabhängiger Systeme zu einem großen Komplex. Dabei muss man diese Unterteilung jedoch immer sehr vorsichtig ziehen, denn mittlerweile gibt es nahezu jede denkbare Vermischung dieser Strategien. So ist es zum Beispiel durchaus möglich, mehrere Rechner mit einem gemeinsamen Arbeitsspeicher zu versehen und so auch deren Zusammenarbeit enger zu gestalten. Jede dieser Bauformen hat ganz ähnliche Vor- und Nachteile wie die Entsprechung in der oben angeführten Analogie. Meist sind die Nachteile des einen die Vorteile des anderen Systems, daher bieten sich vielseitige Mischformen durchaus an.

Dieser Text soll die verschiedenen Ansätze von Clustern vorstellen und

über deren Vor- und Nachteile Aufschluss geben. Dabei gibt es jedoch bei der Einteilung in Cluster- und SMP-Systeme grundsätzliche Unterschiede, die hier zuerst einmal herausgestellt werden sollen. Bei den möglichen Vermischungen dieser Ansätze können die meisten Nachteile von Clustern jedoch beseitigt werden.

## 2.1 Vorteile von Cluster-Systemen

Die Cluster-Rechner sind im Gegensatz zu den Prozessoren eines SMP-Rechners weit möglichst separiert. Dadurch ergeben sich enorme Vorteile im Schutz vor Systemausfällen. Sämtliche Hardware ist mehrfach vorhanden und kann bei einem Defekt die Komponenten in anderen Rechnern nicht beeinträchtigen. Das mehrfache Vorhandensein jeder Systemkomponente eröffnet zudem auch noch einen anderen Vorteil, der sich je nach Anwendungsgebiet unterschiedlich stark ausnutzen lässt. Nämlich die bessere Verteilung der Last auf unabhängige Hardware. Die strikte Trennung zwischen den einzelnen Rechnern erfordert viel weniger Koordination innerhalb eines einzelnen Rechners. Die gemeinsame Nutzung des Hauptspeichers bei einem SMP-System bremst gelegentlich das ganze System aus.

Der Hauptvorteil eines Clusters gegenüber einem Multiprozessorsystem dürfte der Kostenfaktor sein. Während ein Mainboard für ein SMP-System immer eine Spezialanfertigung ist, benötigt man für die Bildung eines Clusters nur einzelne, normale PC-Rechner. Die Bildung eines Clusters ist von der Software gesteuert und braucht keine besonderen Voraussetzungen in der Hardware. Für die angedeuteten Mischformen gilt dies natürlich nur sehr eingeschränkt, da gerade dort oft ganz spezielle Komponenten zum Einsatz kommen.

Durch die externe Verbindung der Computer kann ein Cluster in der Regel auch sehr einfach und kostengünstig erweitert werden. Sofern die benutzte Software die unterstützt, kann so ein Cluster dynamisch je nach Bedarf vergrößert oder verkleinert werden. Dieser Ausbau kann dabei auch eventuell mit bereits vorhandenen Computern gemacht werden, sodass für eine kurzfristige Erweiterung eines Clusters mitunter gar keine Kosten anfallen.

## 2.2 Nachteile von Clustern

Wie fast alles im Leben hat auch das Cluster-Prinzip ein paar nicht zu verschweigende Nachteile, die wiederum als Vorteile eines SMP-Systems angesehen werden können.

Die Kommunikation unter den einzelnen Prozessoren eines Multiprozessorsystems verläuft äußerst effizient, da sie sich in der Regel Arbeitsspeicher und Festplatten teilen und so schnell Daten von einer zur anderen CPU übertragen werden können. Die schnellere Kommunikation über lokalen Arbeitsspeicher kann, passende Programmierung vorausgesetzt, einen erheblichen Vorteil in der Zusammenarbeit ausmachen.

Ein weiterer Nachteil des Cluster-Verfahrens ergibt sich auch bei der gezeigten Analogie zu dem Jagdhunden, nämlich die schlechte Kontrollierbarkeit. Es ist deutlich schwerer, mehrere eigenständige Individuen oder eben auch Computer zu einem gemeinsamen Ziel zu bringen. Bei dieser Koordinationsarbeit geht oft ein relevanter Anteil der Leistung verloren.

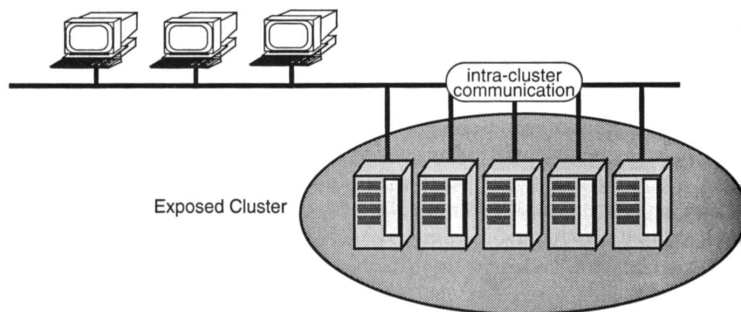
Um diese Verluste bei der Koordination möglichst gering zu halten, akzeptiert man immer einen Kompromiss aus möglichst großer Unabhängigkeit der einzelnen Clients und bestmöglicher Zusammenarbeit. Vor allem wenn man sehr viele Zugriffe auf die Hardware benötigt, ist es wichtig dass die Clients sich nicht gegenseitig ausbremsen können. Möchte man jedoch eine möglichst enge Zusammenarbeit der einzelnen Systeme haben, bei der eine intensive Kommunikation nötig ist, muss dieser Kompromiss eine andere Gewichtung haben.

### 3 Kommunikationstechniken

Im Folgenden wird dieser Text nur noch die oben definierten Cluster-Systeme beschreiben, sowie all die Mischformen, die sich mehr an den Eigenschaften eines Clusters orientieren. Das reine SMP-System sowie die Kommunikation innerhalb eines Mehrprozessor-Rechners wird jetzt nicht weiter beschrieben.

Die Vorgehensweise, wie die einzelnen Cluster-Rechner untereinander und nach außen kommunizieren, lässt sich in zwei grundsätzliche Strategien einordnen. Beim sogenannten „Exposed“-Prinzip sind die einzelnen Rechner über ein gewöhnliches Netzwerk untereinander und mit den Arbeitsrechnern verbunden. Beim „Enclosed“-Prinzip dagegen, verläuft die interne Kommunikation streng getrennt und der Cluster wird typischerweise von außen nur als ein einzelner Rechner wahrgenommen.

### 3.1 Exposed-Kommunikation



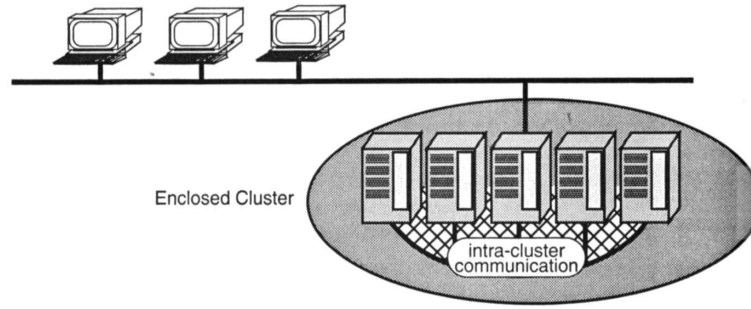
Bei dieser Vorgehensweise müssen die einzelnen Teilrechner eines Clusters ihre Synchronisation über ein gegebenes Netzwerk, typischerweise ein Ethernet, abwickeln. Dabei können im Vergleich zum später vorgestellten „Enclosed“-Prinzip deshalb Probleme auftreten, weil sich die Kommunikation an allgemeine Regeln und Standards des zugrunde liegenden Netzes halten muss. Um Konflikte zu vermeiden, muss also der Cluster seine Verbindung über bereits auf dem Netzwerk vorhandene Protokolle beschränken, was in der Regel einen nicht zu vernachlässigenden Overhead erzeugt und damit nur für derartige Cluster wirklich geeignet ist, die keine intensive Abstimmung brauchen.

Es wird weiterhin mit diesem Modell schwierig sein, Daten geheim zu halten, die der Cluster intern austauscht. Man muss bei der Planung eines derartigen Netzwerkes immer bedenken, dass nicht nur der gewünschte Empfänger ein Datenpaket auf dem Ethernet erhalten kann sondern auch jeder andere der physikalischen Zugriff auf das Netzwerk bekommt. Im Kombination mit immer populäreren drahtlosen Netzwerken erhöht sich diese Gefahr noch einmal sehr drastisch. Dieses Problem lässt sich nur dann effektiv lösen, wenn die Kommunikation innerhalb des Clusters verschlüsselt abläuft, was aber wiederum sehr viel verschwendete Rechenleistung bedeutet.

Aber natürlich hat auch dieses Prinzip einige unschlagbare Vorteile. Der größte davon ist naheliegenderweise der Preis. So muss für diese Form prinzipbedingt eben kein neues Netzwerk verlegt werden, das meist schon bestehende kann einfach mitbenutzt werden. Auch die Netzwerkhardware kann einfach weiter genutzt werden, wenn ein ehemaliger Arbeitsrechner (wie zuvor beschrieben) einfach zu einem Cluster-Rechner umgewandelt wird. Besonders bemerkbar macht sich dieser Vorteil wenn man ein dezentrales Cluster-Netzwerk anstrebt, wenn also die einzelnen Computer nicht in einem Raum oder gar einem Schrank versammelt sind sondern auf einer weiten Fläche

verteilt sind.

### 3.2 Enclosed-Kommunikation



Bei Verwendung dieses Modells wird ein zusätzliches Netzwerk eingerichtet, an dem nur andere Cluster-Computer angeschlossen werden. Daher fallen sehr viele Probleme des letzten Abschnittes weg, denn die Kommunikation muss sich nicht an gegebene Bedingungen und Standards halten sondern kann so erfolgen, wie es für den Ablauf der Berechnungen sinnvoll ist.

Typischerweise kann hierfür auch ein speziell für Cluster-Technik bestimmtes und damit schnelleres Netzwerk eingesetzt werden. Die Datenkommunikation kann damit durchaus Geschwindigkeiten annehmen, die an jene innerhalb eines Rechners herankommt. Bei jedem Vorgehen findet man häufig noch eine separate Verbindung über Ethernet, mit dem der Cluster kontrolliert wird. Diese zusätzliche Verkabelung ist zwar nicht unbedingt notwendig, bietet sich aber an damit die Datenkommunikation für die Berechnung nicht durch Kontrollmechanismen ausgebremst wird. Über dieses zusätzliche Ethernet lässt sich zum Beispiel der Cluster mit einem speziellen Steuerungsrechner verbinden, an dem die Rechenprozesse koordiniert werden. Dieser Rechner muss dann jedoch nicht in das interne Cluster-Netz angebunden sein, was auch wieder die Performance steigert.

Nicht zu vernachlässigen ist auch der Sicherheitsgewinn, denn in diesem eigenen Netz befindet sich kein anderer Computer und somit können die intern übertragenen Daten nicht von außen abgehört werden. Enclosed-Vernetzung eignet sich auch sehr gut für Cluster, die in Form des nachfolgend vorgestellten „Glass-House“ aufgebaut sind und somit werden Manipulationen und Abhörversuche deutlich erschwert, da auch physikalisch kein Zugriff von außen auf die Rechner möglich ist. Diese Vorteile erkaufte sich der Besitzer allerdings mit höheren Anschaffungskosten, da diese spezielle Netzwerk-

Hardware in der Regel teurer ist und oft auch mehrfache Vernetzung gebraucht wird. Was aber nicht bedeutet, dass zwingend ein teureres Spezialprodukt erworben werden muss, ein Netzwerk mit Enclosed-Kommunikation kann durchaus auch mit Ethernet realisiert werden.

Als Beispiel für eine solche Vernetzung bietet sich der hier in Tübingen aufgebaute „Kepler-Cluster“ [Kep00] an, der zur internen Datenkommunikation ein Myrinet<sup>®</sup> mit einer Geschwindigkeit von ungefähr 1 GBit/Sekunde und zur Anbindung nach außen ein 100 MBit-Ethernet benutzt. Die interne Kommunikation der 98 Einzelrechner kann somit völlig getrennt von der Steuerungs-Signalen erfolgen, die über das Ethernet von der Steuerungs-Station gesendet werden. Dieser Cluster ist in einem einzigen Rechnerraum aufgebaut, nach den „Glass-House“-Prinzip.

## 4 Aufstellungsprinzipien

Bei der Konzeption eines Clusters muss man sich nicht nur für eine Methode der internen Cluster-Kommunikation entscheiden, sondern auch wo und wie man die Computer aufstellen möchte. Man kann hier auch nur eine Einteilung in zwei grundsätzlich verschiedene Prinzipien festlegen, Abweichungen und Vermischungen können abhängig vom Einsatzzweck und der Größe des Clusters sinnvoller erscheinen. Die Einteilung in „Glass-House“ und „Campus-Wide“ soll die Unterschiede zwischen einem Cluster in einem speziell dafür reserviertem Raum oder Schrank und der anderen Form, dem Aufstellen der einzelnen Rechner verteilt über einen „Campus“, was hier aber auch genauso gut eine Firma sein kann.

### 4.1 Glass-House

Wenn ein Cluster in einem einzelnen Raum oder Schrank aufgebaut wird, resultieren daraus einige Vorteile, die der Betreiber für sich verbuchen kann. Die interne Kommunikation der einzelnen Cluster-Rechner wird dabei in der Regel nach dem Enclosed-Prinzip aufgebaut, da auch die räumliche Trennung der Computer von anderen Arbeitsrechnern dies begünstigt.

Durch diese Form des Clusters werden die Computer sehr leicht zu kontrollieren. Auch die Wartung bei Fehlern ist sehr einfach wenn sich alles an einem Ort befindet. Ein großer Vorteil zudem ist dass durch diese Aufstellungsform kein Fremder Zugriff auf die Computer bekommt und somit die Funktionsfähigkeit und die Datensicherheit des Clusters besser gewährleistet

werden kann. Die meisten Hochleistungsrechner sind nach diesem Prinzip aufgestellt.

## 4.2 Campus-Wide

Wenn man den Cluster über weite Teile einer Firma oder einer Universität verteilt aufstellt, spricht man von „Campus-Wide“. Der Hauptzweck dieser Form ist eine schnelle Veränderung des Clusters. So können zum Beispiel einfache Arbeitsrechner nach Feierabend zu einem virtuellen Cluster verbunden werden. Diese Methode spart viel Geld, da die kompletten Kosten des Clusters sich auf die Steuerungssoftware begrenzt. Zudem können auf diese Weise sehr einfach mehr oder weniger Computer benutzt werden, je nachdem wie viel Rechenleistung aktuell benötigt wird.

Die Leistung eines derartigen Systems ist dagegen sehr stark abhängig von der Aufgabe, die es zu bewältigen hat. Bei Berechnungen, die sehr intensive Kommunikation voraussetzen wird man hier erhebliche Einbußen in Kauf nehmen müssen. Zum einen durch längere Kabelwege und zum anderen durch Kollisionen auf einem Netzwerk, das nicht für ständige Dauerbelastung konzipiert ist. Für „Campus-Wide“ Aufstellung eignet sich zudem nur eine Verkabelung über einfaches Ethernet und auch dies nach dem „Exposed“-Prinzip. Da hierbei oft weite Wege zu überbrücken sind, wäre eine Vernetzung mit speziellen Systemen sehr aufwändig und teuer.

Sogar eine Verbindung von Rechnern mehrerer Firmen oder Universitäten über das Internet lässt sich nach dieser Methode aufbauen. Ein solcher Cluster eignet sich hauptsächlich für mehrere unabhängige Berechnungen, sodass ein einzelner Rechner unabhängig von den anderen seine Aufgabe erledigen kann und nur am Anfang und am Ende mit der Kontrollstation kommunizieren muss. Ein sehr weit ausgedehntes Beispiel hierfür ist das populäre „Seti-At-Home“ der Universität Berkeley [Set96], bei dem mehrere Millionen Heimcomputer über das Internet Datenpakete erhalten und diese eigenständig auf verdächtige Signale untersuchen. Erst wenn ein Paket fertig berechnet ist, wird wieder eine Verbindung zum Server bei der Universität Berkeley aufgebaut. Auf diese Weise wurden im Zeitraum zwischen dem Start 1997 und 2003, also in 6 Jahren nun schon 1549548,753 Jahre Computerleistung für das Projekt erbracht. Alleine für die selbe Leistung, die das Projekt innerhalb von 24 Stunden erreicht, wäre ein einzelner Prozessor 1898,024 Jahre beschäftigt. Diese Form der Clusterbildung kostet im Vergleich zu einem fest installierten Cluster mit ähnlicher Rechenleistung nahezu kein Geld. Allerdings ist auch der Verlust erheblich, so kommt es doch sehr oft vor, dass ein

Datenpaket zwar an einen Rechner ausgeliefert wird, dieses aber nicht wieder zurückkommt, weil der Benutzer Seti-At-Home ausgeschaltet hat oder ähnliches.

Dabei zeigt sich auch schon der kritischste Schwachpunkt der Vorgehensweise, denn der Administrator des Clusters kann lediglich den Server betreuen, die wirklichen Cluster-Rechner sind immer viel anfälliger als bei der „Glass-House“-Methode.

## 5 Software-Strategien

Die Koordination mehrerer Cluster-Computer stellt auch die Software vor schwere Aufgaben, denn diese muss den Prozess steuern. Unabhängig von der gewählten Kommunikation und des Aufstellungsprinzips kommt es immer wieder vor, dass ein einzelner Computer ausfällt oder aus einem anderen Grund die Berechnung auf einen anderen Computer übertragen werden muss. Die Reaktion der Steuerungssoftware auf dieses Problem lässt sich in drei Strategien einteilen.

### 5.1 Condor

Beim „Condor“-System wird darauf gesetzt, dass die Client-Software, die die Berechnung letztendlich durchführt, regelmäßig ihren aktuellen Stand zum Server überträgt. Dieser sichert diesen Stand für jeden seiner Clients und kann so, wenn eine Störung auftritt dieses Backup auf einem anderen Rechner weiter berechnen lassen. Somit geht zwar immer wieder ein Teil der Berechnung verloren, aber durch das Backup-Intervall kann dieser Verlust vom Benutzer in Grenzen gehalten werden.

Ein Problem dieser Methode ergibt sich immer dann wenn der Server sehr viele Clients zu bedienen hat, denn dann wird die Leitung zu diesem sehr stark belastet und zudem erfordert auch die Speicherung der Daten sehr viel Speicherplatz und eine extrem hohe Performance der Festplatte des Servers.

### 5.2 Piranha

Im Gegensatz zum „Condor“-System, vertraut die Vorgehensweise „Piranha“ darauf, dass der Programmierer der Software, die auf den einzelnen Cluster-Rechnern läuft, selbst festlegt, wie man bei jeder individuellen Berechnung

die Daten vor Verlust schützen kann. Hier wird vom Server nichts derartiges unternommen, lediglich kann bei Erkennung eines Defekts oder bei einem manuellen Abbruch der Berechnung auf einem Client ein Benutzerprogramm aufgerufen werden, das das individuell angelegte Backup bearbeiten kann. Der unschlagbare Vorteil dieser Methode zeigt sich darin, dass hier kaum verschwendete Systemleistung auftritt. Der Programmierer des Client-Programms kann eine optimal auf die Bedürfnisse der jeweiligen Aufgabe zugeschnittene Sicherungsfunktion einbauen oder auch nicht, wenn dies nicht benötigt wird.

### 5.3 Sprite/Locus

Dieser sehr elegante Ansatz beschreibt das Vorgehen der Übertragung eines Rechenprozesses von einem Rechner auf einen anderen ohne dass dabei Datenverlust auftritt. Diese Technik lässt sich freilich nur dann anwenden, wenn dieser Prozess vom Benutzer manuell ausgelöst wird, bei einer technischen Störung wird dies nicht funktionieren.

Es ist leicht nachzuvollziehen, dass diese Methode ein technisch sehr ausgefeiltes und aufwändiges System darstellt. Es müssen dabei nicht nur die bereits berechneten Daten übertragen werden, sondern auch alle Zustände, die das Programm auf dem Computer herstellt. Dazu zählen zum Beispiel geöffnete Dateien, Prozessorregister- und Arbeitsspeicher-Inhalte sowie angesprochene Peripherie. Um alle diese Zustände auf dem Zielcomputer wieder herstellen zu können, bedarf es zum einen nahezu exakt gleicher Rechner und zum anderen braucht man ein speziell dafür geeignetes Betriebssystem.

## 6 Programmiermodelle

Der Programmierer, der ein Programm schreibt, das später auf einem Cluster oder einem Multiprozessorsystem betrieben werden soll, kann mit dem verwendeten Programmiermodell direkten Einfluss auf die Performance des gesamten Systems ausüben. So orientiert sich diese Vorgehensweise meist an der Zahl der aktiven Prozessoren (beim SMP-System) oder Clients (beim Cluster). Da sehr viele moderne Cluster einen Zusammenschluss mehrerer kleiner SMP-Systeme darstellen, sind hier natürlich derartige Vermischungen auch einzubeziehen und vom Programmierer zu berücksichtigen.

Der Ausdruck *Small-N* bezieht sich hierbei auf Systeme mit wenigen Prozessoren oder Clients, üblicherweise einzelne Mehrprozessorsysteme mit bis

zu 16 Prozessoren. *Large-N* dagegen beschreibt eine größere Anzahl aktiver CPUs oder Clients und bezieht sich meist auf Cluster mit mehreren Clients.

## 6.1 Small-N

Das Problem eines Mehrprozessorsystems zeigt sich oft im Umgang mit den Ein-/Ausgabegeräten des Computers. Ein Programm kann durch verzögerte Kommunikation nach außen erheblich ausgebremst werden und dadurch kann es passieren, dass wertvolle Prozessorzeit verloren geht. Um diesem Missstand zu beheben, kann der Programmierer sein Programm in mehrere *Threads* aufteilen, die mehr oder weniger unabhängig voneinander agieren können und sobald ein Thread von der Peripherie angehalten wird, belegt dieser den Prozessor nicht weiter sondern ein anderer Thread kann die CPU solange nutzen bis der ursprüngliche Thread wieder aktiv ist.

## 6.2 Large-N

Dieses System lässt sich immer dann besonders gut anwenden, wenn die Zugriffe auf die Ein-/Ausgabegeräte zwar nötig sind um die Arbeit fortzusetzen, aber die anderen Prozessoren oder Cluster nicht davon betroffen sein sollen, wenn das Programm von der Hardware kurzzeitig gestoppt wird.

Dabei wird das Programm in einzelne Prozesse aufgeteilt, was letztendlich bedeutet, dass ein und dasselbe Programm mehrfach gestartet wird oder sich selbst im Speicher dupliziert. Das Resultat ist eine völlige Unabhängigkeit der einzelnen Rechenprozesse. Jeder Prozess kann eine CPU oder einen Client für sich in Anspruch nehmen. Bei der Verwendung dieses Systems in einem Cluster zeigt sich auch hier der Vorteil von mehrfach vorhandener Hardware. Bei nur einem Prozess auf einem Client braucht auch nur dieser Prozess Zugriff auf die Hardware und es kommt nicht zu Störungen der Prozesse untereinander.

## 7 Single-System-Image

Um ein Programm zu schreiben, das komplizierte Berechnungen auf einem Cluster durchführen soll, bedarf es sehr viel manueller Anpassung. Denn nur wenn das Programm optimal auf die Gegebenheiten des Clusters zugeschnitten ist, kann es dessen Leistung auch optimal nutzen. Um dieses Problem zu lösen, braucht man ein so genanntes *Single-System-Image*, kurz SSI. Ein solches System hat die Aufgabe, den kompletten Cluster so zusammen zu

fassen, dass der Benutzer und der Programmierer nur ein einziges System sehen, das selbstständig die Belastung auf die einzelnen Prozessoren oder Clients aufteilt.

Mit Hilfe eines derartigen Systems kann der Programmierer einfach so viele Prozesse erzeugen wie für die Bearbeitung der Aufgabe nötig sind. Das SSI-Verwaltungs-System übernimmt dann die Verteilung entweder auf mehrere CPUs eines SMP-Systems oder auf mehrere Clients eines Clusters. Somit können auch mehrere Berechnungen parallel erfolgen, die dann jeweils eine Teilmenge der Computer des Clusters zugewiesen bekommen.

Die Praktische Umsetzung eines wirklichen SSI funktioniert bisweilen nur sehr eingeschränkt, da eine völlig automatische Verteilung als Voraussetzung 100% identische Computer benötigt. Dies betrifft zum einen die Hardware und zum anderen, was viel mehr Probleme bereitet, die Software. Bei einem speziell dafür eingerichteten Cluster ist es nicht weiter relevant, da dort ja nach Bedarf ein passendes Betriebssystem installiert werden kann. Wenn man jedoch normale Arbeitsrechner als Teil eines Clusters betrachten möchte, wird es schwierig ein und dasselbe Programm auf verschiedenen Gegebenheiten auszuführen.

Das SSI-Prinzip ist ein leistungsfähiger Ansatz zur eleganten Programmierung eines Clusters. Die jeweilige Implementierung ist dagegen weiterhin eine Arbeit die speziell an das betreffende System angepasst werden muss. Momentan werden solche SSI-Systeme als Aufsatz auf das Betriebssystem eingesetzt und damit erhält man einen spürbaren Verlust. Eine effizientere Implementierung könnte direkt im Kern des Betriebssystems erfolgen, so dass die Lastverteilung direkt an die Hardware angepasst werden kann.

## 8 Zusammenfassung und Ausblick

Ein Cluster ist nicht gleich einem Cluster, denn jeder Zusammenschluss mehrerer Computer oder Prozessoren zu einem großen Rechner resultiert in einem individuellen System. Daher ist es auch nicht möglich klare Grenzen zu ziehen, welche Bauformen als SMP und welche als Cluster anzusehen sind. Stark abhängig vom späteren Aufgabenbereich und vom Budget des Eigentümers gibt es immer wieder ganz besondere Bauformen.

Die Kommunikation innerhalb eines Clusters lässt sich nach der „Exposed“- oder der „Enclosed“-Methode bewerkstelligen, auch hier sind Durchmischun-

gen durchaus möglich und werden auch praktiziert. So kann für eine spezielle Aufgabe die Rechenleistung mehrerer einzelner Cluster zu einem großen Komplex über das Internet verbunden werden. Hier muss der Programmierer die Durchmischung der Übertragungsgeschwindigkeiten berücksichtigen und kann damit den entstehenden Engpass kompensieren. Die Hardware-Hersteller haben sich hierbei nicht auf einen Standard geeinigt und damit kann ein bestehendes Hochgeschwindigkeitsnetz nach der „Enclosed“-Methode später nur mit weiteren Komponenten des betreffenden Herstellers aufgerüstet werden.

Ein Cluster lässt sich sehr leicht in der Größe Variieren, wenn man die Rechenleistung nach dem Aufstellungsprinzip „Campus-Wide“ verteilt und damit auch die verborgenen Ressourcen normaler Arbeitsplatzrechner nutzen kann. Die daraus resultierenden Probleme der Unsicherheit und der schlechteren Kontrollierbarkeit muss der Betreiber mit den eingesparten Kosten abwägen und kann auch hier beliebige Mischformen anstreben.

Die Erzeugung eines *Single-System-Image* erfordert eine sehr enge Zusammenarbeit der einzelnen Rechner und zudem eine Unterstützung im Kern des Betriebssystems um optimale Ergebnisse zu erhalten. Die Implementierungen als Aufsatz auf dem Betriebssystem erzeugt dabei immer unnötige Verluste.

Als zunehmendes Problem der Großrechner stellt sich die Übertragungsgeschwindigkeit dar. Und zwar die Kommunikation innerhalb eines einzigen Computers. Die Übertragungsraten von Rechner zu Rechner haben sich in der letzten Zeit sehr rasant erhöht. Dagegen sind die Geschwindigkeiten innerhalb eines Rechners, also zwischen Prozessor und Arbeitsspeicher sowie auch die Anbindung an die Peripherie nicht sehr viel schneller geworden.

## Literatur

[Pfi95] Gregory F. Pfister: „In Search Of Clusters“, 1995

[Kep00] Universität Tübingen: „KEPLER - MPP Rechner des SFB382/ZDV“, 2000  
<http://kepler.sfb382-zdv.uni-tuebingen.de/kepler/index.shtml>

[Set96] University of Berkely: „Search for Extraterrestrial Intelligence (SETI)“, 1996  
<http://setiathome.ssl.berkeley.edu/>