1.1 Der Application-Compiler - ein mächtiges Werkzeug zur Programmentwicklung auf der C2

Mit der Umrüstung ist auf der C2 ein neuer Compiler verfügbar, der sogenannte "Application-Compiler". Er ist kein eigentlicher Compiler, sondern ruft den Fortran- und/oder C-Compiler sowie den Loader auf. Im Gegensatz zu konventionellen Compilern analysiert der Application-Compiler nicht einzelne Routinen, sondern das gesamte Programm. Dabei legt er im Directory, in dem er aufgerufen wird, eine umfangreiche Datenbasis (PDB) an (etwa 1 kB pro Zeile Sourcecode), die seine Informationen zum Programm enthält. Mit diesen Informationen versehen kann er wesentlich weitergehende Fehlerprüfungen und Optimierungen durchführen als normale Compiler.

Die ausführlichen Fehleranalysen halfen in einem ersten Testlauf am RZTU, bisher unbekannte Fehler in einem 4000 Zeilen langen Fortran-Programm zu entdecken. Insbesondere falsche Parameterübergaben, Fehler in COMMON-Blöcken und Index-Überschreitungen wurden so entdeckt.

Ist ein Programm mit den normalen Compilern gut vektorisierbar, gewinnt man mit dem Application Compiler im Schnitt 15 % (Aussage CONVEX); bei Programmen, die sonst handoptimiert werden müßten, kann er aber sehr dramatische Laufzeitgewinne bewirken.

Die Schnittstelle zum Compiler ist das "build"-Programm: Ähnlich wie beim make muß man ein spezielles File namens "Buildfile" erstellen, das die Source-Files, Libraries und Optionen enthält. Allerdings bracht man im Gegensatz zu make keine Abhängigkeiten anzugeben, die findet der Compiler selbst heraus. Für gewiefte Optimierer gibt es viele Optionen und Flags (s. man build); er arbeitet aber sehr gut schon mit seinen Standard–Einstellungen.

Wer den Application-Compiler schnell mal ausprobieren möchte, sollte folgendes zu tun:

1. Der Pfad muß erweitert werden um /usr/convex/ipo/bin .
2. Im Directory mit den Sourcen muß ein Buildfile angelegt werden (s. u. für zwei einfache Beispiele).
3. "build" wird aufgerufen.

Es folgen zwei einfache Beispiele für Buildfiles:

```
#
# Alle Source-Files einzeln mit ihren Compiler-Optionen (ohne -c !)
#
main.f -O2
~/extras/sub1.f -O2
sub2.f -O2
#
# Binde folgende Libraries dran:
#
link FORTRAN -lveclib
#
# Name des fertigen Programms  (-o bla) hier, dazu build-Optionen
#   -show all : sehr geschwaetzig, am besten Output umlenken !
#   -times:     wie lange brauchte der Appl. Compiler ?
#
options -o appltest -check all -show all -times
```

Falls alle benötigten Source-Files (und nur diese) in einem Directory liegen und dieselben Optionen haben sollen, geht es noch einfacher:

```
#
# alle Files im selben Directory wie dieses Buildfile, alle mit -O2 und -db
#
. -O2 -db
#
# ein File mit O3
#
schnell.c -O3

link C -db

options -o ctest -check all -show all -times
```

Mit diesen Beispielen versehen, sollte die Anwendung des Application-Compilers keine Probleme machen. Weitergehende Tips und Tricks, auch zum neuen Debugger CXdb und zum "Performance Analyzer" CXpa, gibt es in der Veranstaltung "Programmentwicklung auf der Convex", die im Sommersemester Dienstags, 14 - 16 Uhr, stattfinden wird.

Peter Junglas