Nutzung der IMSL auf der Superdome mit Openmp:

Auf der NWZSuperdome kann die IMSL-Library genutzt werden. Hier wird gezeigt wie die IMSL aus einem Fortran-Programm aufgerufen wird.
Einige Routinen zur Lösung algebraischer Probleme sind bereits in der IMSL-Library parallelisiert. Ohne viel Aufwand können die parallelisierten Routinen auf der Superdome genutzt werden. Wir beschränken uns hier allerdings auf die Parallelisierung via Openmp.

Aufrufen der IMSL-Library aus einem Fortran-Programm:

Wir wollen als Beispiel in einem einfachen Programm in Fortran die IMSL-Library nutzen.
Beispielcode:

```
program Matrix
implicit none
real(8) :: A(500,500),B(500,500),C(500,500),t1,t2
integer(4) :: n,i,j
n=500
C=0.0d0
do i=1,n
do j=1,n
A(i,j)=dble(i+j)/2.0d0
B(i,j)=dble(i-j)/2.0d0
end do
end do
write(*,*) A,B
CALL DGEMM('n','n',n,n,n,1.0d0,A,n,B,n,0.0d0,C,n)
!1.und 2. Argument besagt, dass die Matrix A bzw. B benutzt wird, (und nicht A trans)
!3. und 4. Argument sind Anzahl der Zeilen und Spalten der Matrix C
!5. Argument: Anzahl Zeilen der Matrix A
!6.Argument: Faktor mit dem das Ergebnis multipliziert wird
!7. und 8. Argument: Matrix A und  leading dimension
!9. und 10. Argument: Matrix B und  leading dimension
!11. Argument Faktor für additive Komponente
write(*,*) 'Ergebnis:',C
end program
```

Das Programm erzeugt zwei Matrizen A, B und berechnet daraus die Zuweisung C=AB+C. Bei der Routine DGEMM handelt es sich um eine Level 3 BLAS-Routine (Basic Linear Algebra Subprograms), da eine Matrix-Matrix-Operation durchgeführt wird.

Man erkennt, dass der Aufruf einer IMSL-Routine bzw. –function genau wie der gewohnte Aufruf derselbigen verläuft. Die Dokumentation der verfügbaren IMSL-Routinen findet man unter N:\Start Menu\Programming\Math Libraries\IMSL Reference .
Die eigentliche Schwierigkeit taucht bei der Kompilierung bzw. Ausführung des Programms auf. Hier muss dem Compiler mitgeteilt werden, wo sich die IMSL-Library befindet.

Hat man sich nun auf der Superdome eingeloggt und das Beispielprogramm Matrix.f90 im Work-Verzeichnis gespeichert, in meinem Fall in dem Ordner /work/t_hein03, kann man das Programm auf folgende Weise kompilieren und ausführen:

Zunächst setzt man mit folgendem Befehl, die korrekte Kompilierungs-Umgebung:
$. /opt/IMSL/imsl/fnl600/itanium/bin

Die Kompilierung selbst geschieht durch folgende Zeile:
$ $FF $FFLAGS –o Matrix.exe Matrix.f90 $LINK_FNL_STATIC

Die Auszuführende Matrix.exe wurde dann im selben Ordner, bei mir /work/t_hein03 , erstellt. Um die Auszuführende zu starten, muss man ein PBS-Skript schreiben:

```
#PBS -q short
#PBS -N Matrix
#PBS -e Matrixerror.txt
#PBS -o Matrixout.txt
/work/ ... /Matrix.exe
```

In der letzten Zeile steht der Pfad der Auszuführenden. Anstelle der drei Punkte schreibt man also den Namen des Ordners, den man sich im Work-Verzeichnis erstellt hat.
Das PBS-Skript „Matrix.PBS“ lässt sich mit dem Befehl

qsub Matrix.PBS

starten. Ist dies geschehen sollte das Programm Matrix.exe ausgeführt werden und das Ergebnis in der Datei Matrixout.txt stehen.

Nutzung der parallelisierten IMSL-Routinen unter Fortran:

Wie schon erwähnt handelt es sich bei der Routine DGEMM um eine BLAS-Routine. BLAS-Routinen sind in der IMSL-Lybrary schon vor parallelisiert. Die Parallelisierung soll nun genutzt werden.

Am Quellcode selbst muss nichts verändert werden. Der Kompilierungs-Befehl enthält allerdings weitere Argumente.
Zunächst muss, falls nicht schon geschehen, die richtige Umgebung gesetzt werden.
$. /opt/IMSL/imsl/fnl600/itanium/bin

Die Kompilierungsbefehl sieht nun so aus:
$ $FC $FFLAGS `-openmp -O3 -mp -fpp -fpic` –o Matrixparallel.exe Matrix.f90 $LINK_FNL_STATIC

In dem PBS-Skript müssen wir der Queue nun mitteilen auf wievielen Knoten das Programm gerechnet werden soll:

```
#PBS -q short
#PBS -N Matrixparallel
#PBS -e Matrixparallelerror.txt
#PBS -o Matrixparallelout.txt
#PBS -l nodes=1:ppn=4
PARNODES=`wc -l $PBS_NODEFILE |gawk '{print $1}'`
export OMP_NUM_THREADS=$PARNODES
/work/t_hein03/Matrixparallel.exe
```

In der Zeile #PBS –l … wurde als Beispiel die Anzahl der Prozessoren, auf denen gerechnet werden soll, auf vier gesetzt.
Das Programm kann wiederum mit der Zeile

qsub Matrixparallel.PBS

gestartet werden, falls man das Skript unter diesem Namen gespeichert hat. Das Ergebnis steht dann in der Datei Matrixparallelout.txt.

Hinweis:
Die Nutzung der IMSL unter C++ ist natürlich auch möglich. Die Dokumentation hierzu ist in Arbeit.
Bei Fragen wenden Sie sich bitte an t_hein03@uni-muenster.de .

Quellen:
-IMSL-Intro, Benno Süselbeck
-Usage Notes for IMSL(R) Fortran Numerical Library Version 6.0.0