PARALLELES RECHNEN

Erste Ergebnisse mit MPI auf der NEC SX-4

Message Passing-Anwendungen auf einer shared-memory Architektur

Erste Ergebnisse mit MPI auf der NEC SX-4

Jürgen Lepper / Thomas Beisel / Rolf Rabenseifner

Anfang Juli 1996 wurde der neue Höchstleistungsrechner NEC SX-4/32 am Rechenzentrum der Universität Stuttgart installiert. Der Artikel beschreibt die Anwendung von MPI auf diesem shared-memory Rechner und zeigt erste Ergebnisse. Es soll verdeutlichen, wie MPI auf der NEC SX-4 verwendet wird, wo Probleme mit den derzeitig vorhandenen Implementierungen sind und welche Leistung auf dem Multiprozessorsystem möglich ist. Dies soll insbesondere den Benutzern helfen, deren Anwendungen auf der Basis des Message Passing- Modells bereits parallelisiert sind. Mit einem einfachen Pingpong-Programm werden die Kommunikationszeiten und die Datentransferraten für MPI/SX und MPICH bestimmt, und eine CFD-Applikation zeigt die Leistung mit MPI/SX bei einer realen Anwendung.

Die NEC SX-4 ist ein Multiprozessorsystem mit 32 Vektorprozessoren und einer shared-memory Architektur [1], das auch die Verwendung des Message Passing-Programmiermodells unterstützt. Neben PVM und PARMACS wird MPI-1 auf dem Rechner angeboten, wobei eine spezielle Implementierung von NEC (MPI/SX) als auch MPICH verfügbar sind. Die beiden Implementierungen unterscheiden sich u. a. durch die Vorgehensweise, wie die parallele Applikation vom System bearbeitet wird: Als Thread oder als Unix-Prozess [2]. Diese Unterscheidung ist aber nur internes Detail der Implementierung, denn für parallele Anwendungen handelt es sich in beiden Fällen um eigenständige MPI-Prozesse (vgl. auch /HLRS/mpi_sx4.html).

In diesem Artikel werden die Erfahrungen mit der aktuellen MPI-Umgebung auf der Architektur der NEC SX-4 beschrieben und einige Ergebnisse mit parallelen Applikationen dargestellt. Der Schwerpunkt soll dabei auf den Anwendungen liegen, die zuvor für eine distributed-memory Umgebung parallelisiert worden sind und auf die NEC SX-4 portiert werden sollen.

Verwendung von MPI

MPI/SX

Die MPI/SX-Implementierung beinhaltet zwei unterschiedliche Vorgehensweisen: Eine thread-basierte und eine prozeß-basierte. Die benötigten Dateien für beide Varianten liegen unter `/usr/lib` bzw. `/usr/include`, sowohl für C als auch für Fortran. Für das Übersetzen und Linken müssen aber die *Floating Point*-Zahlenformate und -Arithmetik berücksichtigt werden. Da die NEC SX-4 neben IEEE- auch Cray- und IBM-Zahlenformate unterstützt, gibt es für jedes Format eine eigene Bibliothek, die

mit einem anders gewählten Zahlenformat im Programm nicht kompatibel ist. Es wird empfohlen die IEEE-Darstellung zu verwenden, die auf der NEC SX-4 defaultmäßig eingestellt ist. Durch die Option `-float0` und die Umgebungsvariable FLMOD = float0 kann dieses Format auch explizit gesetzt werden.

Parallele Applikationen werden normalerweise mit:

```
cc -pthread -o C_code c_code.c -lmpi -lpthread
f77 -I/usr/include -P multi -G local -o f_code f_code.f
-lmpi -lpthread
```

erzeugt. Soll bei Fortran-Programmen die 8-Byte-Genauigkeit (Option `-ew`) für alle INTEGER-, REAL- und DOUBLE PRECISION-Datentypen genutzt werden, muß folgender Aufruf verwendet werden:

```
f77 -I/usr/include -P multi -G local -ew -o f_code f_code.f -lmpiw -lpthread
```

Die Option `-G local` ist für Fortran in jedem Fall erforderlich, um alle globalen Daten in lokale Objekte zu verwandeln, damit bei einer thread-basierten Anwendung jeder MPI-Prozess einen getrennten Variablenbereich besitzt.

Der Aufruf der parallelen Applikation mit MPI/SX basierend auf Threads erfolgt mit:

```
mpisx -p #procs -thread -e code args ...
```

Der vollständige Pfad zum ausführbaren Programm muß angegeben werden, falls sich dieses nicht im aktuellen Verzeichnis beim Aufruf von mpisx befindet. Falls statt des thread-basierten MPI/SX das prozeß-basierte benutzt werden soll, entfällt die Option `-thread`. Wegen der deutlich schlechteren Kommunikationsleistung wird dies jedoch nicht empfohlen. Die Anzahl der Prozesse bzw. Threads wird durch `#procs` bestimmt.

Die Verwendung von MPI/SX basierend auf Threads verlangt bei SPMD-Programmen auch bei unterschiedlichen Dateinnamen einen eigenen Filedescriptor (C) bzw. Unit- Nummer (Fortran) für jeden Thread. Auf der NEC SX-4 sind derzeit nur 100 Filedes-kriptoren gleichzeitig mö glich, so daß z. B. nur maximal drei Dateien auf 32 Prozessoren offen gehalten werden können. Diese Einschränkung ist nicht erforderlich, wenn die parallele Applikation das prozeß-basierte MPI/SX benutzt.

Eine aktuelle Liste aller bekannten Einschränkungen von MPI/SX auf der NEC SX-4 ist unter /HLRS/mpi_sx4.html zu finden.

MPICH

Neben MPI/SX ist auch das MPICH des Argonne National Laboratory installiert, wobei diese Version nicht offiziell unterstützt wird. Die Implementierung wird jedoch bereitgestellt, um denjenigen Benutzern eine alternative MPI-Umgebung zu geben, die Probleme mit ihren Anwendungen unter MPI/SX haben.

Alle benötigten Dateien sind unter `/usr/local/mpich` zu finden, und der vollständige Pfad zu den Bibliotheken und include-Dateien lautet: `/usr/local/mpich/lib/SX_4_float0/ch_lfshmem` Im Gegensatz zu MPI/SX liegen die Dateien von MPICH z. Z. ausschließlich im IEEE-Format (float0) vor.

Es wird empfohlen, die parallele Applikation mit den entsprechenden Skripten von MPICH (`mpicc`, `mpif77`) zu erzeugen. Soll stattdessen wie üblich mit `cc` und `f77` gearbeitet werden, müssen die

geänderten Bibliotheken verwendet werden, und die Optionen `-pthread` bei **C**, sowie `-P multi` und `-G local` bei Fortran entfallen.
Der Aufruf des parallelen Programms erfolgt über:

```
/usr/local/mpich/lib/SX_4_float0/ch_lfshmem/mpirun
-np #procs code args ...
```

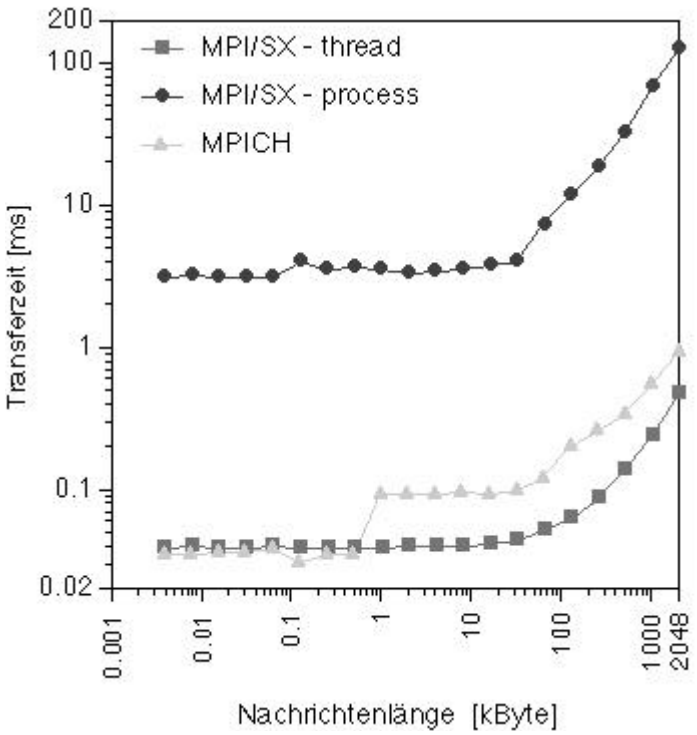
Der direkte Zugriff auf `/usr/local/mpich/` sollte nicht verwendet werden, da sich in diesem Verzeichnis nur die jeweils zuletzt installierte Version befindet, die ein anderes Zahlenformat aufweisen kann als die Anwendung.

Obwohl MPICH nicht alle Installationstests fehlerfrei absolviert hat, ist es für C-Programme nahezu uneingeschränkt anwendbar, während sich bei einigen Fortran-An-wendungen spezielle Probleme ergeben haben, die derzeit noch nicht gelöst sind. Aktuelle Informationen sind wiederum auf den WWW-Seiten des HLRS zu finden.

Kommunikationsleistung

Die Leistung der beiden Implementierung wird mit der einfachen Pingpong-Applikation getestet. Bei diesem Programm werden zwei Prozessoren verwendet, die gegenseitig Nachrichten mit verschiedenen Längen austauschen. Um die Aussagefähigkeit der Er-gebnisse zu verbessern, werden die gemessenen Werte über einige hundert Durchläufe gemittelt. Die Nachrichtenlänge variiert bei allen Messungen jeweils von 0 Byte bis 2 MByte. Das Programm ist sowohl in C als auch Fortran77 geschrieben, und verwendet ausschließlich die Standard *MPI_Send* und *MPI_Recv*-Aufrufe.

Nach der Spezifikation von MPI-1 ist die Implementierung des Standard *MPI_Send* nicht festgelegt [3]. Es ist daher wichtig darauf hinzuweisen, daß bei MPI/SX auf der NEC SX-4 dieser Aufruf blockierend ist.



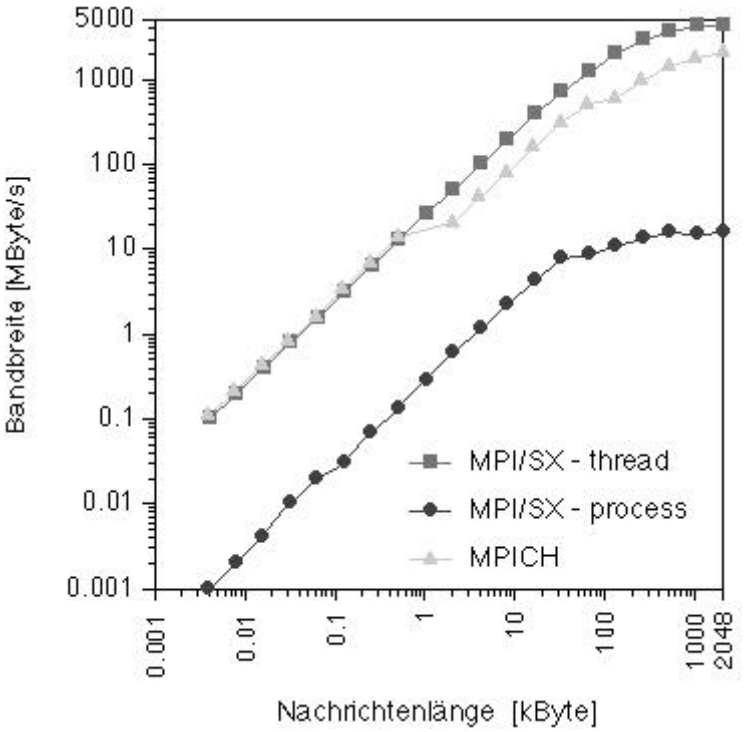


Bild 1: Mittlere Kommunikationszeit und Datentransferrate mit MPI/SX und MPICH

Die Ergebnisse mit MPI/SX sind durch Verwendung von Threads und Prozessen ermittelt worden. Aus Bild 1 wird deutlich, daß die Verwendung von MPI/SX basierend auf Threads die schnellste Kommunikation liefert, mit einer *Latency* von 36 µs und einer Datentransferrate von 4,2 GByte/s bei einer Nachrichtenlänge von 2 GByte. Wird anstelle von Threads mit Prozessen gearbeitet, verschlechtert sich die Leistung merklich: Die *Latency* liegt bei 3,2 ms und die Datentransferrate erreicht auch bei großen Nachrichten nur noch 16 MByte/s.

Obwohl MPICH ausschließlich mit Prozessen arbeitet, liegt die *Latency* hier mit 33 µs ähnlich niedrig wie beim thread-basierten MPI/SX, und die Datentransferrate erreicht noch 2,1 GByte/s für Nachrichten mit einer Länge von 2 GByte.

Die mittleren Kommunikationszeiten von MPICH liegen etwa doppelt so hoch wie bei MPI/SX basierend auf Threads, während bei MPI/SX auf der Basis von Prozessen die Zeiten um einen Faktor 100 über diesen minimalen Werten liegen.

Aus den Ergebnissen folgt, daß die Verwendung von MPI/SX basierend auf Prozessen derzeit nicht als sinnvolle Alternative zum thread-basierten MPI/SX auf der NEC SX-4 gesehen werden kann.

Leistung einer CFD-Applikation

Als reale Anwendung wird hier die Leistung eines Finite-Volumen-Codes zur dreidimensionalen Simulation turbulenter und reaktiver Innenströmungen betrachtet. Obwohl nur strukturierte Gitter bei der Berechnung zulässig sind, verwendet die Datenstruktur durchweg die indirekte Adressierung. Dies erlaubt zwar eine gute Vektorisierung aller Schleifen, aber die *Floating Point Performance* auf Vektorrechnern ist entsprechend geringer als bei direkter Adressierung.

Die explizite Parallelisierung des Fortran77-Programms basiert auf dem SPMD-Programmiermodell, der statischen Domain Decomposition-Methode und der Verwendung von Message Passing-Bibliotheken. Da dasselbe Programm auf jedem Prozessor der NEC SX-4 abgearbeitet wird, können die Vektoreigenschaften dort entsprechend genutzt werden. Wegen Problemen mit MPICH

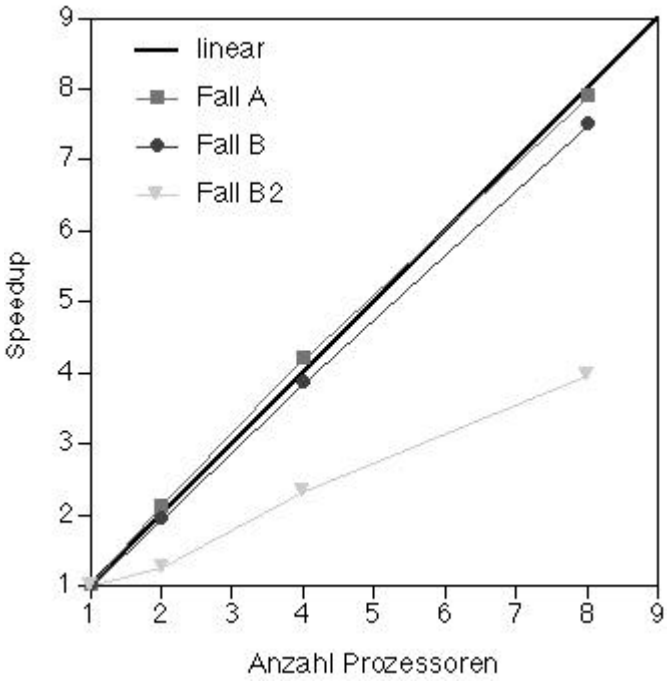
wurde ausschließlich MPI/SX auf der Basis von Threads verwendet, nachdem die bereits beschriebene Anpassung der Unit Nummern im Programm vorgenommen wurde.

Anhand einer stationären, inkompressiblen Strömung mit Verbrennung auf zwei unterschiedlich fein diskretisierten Berechnungsgittern - Fall A mit ca. 300.000 Gitterpunkten und Fall B mit ca. 2.500.000 Gitterpunkten - wird die Leistung der Anwendung be-stimmt. Alle Programmeinstellungen bleiben für die beiden betrachteten Fälle gleich und ausschließlich das zeitliche Verhalten der Applikation wird betrachtet.

Die *Floating Point Performance* und weitere Informationen (wie z.B. der Vektorisierungsgrad) werden mittels einer vorhandenen Funktion bestimmt, die durch die Option -acct beim Übersetzen und Linken aktiviert wird, und durch das Setzen der Umgebungsvariablen F_PROGINFO = DETAIL in der Arbeitsumgebung verfügbar ist [4].

Die Laufzeiten werden für 1, 2, 4 und 8 Prozessoren bestimmt, da die größte derzeit verfügbare Batchqueue auf der NEC SX-4 maximal acht Prozessoren erlaubt. Wegen der Hauptspeicherbeschränkung auf 2 GByte ist die Applikation mit normaler Genauigkeit übersetzt. Bei einer Arithmetik mit 8-Byte-Genauigkeit ist zwar eine höhere Leistung auf einem Prozessor mö glich, allerdings verdoppelt sich auch der notwendige Hauptspeicherbedarf der Anwendung.

Die Applikation zeigt für beide Fälle noch einen linearen *Speedup*, d. h. eine ideale Verringerung der Berechnungszeiten (Bild 2). Allerdings sind einige der effektivsten numerischen Lösungsalgorithmen für die linearen Gleichungssysteme implizit, und wegen der daraus resultierenden Datenabhängigkeiten nicht vektorisierbar. Bei den beiden gezeigten Berechnungen wird der implizite Stone-Algorithmus für die Druckkorrekturgleichung und der vektorisierbare Red Black Gauss-Seidel-Algorithmus für alle anderen Gleichungen verwendet, was einen Vektorisierungsgrad von ca. 97% ergibt.



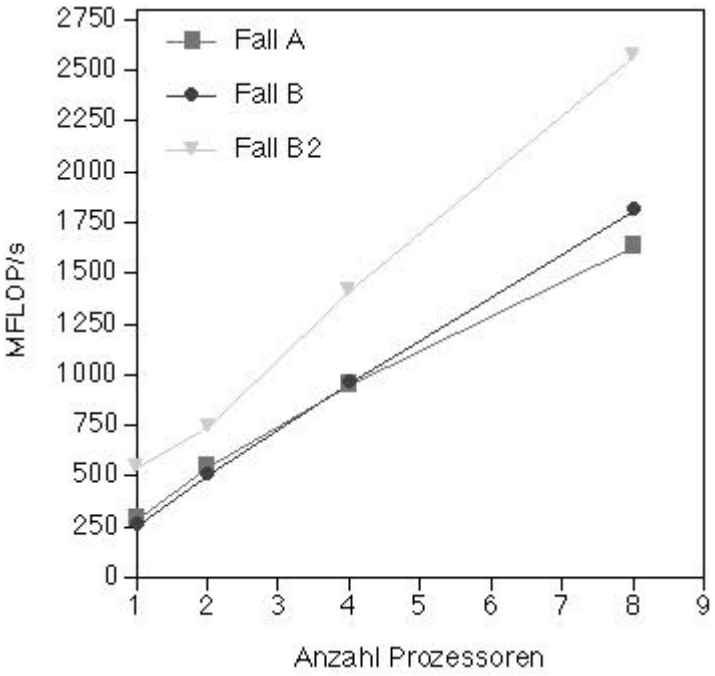


Bild 2: Speedup und *Floating Point Performance* der CFD-Applikation auf bis zu acht Prozessoren

Wird das Red Black Gauss-Seidel-Verfahren für alle Gleichungssysteme angewendet (Fall B2), ist ein Vektorisierungsgrad von über 99% möglich, aber die benötigte Anzahl an Iterationen steigt erheblich an. Bei der parallelen Applikation führt dieser vektorisierbare Lösungsalgorithmus zu einer Verringerung des *Speedup*, da die Kommunikationszeiten bei gleichem Datenaustausch einen stärkeren Einfluß auf die Gesamtlaufzeit haben. Die Unterschiede in der Lastverteilung der Prozessoren führen so bei der Synchronisation zu sichtbaren Verlusten.

Die gemessene *Floating Point Performance* ist auf acht Prozessoren für den Fall B2 mit 2,6 GFLOP/s deutlich besser als der Wert von 1,8 GFLOP/s für den Fall B. Allerdings liegen alle Ergebnisse weit unter der möglichen *Peak Performance* von 2 GFLOP/s pro Prozessor. Wegen der indirekten Adressierung im Programm sind die Ergebnisse trotzdem akzeptabel, da bei den hier gezeigten Berechnungen noch nicht alle möglichen Optimierungen eingesetzt worden sind.

Literatur

[1] Geiger, A., Küster, U., The NEC SX-4 at HWW - First Experiences in an Engineering Environment, in Supercomputer `96, Hrsg.: H. Meuer, Saur Verlag 1996
[2] Hwang, K., Advanced Computer Architecture, McGraw-Hill, 1993
[3] Message Passing Interface Forum, MPI: A Message Passing Interface Standard, Version 1.1, University of Tennessee, Knoxville, 1995
[4] NEC Corporation, SUPER-UX Fortran77/SX Programmer's Guide, G1AF11E-1, 1995

Jürgen Lepper, NA-5719
E-Mail: lepper@rus.uni-stuttgart.de

Thomas Beisel, NA-5793
E-Mail: beisel@rus.uni-stuttgart.de

Rolf Rabenseifner, NA-5530
E-Mail: rabenseifner@rus.uni-stuttgart.de