Benutzung des Convex SPP Parallelrechners

Hardwarekonfiguration

Die Convex SPP1000/XA 48 ist ein Parallelrechner mit 48 Prozessoren vom Typ HP7100. Der Rechner ist mit 6 GB Hauptspeicher und 46 GB Plattenspeicher ausgestattet. Jeder der mit 100 MHz getakteten Prozessoren verfuegt ueber 1 MB Daten und 1 MB Instruktionscache. Daraus resultiert eine Prozessor-Peak-Leistung von 200 MFLOP/s. Das Gesamtsystem erreicht damit eine Spitzenleistung von 9.6 GFLOP/s.

Subcomplex-Konfiguration

Der Rechner ist in sogenannte "Subcomplexe" partitioniert. damit Applikationen mit unterschiedlichem

Bedarf an Ressourcen (Prozessoren/Speicher) behinderungsfrei nebeneinander ablaufen koennen. Das System ist in 4 Subcomplexe mit folgenden Ressourcen und Anwendungsbereichen eingeteilt:

- System: 2 Prozessoren
- default: 6 Prozessoren
- gsm8 : 8 Prozessoren
- gsm32 : 32 Prozessoren

Beim Einloggen landet man automatisch im default-Subcomplex.

Auf gsm8 und gsm32 sollten keine interaktiven Rechenjobs laufen. Fuer dies beiden Subcomplexe sind Warteschlangen des NQS-Batchsystems konfiguriert.

Auf System laufen nur Systemdienste.

Speicherklassen

Es gibt 2 verschiedene Speicherklassen auf dem spp

node-local:
: Die Speicher wird von sequentiellen Prozessen genutzt (d.h. z.B. auch von PVM-Prozessen)

global:
: Dieser Speicher wird von parallelen Prozessen genutzt (d.h. z.B. vom parallelisierenden Compilere erzeugte Applikationen, solche die die parallelisierten Libraries verwenden und solche die cps- und cnx-Threads verwenden)

Der verfügbare Hauptspeicher wird beim Systemstart den beiden Speicherklassen zugewiesen. Die momentane Konfiguration sieht so aus:

gsm8
- 512 MB global
- 350 MB local

gsm32:
- 512 MB global pro node -> 2 GB Global Memory im gsm32 Subcomplex
- 350 MB local pro node -> Nutzen sequentielle Prozesse, die auf einem Node laufen mehr als ca. 350 MB, beginnt die Maschine Speicher auszulagern -> Die Effizienz sinkt betraechtlich, da die Platte die Geschindigkeit bestimmt!

Diese Konfiguration ist nicht endgültig, und kann bei Bedarf abgeaendert werden.

Nicht parallele Prozesse koennen ihren Speicherbedarf aus dem Global-Memory Pool befriedigen, wenn man das Executable durch den mpa-Befehl modifiziert. Dies funktioniert nur, wenn das Executable mit dem Convex Linker gebunden wurde (ESOM Format).

Vorgehensweise:

mpa -m -v -private near_shared -stacktype near_shared -specific near_shared executable

Dadurch wird das Executable so modifiziert, dass aller Speicher als near_shared deklariert wird. (Bitte man mpa studieren !)

Gaebe man far_shared an, würde der Speicher ebenfalls im globalen Speicher abgelegt, allerdings über die Nodes verteilt, was bei sequentiellen Applikationen zu Performance-einbussen führen würde. Allerdings ist es immer noch besser far_shared zu waehlen, wenn man mehr Speicher für einen Prozess haben will, als Node lokal vorhanden ist.

Denn far_shared ist schneller als Paging.

< = schneller als

lokaler cache < near_shared < far_shared < Paging

Plattenspeicher

Es exisitieren momentan 3 tmp-Bereiche

/tmp
: ist relativ klein (2GB) nicht(!) für grosse Datenmengen und Produktionslaeufe gedacht .

/tmp_gsm8
: 4 GB gross, schnell und direkt an dem Node gelegen, auf dem der gsm8 Subcomplex liegt.

/tmp_gsm32
: 8 GB gross, schnell und direkt an einem der Nodes des gsm32 Subcomlexes.

Wenn Sie den zur ihrem Subcomplex passenden tmp-Bereich nutzen, erreichen sie einen sehr guten Plattendurchsatz, wenn nicht, nur einen guten.

Abnahme

Zum 2. Jnui wurde die Convex SPP des RRZE nach einer einmonatigen Testphase abgenommen. Die Verfügbarkeit betrug in diesem Zeitraum 98%. Nach erfolgter Abnahme ging die Maschine in den regulären Produktionsbetrieb über. Gleichzeitig damit wurde die Aufsicht des RRZE (email: aufsicht@rrze) zuständig für die Verwaltung der SPP-Benutzer.

Zugang zum System

Benutzeraccount

Der Parallelrechner wurde fuer die Entwicklung paralleler Programme und deren Einsatz in der Produktion beschafft. Daher ist er nicht als Durchsatzmaschine fuer sequentielle Anwendungen gedacht. Fuer sequentielle Jobs stehen die Rechner des Compute-Server-Clusters zur Verfuegung.

Wissenschaftler/innen der FAU koennen ueber die Aufsicht des RRZE ein "Schnupperaccount" erhalten, wenn Sie parallele Programme entwickeln oder anwenden moechten und dafuer qualifiziert sind. Soll der Rechner im Produktionsbetrieb genutzt werden, ist mit der Leitung des RRZE eine Absprache ueber die Finanzierung zu treffen.

Netzanbindung

Der Rechner ist unter den Namen spp.rrze, gsm.rrze und convex.rrze bekannt. Die IP-Adresse lautet: 131.188.3.30

Hauptbenutzergruppen

- Theoretische Physik II/III
- Organische Chemie/Computer Chemie Zentrum
- Lehrstuhl für Strömungsmechanik
- Lehrstuhl für Nachrichtentechnik
- Lehrstuhl für Betriebssysteme
- Lehrstuhl für graphische Datenverarbeitung

Programmierung

Compiler

/usr/convex/bin/cc, /usr/convex/bin/fc

Debugger

cxdb

Profiler

cxpa:

Applikationen, welche die cps-Library nutzen (d.h. alle mit -O3 compilierten Programme) koennen mit dem cxpa Profiler sehr gut analysiert werden.

Kurzdokumentation

Batch-Betrieb

Es sind 2 Batchklassen konfiguriert:

gsm8
: Der Job wird auf den gsm8 Subcomplex geleitet

gsm32

Dokumentation

- Online-Dokumentation

```
Aufruf: /usr/bin/pinpoint
             oder für SUN-Liebhaber
        /usr/bin/answerbook
             oder für HP-Liebhaber
        /usr/bin/laserX

Hinweis:
     - Dies ist nur eine Vorversion im Field-Test (Drucken wird z.B.
       noch nicht unterstützt)
     - An wesentlicher Dokumentation ist bislang nur der
       C-User-Guide, der Fortran-User-Guide, NQS+ Guide und der
       Exemplar-Architekture-Guide verfügbar. Weiter Doku folgt.
```

- Online-Manual **man** (setenv MANPATH /usr/contrib/man:/usr/local/man:/usr/convex/man:/usr/convex/all/man:/usr/man)
- Handbücher in Postscriptformat sind in Verzeichnis **/local/doc/spp/ps**
- Release-Notes in Verzeichnis **/usr/doc**
- Spezielle Informationen zur Programmierung Shared Memory Programmierung , PVM-Programmierung

---

Hinweise

Backup

Der Rechner ist in das RRZE-Backup System integriert.

Benutzer-Umgebung

Unter ~demo befindet sich eine Umgebung, unter der ein Arbeiten mit allen Compilern, Libraries, Tools und PVM-Tools moeglich ist. Bitte passen Sie diese Umgebung Ihren Beduerfnissen an. Bei der Einrichtung folgender Umgebung sollten keine Probleme auftreten:

Zunaechst sollten vom HOME-Directory aus folgende Kommandos ausgeführt werden:

```
mkdir pvm3
mkdir pvm3/bin
mkdir pvm3/bin/CSPP
ln -s /usr/convex/pvm/lbi/CSPP/pvmgs pvm3/bin/CSPP/pvmgs
```

und ein Eintrag in der .cshrc folgender Art erfolgen:

```
if (-d ~/pvm3) then
 if (-d /usr/convex ) then
     setenv PVM_ROOT /usr/convex/pvm
     set path = ( /usr/convex/bin $path )
```

```
else
    setenv PVM_ROOT /local/pvm3
endif
setenv PVM_ARCH UNKNOWN
if (-x $PVM_ROOT/lib/pvmgetarch) then
        setenv PVM_ARCH `$PVM_ROOT/lib/pvmgetarch`
endif
set path = ( $path $PVM_ROOT/lib $PVM_ROOT/lib/$PVM_ARCH ~/pvm3/bin/$PVM_ARCH)
endif
```

Wichtige Kommandos

Modifizieren der Programmattribute

mpa(1) modifiziert die Attribute der Programmausführung. Es dient zum

- Starten eines Programms auf einem alternativen Subcomplex (ein anderer als `default`)
- Modifizieren der die parallele Ausführung betreffenden Attribute eines ESOM-Datei (Format eines parallel ausfürbaren Programms)

Wahl eines Subcomplexes

- Nach dem Login sind Sie in default.
- Sie koennen ein Programm in einem anderen Subcomplex starten:

```
mpa -sc gsm32 myprog     # führt myprog in gsm32 aus
```

- Sie koennen auch den Arbeits-Subcomplex wechseln, indem sie eine Shell in einem anderen Subcomplex starten:

```
mpa -sc gsm8 tcsh        # startet eine tsch in gsm8; alle programme,
                           die in dieser Shell gestartet werden,
                           laufen auch in gsm8
```

gmake

Unter /usr/contrib/bin liegt auf den SPP Knoten ein gmake. Bitte diesen Pfad in den Suchpfad $path aufnehmen.

Laufzeit-Überwachung

/local/bin/fau_top
    das bekannte top, von uns an den SPP angepasst
/usr/convex/bin/syspic
    X-Window-Tool, Anzeige, von CPU-Auslastung, Memory-Verbrauch, Paging-Aktivitaet, ...

Achtung: Beide Tools sind extrem CPU-Intensiv, weil sie sehr grosse Datenmengen aus dem Betriebssystem holen. Bitte lassen Sie diese Tools nur laufen, um einen Ihrer Programmlaeufe zu beobachten.

Convex-PVM-Programmiertools:

Der Job wird auf den gsm32 Subcomplex geleitet

Zu einem Zeitpunkt ist auf einem Subcomplex immer nur ein Batch-Job aktiv.

- In gsm8 und gsm32 dürfen nur noch kurze (< 10 Minuten) Jobs interaktive gestartet werden. Ausnahmen z.B. wegen Debugger oder Profiler Laeufen müssen abgesprochen werden. Für solche Zwecke kann die Queue gestoppt werden.
- Bei einem Job-Submit darf keine Prioritaet (-p n) angegeben werden. Alle Jobs bekommen dann die Defaultprioritaet 31. Statt dessen werden die Prioritaten der wartenden jobs alle 10 Minuten neu berechnet und gesetzt. Der Algorithmus (etwas verkürzt):
  - Wer schon in einer Queue rechnet bekommt für alle wartenden Jobs die Prioritaet 32.
  - Alle anderen werden nach der bisher im Batch verbrauchten Rechenzeit sortiert und bekommen die naechsten Prioritaeten zugewiesen. Der mit der meisten Rechenzeit die 33, der mit der wenigsten die hoechste Prioritaet.
  - Falls für einen Benutzer noch gar keine Rechenzeit protokolliert ist (neuer Benutzer). wird er noch eine Prioritaetsstufe hoeher eingestuft.

  Dies führt zu folgenden Verhalten: Wer gerade rechnet steht ganz hinten. Die anderen sind in umgekehrter Reihenfolge ihrer bisherigen Rechenzeit sortiert.

  Bei der Berechnung der bisherigen Rechenzeit wird wie folgt vorgegangen:
  - Es zaehlt nur die Rechenzeit der letzten Woche.
  - Die Rechenzeit setzt sich aus User+Systen Zeit zusammen. Falls Realzeit verwenden werden würde, würden ordnungsgemaess im Batch laufende Jobs bestraft, wenn sie durch interaktive Jobs gebremmst werden.
  - Damit das Ergebnis dennoch ungefaehr mit der eigentlich wünschenswerten Realzeit übereinstimmt, wir die User+System Zeit durch die Anzahl der Prozessoren geteilt (8 in gsm8, 32 in gsm32).
  - Jobs unter 10 Minuten gelten als Kurzlaeufer und werden nicht mitgerechnet. Bei Jobs über 12 Stunden zaehlt die Zeit, die über der 12 Stunden Grenze liegt, dreifach.

  Das System hat eine Lücke: Wenn beim Submit eine hoehere Prioritaet angegeben wird und die Queue wird leer, ehe das Prioritaetsberechnungsskript das naechste mal laeuft, dann draengelt sich der hochpriore Job vor (seine Prioritaet konnte nicht rechtzeitig runtergesetzt werden).

  DESHALB IST DIE ANGABE VON PRIORITAETEN nicht gestattet !!!!!

  Wenn Jobs gefunden werden, die mit einer solchen gesetzten Prioritaet laufen (sich also vorgedraengelt haben), behalten sich die Betreuer vor, diese Jobs abzubrechen.
- Sequentielle Jobs: Sequentielle Jobs werden bei diesem Verfahren zu gut behandelt, da auch ihre Rechenzeit durch die Anzahl der Prozessoren im Subcomplex geteilt wird (wir koennen das aus dem Accounting nicht erkennen). Daher gilt: Sequentielle Jobs gehoeren auf das Compute-Cluster!
- Parallele Jobs: Parallele Jobs sollten die Anzahl der Prozessoren ausnutzen. Falls nicht, tritt das gleiche Problem wie bei den sequentiellen ein. Wenn uns was Besseres einfaellt, werden wir das Verfahren aendern. Die beste Loesung waere die Zeit, die der Job in der Queue rechnet. Dies hat aber oben genannten Nachteil (wird durch interaktive Jobs hochgetrieben, was nicht fair ist) und ist schwieriger zu realisieren, weil die Auswertung der Accountingdaten komplizierter wird.