Neuer Computeserver

Daniela-Maria Pusinelli
pusinelli@cms.hu-berlin.de

Beowulf, LINUX-Cluster, Computeservice, Batchbetrieb, LSF

Seit dem Umzug des Computer- und Medienservice nach Adlershof steht den Mitarbeiter/innen der Humboldt-Universität ein neuer Computeserver zur Verfügung. Das LINUX-Cluster CLAN (Cluster of Linux Application Nodes) ist ein Beowulf-Cluster. Es besteht aus 32 PCs, die mit einem schnellen Netz (Myrinet 2000) über einen Masterknoten miteinander verbunden sind. Das Rechnersystem ist einerseits mit entsprechender Software zur Programmentwicklung ausgerüstet und andererseits mit Anwendungssoftware, wie Turbomole und Gaussian, ausgestattet. Zur Parallelisierung der Anwendungen steht MPI (Message Passing Interface) zur Verfügung.

Was ist ein Beowulf-Cluster?

Beowulf ist ein englisches Poem über einen großen, kräftigen, couragierten Helden, der das Monster Grendel schlägt. In Anlehnung an diese poetische Geschichte haben Donald J. Becker und Thomas Sterling [1] 1994 den Begriff »Beowulf-Cluster« geprägt. Sie verstanden darunter ein Rechnersystem, das aus gewöhnlichen, preiswerten, mit 2–4 Prozessoren ausgestatteten Rechnern besteht, die miteinander durch ein leistungsfähiges Netzwerk verbunden sind. Durch das frei verfügbare Betriebssystem LINUX entstand aus dem Netzwerk von »kleinen« Computern ein leistungsfähiger »großer« Computer, der es mit den herkömmlichen Supercomputern aufnehmen kann. Die Parallelisierung der Programme erfolgte über Kommunikationsbibliotheken wie PVM und MPI. Je schneller das Netzwerk wurde, desto leistungsfähiger wurde das Gesamtsystem, besonders bei Anwendungen, die sich gut parallelisieren lassen und deren Kommunikation hoch ist. Das Netzwerk ist der Computer, war das neue Schlagwort. So gelang es diesen »Eigenbauten« auf die vorderen Plätze in die Liste der Top 500 Supercomputer [2] zu gelangen. Beispielsweise belegt ein Rechnersystem des NCSA, bestehend aus 2500 P4 Xeon Prozessoren mit 3.06 GHz, den Platz 4. Von großem Vorteil ist das sehr gute Preis-Leistungs-Verhältnis, das auch akademischen Einrichtungen den Zugang zum Hochleistungsrechnen ermöglicht. Neben den geringeren Beschaffungskosten entfallen die enormen Hard- und Software-Wartungskosten herkömmlicher kommerzieller Supercomputer. Andererseits darf nicht außer Acht gelassen werden, dass diese Systeme einen hohen manuellen Aufwand für die Betreuung und Pflege benötigen. Auch an die Sicherheit des Rechnersystems müssen hohe Anforderungen gestellt werden, da LINUX ein Public Domain Betriebssystem und die Einbruchgefahr groß ist.

CLAN = Cluster of Linux Application Nodes im CMS

Die Zentraleinrichtung CMS entschied sich bei der Beschaffung eines neuen zentralen Computeservers für ein Beowulf-Cluster, nachdem die einzelnen Institute nach ihren Anforderungen befragt wurden. Die Beschaffung erfolgte mit dem Umzug des CMS auf den Campus Adlershof Mitte des Jahres 2003. Für das Kommunikationsnetzwerk wurde Myrinet2000 gewählt, da es ein sehr schnelles Netzwerk mit kurzen Latenzzeiten ist. In der ersten Ausbauphase bestand das Cluster aus einem Masterknoten und 24 Rechenknoten. Durch eine Erweiterung zu Beginn des Jahres 2004 konnte die Zahl der Rechenknoten auf 32 erhöht werden. Außerdem ist ein dedizierter Knoten als Ersatzmaster aufgerüstet worden, der im Falle des Ausfalls des ursprünglichen Masterknoten dessen Funktion übernimmt. Alle Knoten sind 19-Zoll-Einschübe mit einer Höheneinheit, die Masterknoten belegen zwei Höheneinheiten. Hinzu kommen diverse Switche (KVM, APC, Myrinet) und ein TFT-Bedienpanel. Das gesamte System belegt damit etwas mehr als einen 19-Zoll-Schrank.

Prozessoren	Dual Xeon
Taktfrequenz	2,6-3,0 GHz
Hauptspeicher	4 GB
Daten Caches	512 KB L2 Cache
Plattenspeicher	150 GB lokale Platte
Netzkomponenten	Fast Ethernet und Myrinet 2000

Tab. 1: Aufbau der Rechenknoten

Clusterverwaltung

Das Beowulf-Cluster wird über einen Masterknoten verwaltet. Alle Knoten werden über das Netz gestartet, es wird ihnen ein auf dem Master befindliches Slave-Image installiert und eine

DHCP-Adresse zugeordnet. Das hat den Vorteil, dass beim Ausfall einer Platte oder einer anderen Hardware auf einem Knoten die Hardware nur ersetzt zu werden braucht und dann der Knoten neu gestartet werden kann. Beim Austausch einer Harddisk dauert es etwa 10 Minuten, bis der Knoten wieder verfügbar ist. Eine Änderung an der Konfiguration der Knoten wird auf dem virtuellen Image vorgenommen und wird nach dem Reboot aktiv. Sollen Änderungen sofort aktiviert werden, so gibt es Cluster-Skripte, die auf allen Knoten Änderungen vornehmen: zum Starten (`pstart-slaves`), Stoppen (`pstop-slaves`) und Aktualisieren (`pupdate-slaves`) der Knoten, Kopieren von Daten (`pcopy`) und Ausführen von Kommandos (`pexec`). Die Kommandos werden auf allen Knoten ausgeführt, die in einer Datei `nodes.conf` definiert sind. Sie können auch für einzelne Knoten ausgeführt werden. Mit dem Kommando `pping` ist ein einfaches Überprüfen der Erreichbarkeit aller konfigurierten Knoten möglich.

Zugang zum Server

Mitarbeiter, die Aufträge parallel rechnen möchten und eine Vielzahl von Prozessoren für ihre Berechnungen benötigen, werden auf dem Cluster zugelassen. Sie beantragen ein Kennzeichen bei der Benutzerberatung des Computer- und Medienservice. Zusätzlich muss der Zugang zum Computeservice frei geschaltet werden. Berechtigte Nutzer loggen sich auf dem Masterknoten *clan.cms.hu-berlin.de* mittels Secure Shell ein. Eine Firewall schirmt diesen nach außen hin ab und ermöglicht ein Login nur aus der HU-Domäne. Ein direktes Einloggen auf die einzelnen Knoten aus dem HU-Netz ist nicht möglich, ein Zugang über den Master aber im Bedarfsfall.

Auf jedem Knoten des LINUX-Clusters stehen den Nutzern Dateisysteme zur Ablage von temporären Daten zur Verfügung: ein lokales Dateisystem `/scratch` und ein NFS-Dateisystem `/scratch2` des Masters. Zu beachten ist, dass die `/scratch` Verzeichnisse einem Kontrollmechanismus unterliegen und Daten dort nicht für einen längeren Zeitraum verbleiben können.

Daten, die permanent abgelegt werden sollen und die im Home-Verzeichnis keinen Platz finden, können entweder im Verzeichnis `/perm` oder im erweiterten Speicher des Tivoli Storage Manager (TSM) gesichert werden.

Entwicklungsumgebung

Für die Programmentwicklung sind die Compiler von Intel, der Portland Group, der Numerical Algorithms Group (NAG) sowie die frei verfügbaren des GNU-Projektes installiert. Zu allen Compilern sind die dazugehörigen Debugger verfügbar. Außerdem steht der parallele Debugger TotalView der Firma Etnus [3] zur Verfügung. Die Parallelisierung der Programme erfolgt mit MPICH. Die Tabelle 2 liefert eine Zusammenstellung der Entwicklungssoftware. Für eine optimale Performance können die Werkzeuge `pgprof` von der Portland Group und `vampir` von der Firma Pallas [4] verwendet werden.

Prinzipiell können auch serielle Programme gerechnet werden, aber in der Regel ist der Rechner parallelen Programmen vorbehalten. Zum Parallelisieren von C- oder Fortran-Programmen stehen die MPI-Bibliotheken (Message Passing Interface) zur Verfügung. Die Wahl des Compilers wird durch Umgebungsvariablen gesteuert, die automatisch beim Login gesetzt werden.

Die Kommandos `mpicc`, `mpiCC`, `mpif77`, `mpif90` erzeugen parallele Programme, in die die MPI-Bibliotheken bereits eingebunden sind. Sie werden mit dem Kommando mpirun gestartet. Die Wahl des Compilers, der sich hinter dem Aufruf von beispielsweise `mpif90` verbirgt, erfolgt über die vordefinierte Umgebung, die nach Abarbeitung eines der `/usr/local/bin/use-*` Skripte gesetzt wird.

Entscheidend für die Wahl des gewünschten `use-*` Skriptes ist einerseits die Wahl des Compilers und andererseits die des Kommunikationsnetzes. Programme, die sehr viel kommunizieren, sollten unbedingt über die schnellere Myrinet2000-Verbindung arbeiten. Das heißt, es muss das `use-*` Skript verwendet werden, welches die Myrinet2000-Bibliotheken (GM) einschließt. Soll beispielsweise das Programm zur Berechnung der Zahl π parallelisiert werden, so könnte das folgendermaßen aussehen:

```
cd /scratch2/$USER
cp /scratch2/test/cpi.c .
. /usr/local/bin/use-mpich-125-
   gm2010-intel.sh
mpicc -o cpi cpi.c
mpirun -np 8 -machinefile nodefile cpi
```

Das erzeugte Programm `cpi` ist ein paralleles Programm, das mit dem Intel Compiler übersetzt wurde und die Myrinet2000- sowie die MPI-Bibiliotheken eingebunden hat, d. h. die einzelnen Tasks kommunizieren über das Myrinet2000-Netzwerk. Die Datei `nodefile` enthält die Knotennamen, auf denen das Programm rechnen soll.

Zum Compilieren ist die LSF-Queue `compile` eingerichtet worden. Sie hat die höchste Priorität und wird direkt auf einen freien Knoten geschickt (es ist aus diesem Grund immer ein Knoten frei), um die Übersetzung des Programms zu starten (siehe auch den Artikel zur Ressourcenverwaltung).

Betriebssystem	LINUX Red Hat 7.2
Compiler	Fortran: ifc, pgf77, pgf90, pghpf, f95, g77 C: icc, pgcc, gcc C++: icc, pgCC, g++
Parallelisierung	MPICH, LAM, GM
Bibliotheken	BLAS, LAPACK, NAG, MKL
Debugger	idb, pgdbg, gdb, totalview
Performance	pgprof, vampir

Tab. 2: Zusammenstellung der Entwicklungssoftware

Anwendungssoftware

Als Anwendungssoftware werden die Pakete Turbomole und Gaussian zur Verfügung gestellt.

Turbomole

Turbomole ist ein Programmpaket für ab initio quantenchemische Berechnungen, das von der Gruppe um Prof. Dr. Reinhard Ahlrichs an der Universität Karlsruhe entwickelt wurde.

Die parallele Version ist MPI basiert und verwendet MPICH (mit ch_p4), allerdings nicht das Myrinet2000-Netzwerk. Auf dem Cluster ist die Version 5.6 installiert. Die Moduln `dscf`, `grad`, `mpgrad`, `rdgrad` und `ridft` sind parallelisiert, alle anderen laufen seriell. Alle Variablen sind automatisch voreingestellt, nachdem die Datei `/volume/skel/.profile` beim Login abgearbeitet wurde.

Alle Turbomole Rechnungen sind im Batchbetrieb zu starten, d. h. die Rechnung ist in ein Batchskript einzupacken. In diesem muss das Skript `use-mpich124-intel.sh` aufgerufen werden. Sie wird an das LSF geschickt mit dem Kommando

```
bsub < /perm/test/run_turbo16
```

Gaussian

Das Programmpaket Gaussian dient der Berechnung der elektronischen Struktur von Molekülsystemen. Es ist für theoretische und experimentelle Chemiker von Interesse ist.

Gaussian98 ist auf dem Cluster in der Version A11.3 installiert. Es kann auf einem Knoten mit 2 CPU als SMP-Version oder als serielle Version gerechnet werden. In der Datei `Default.Route` sind der Plattenplatz auf 30 GB und der Hauptspeicherbedarf auf 2 GB begrenzt.

Rechnungen sind im Batchbetrieb auszuführen. Dazu ist ein Skript zu erstellen, in dem die Gaussian Rechnung gestartet wird. Ein Beispielskript `run_g98` befindet sich im Verzeichnis `/perm/test`. Es ist abzuschicken mit `bsub < run_g98`. Die Laufzeiten der Tests aus der Testsuite auf einem Prozessor sind vergleichbar mit denen auf einem der COMPAQ ES40.

Gaussian03 ist ebenfalls auf dem Cluster installiert und kann analog zu Gaussian98 gestartet werden. Die parallele Linda-Version ist in Vorbereitung. Die Laufzeiten der Testsuite ergaben ein Speedup von 3, bezogen auf eine CPU, gegenüber der alten Version Gaussian98.

Laufzeitverhalten paralleler Anwendungen

Von einzelnen Arbeitsgruppen werden noch andere Pakete, wie z. B. CPMD und VASP (beides Anwendungen aus der Chemie) genutzt [5].

- CPMD
  beispielsweise benötigt viel Speicher und profitiert von dem schnellen Myrinet2000 Kommunikationsnetz. Der Zeitunterschied zwischen CPU- und Echtzeit ist verschwindend gering und liegt unterhalb einer Minute. Allerdings ist der Speedup bei Verwendung von 2 Prozessoren je Knoten viel geringer als bei einem je Knoten. Es ist aber trotzdem sinnvoll, beide CPU zu verwenden, um nicht eine leer laufen zu lassen.

Programm	Prozessoren	Knoten	Speedup
CPMD	8	4	4,35
	8	8	7,07
VASP	8	4	5,51
	8	8	7,84

Tab. 3: Laufzeitverhalten verschiedener Anwendungen

- VASP
  benötigt ebenfalls viel Speicher, der Verlust an Speedup bei der Verwendung von 2 Prozessoren ist aber geringer als bei CPMD. Er beträgt nur etwa ⅓.

Literatur

[1] http://www-106.ibm.com/developerworks/library/l-beow.html
[2] http://www.top500.org/list/2003/11/
[3] http://www.etnus.com/Products/TotalView/index.html
[4] http://www.pallas.com/e/products/vampir/
[5] Laufzeitdaten von Ch. Tuma und Dr. A. Hofmann, Institut für Chemie