Hilfsmittel zur Programmierung am Beispiel von C++

Frederik Stork

Technische Universität Berlin

Januar 12, 1999

Überblick

1. Unerläßliche Zusatzprogramme
2. Experimente
3. Sonstiges

Ein Beispiel

```
// main.cc
#include<iostream.h>

int main()
{
  // MyAlgorithm();

  cout << "Program has terminated." << endl;

  return(0);
}
```

- möglich: `g++ main.cc`
  besser: `g++ -Wall main.cc -o myprogram`

- Ein modernerer Compiler:
  `/usr/local/egcs/bin/g++`

Infos zu Compileroptionen: `man g++`

Erweitertes Beispiel

```
// myalgo.h
class MyClass
{
  // data

  public:

  int MyAlgorithm();
};
```

```
// myalgo.cc
#include"myalgo.h"

int MyClass::MyAlgorithm()
{
  const int i=5;

  return(i*(i+1)/2);
}
```

```
// main.cc
#include<iostream.h>
#include"myalgo.h"

int main()
{
  int i;
  MyClass d;

  i = d.MyAlgorithm();

  cout << "Result: " << i << endl
       << "Program has terminated." << endl;

  return(0);
}
```

Makefiles

- In einem großen Programm dient ein Makefile der automatischen Erkennung von Komponenten, die neu kompiliert werden müssen.

- Das Kommando `make` sucht im aktuellen Verzeichnis eine Datei mit dem Namen `Makefile` oder `makefile` und führt es aus.

```
# makefile

CXXFLAGS = -Wall
CXX      = g++

CCFILES = \
        myalgo.cc\
        main.cc\

OFILES =  $(CCFILES:%.cc=%.o)

myprogram: $(OFILES)
        $(CXX) -o $@ $(OFILES)
```

Infos zu make: vom xemacs aus Ctrl-h Crtl-i

Fehlersuche im Programm (Debugging)

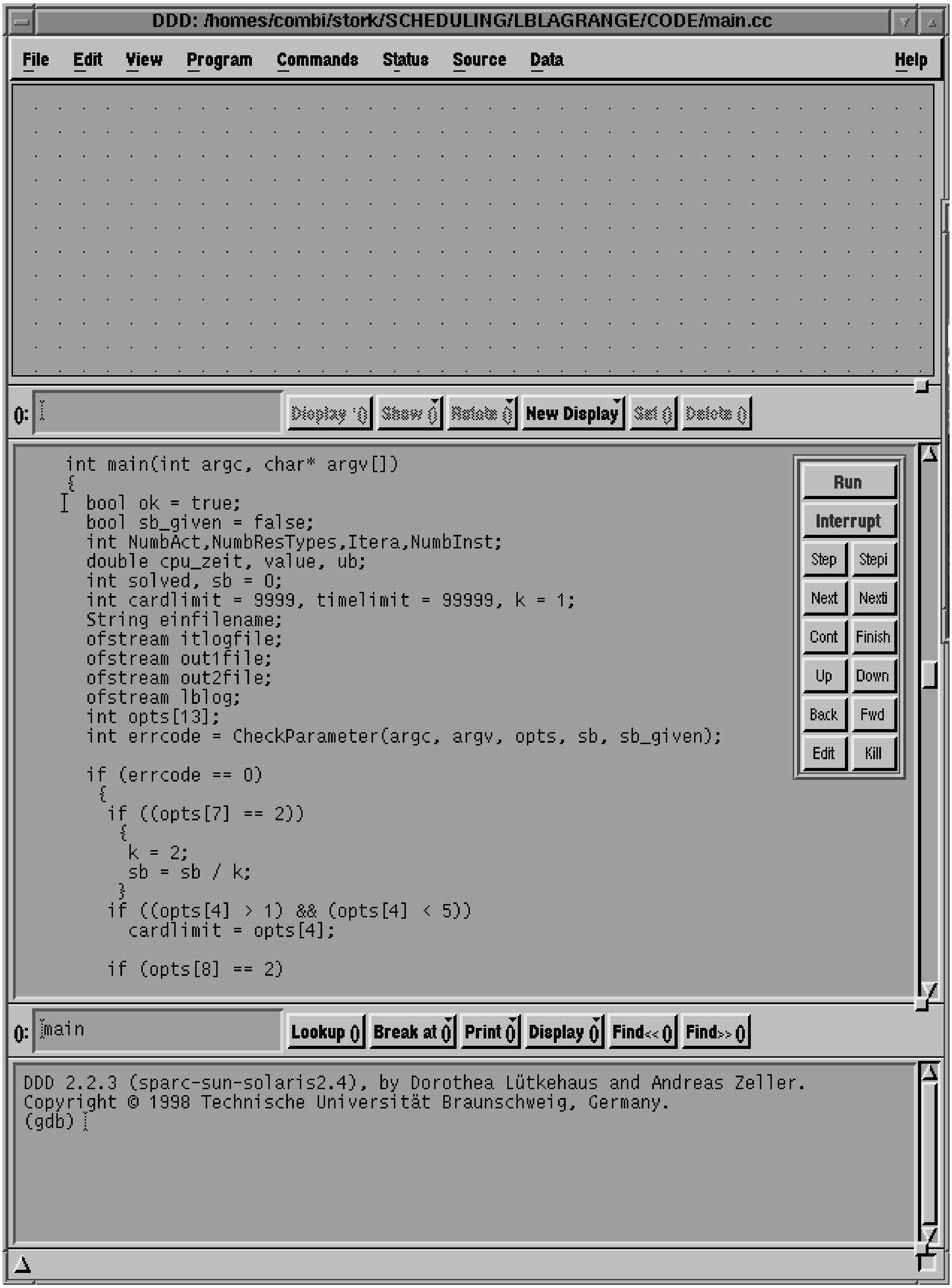
- Programmabsturz:
  Meldung `Segmentation fault` oder `Bus error`.

- Compileroption `-g` (Integriert Debuginformationen)

- Eine einfache und interaktive Oberfläche zum Debuggen: `ddd`

- Aufdecken von unerlaubten Speicherzugriffen: `purify`

Infos zu ddd: http://www.cs.tu-bs.de/softech/ddd/
Infos zu purify: Handbuch


DDD: /homes/combi/stork/SCHEDULING/LBLAGRANGE/CODE/main.cc
File
Edit
View
Program
Commands
Status
Source
Data
Help
0:
Display ()
Show ()
Rotate ()
New Display
Set ()
Delete ()
int main(int argc, char* argv[])
{
bool ok = true;
bool sb_given = false;
int NumbAct,NumbResTypes,Itera,NumbInst;
double cpu_zeit, value, ub;
int solved, sb = 0;
int cardlimit = 9999, timelimit = 99999, k = 1;
String einfilename;
ofstream itlogfile;
ofstream out1file;
ofstream out2file;
ofstream lblog;
int opts[13];
int errcode = CheckParameter(argc, argv, opts, sb, sb_given);
if (errcode == 0)
{
if ((opts[7] == 2))
{
k = 2;
sb = sb / k;
}
if ((opts[4] > 1) && (opts[4] < 5))
cardlimit = opts[4];
if (opts[8] == 2)
Run
Interrupt
Step
Stepi
Next
Nexti
Cont
Finish
Up
Down
Back
Fwd
Edit
Kill
0: main
Lookup ()
Break at ()
Print ()
Display ()
Find<< ()
Find>> ()
DDD 2.2.3 (sparc-sun-solaris2.4), by Dorothea Lütkehaus and Andreas Zeller.
Copyright © 1998 Technische Universität Braunschweig, Germany.
(gdb)

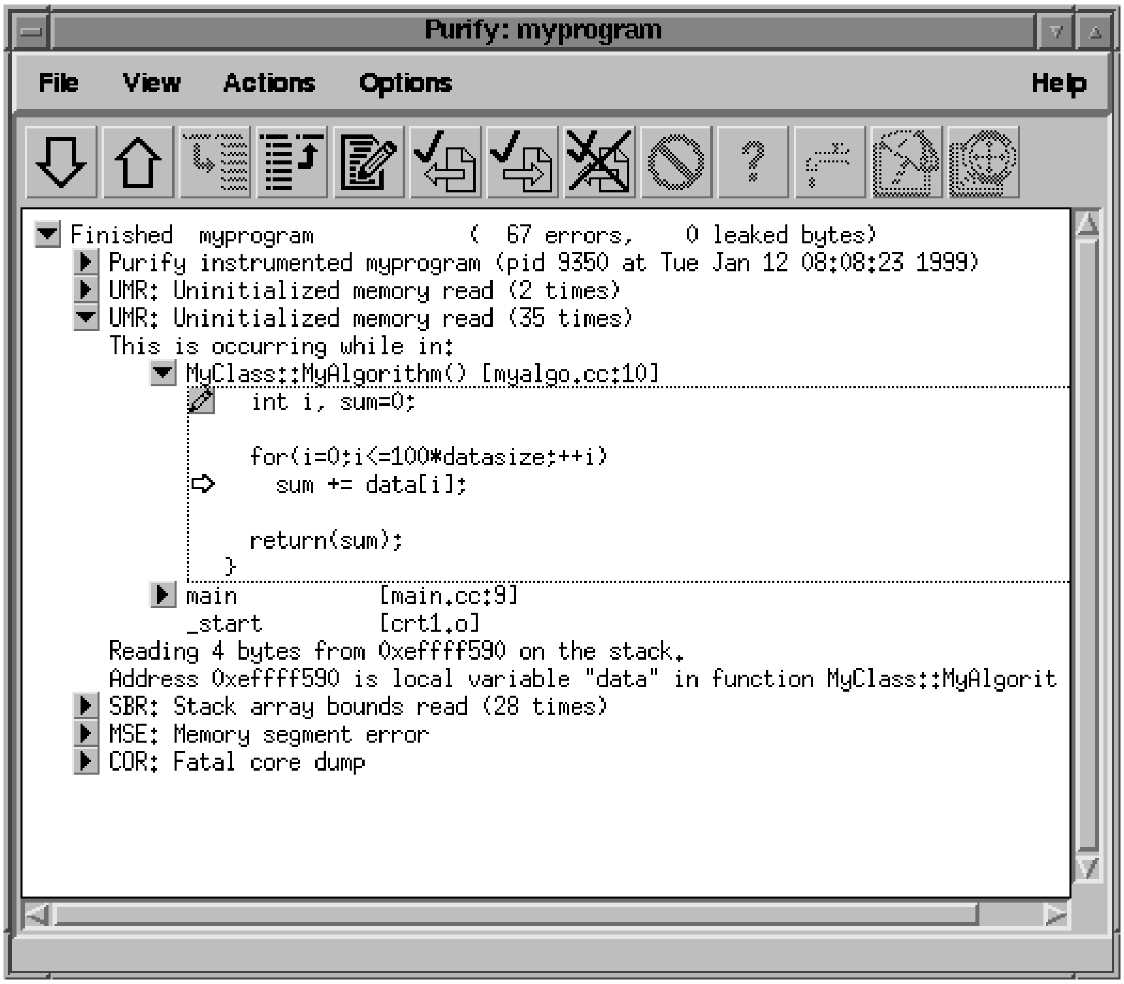
Purify

Programmabstürze werden meist durch Zugriffe auf nichtreservierte Speicherbereiche verursacht.

- Überschreitung von Arraygrenzen
- Operationen mit dem NULL-Pointer

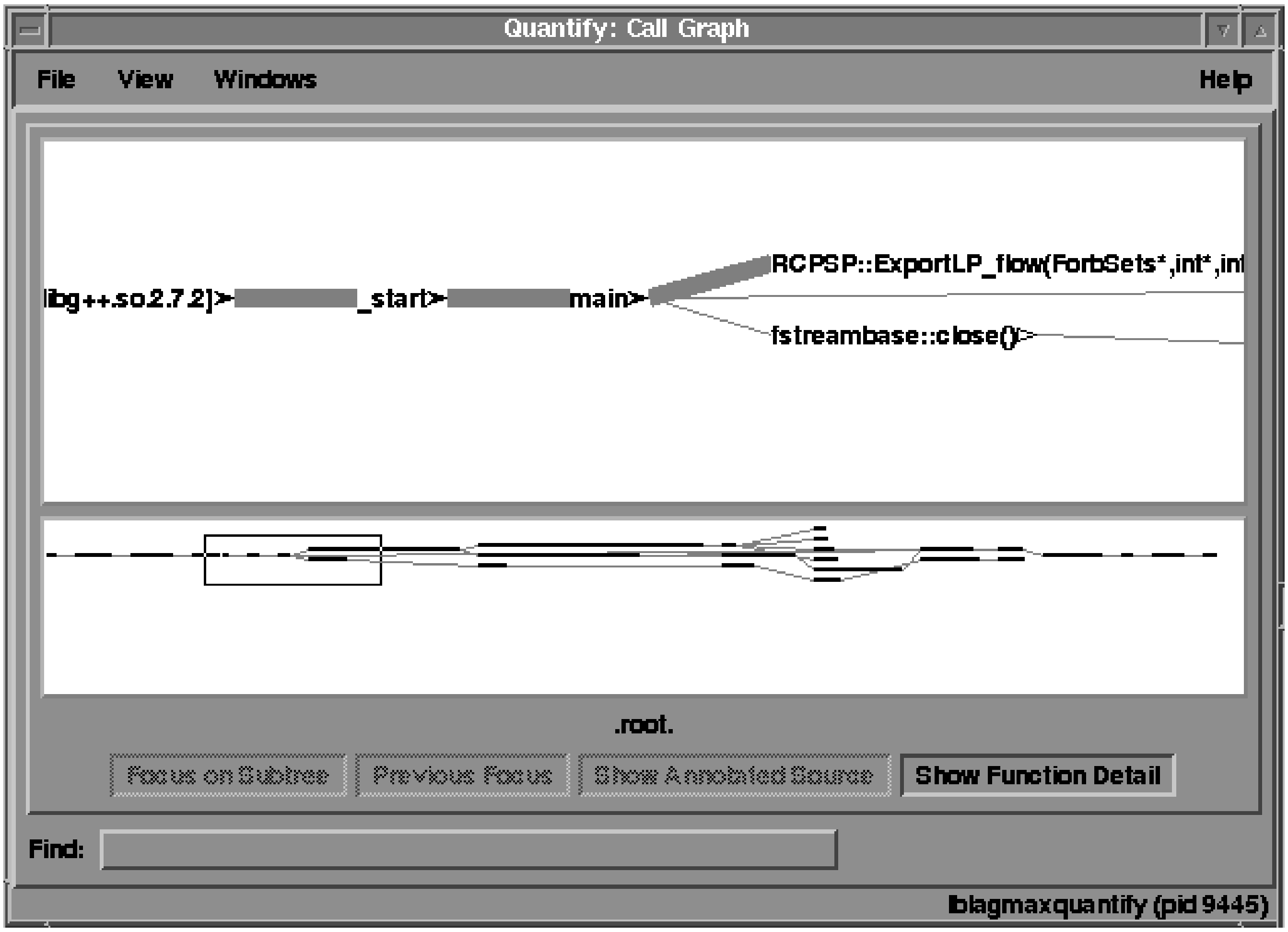
```
const int datasize = 10;
int i, sum=0, data[datasize];

for(i=0;i<=100*datasize;++i)
  sum += data[i];
```



Codeoptimierung

- Compileroptimierung: `g++ -O4 ...`
- Hardwareinformation: `g++ -msupersparc ...`
- Lokalisierung von Bottlenecks im Code durch Profiler: `Quantify`



Infos zu quantify: Handbuch

Dokumentation

- Extrem wichtig! Niemals “auf die lange Bank schieben”!

- Integration von Dokumentation in den Code und anschließende Generierung von z.B. HTML-Seiten: `javadoc`, `perceps`, `doc++`.

```
// myalgo.cc
#include"myalgo.h"

/** Computes the sum of the elements 1 to i.
  * @param i Number to which the sum should
  *          be computed
  * @return  0 if the parameter i is negative,
  *          the sum otherwise
  * @author  Wera Uchimmer
  */
int MyClass::MyAlgorithm(int i)
{
  if (i<0) return(0);
  else return(i*(i+1)/2);
}
```

Sicherungskopien

- Bei uns: regelmäßig auf ein Band
- Von Euch aus: regelmäßig auf Diskette oder per Modem nach Hause.
- RCS/CVS

RCS (Revision Control System):

- Es kann jede einmal “eingecheckte” Version einer Datei wiederhergestellt werden.
- Verhindert das gleichzeitige Bearbeiten einer Datei von mehreren Nutzern
- Einzige Voraussetzung für die Benutzung: Ein Verzeichnis mit dem Namen `RCS` im Verzeichnis des Projekts.
- Befehle:
  ```
  ci -u <datei> (xemacs:  Ctrl-x Ctrl-q)
  co -l <datei> (xemacs:  Ctrl-x Ctrl-q)
  ```

Hilfe: http://fgnn.iait.de/n2e4_developper/n3e4_rcs.htm

Parametersteuerung

`myprogram -o -u4711 -l0815 <instance.dat>`

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
  bool opt_o=false;
  int lb=0, ub=0;
  char c;

  while ((c = getopt(argc, argv, "ou:l:")) != -1)
    switch (c)
    {
      case 'o':  opt_o=true; break;
      case 'u':  ub = atoi(optarg); break;
      case 'l':  lb = atoi(optarg); break;
      case 'h':  cerr << "usage: ... " << endl;
    }
  ...
}
```

Bei sehr vielen Parametern ist es sinnvoll, diese über eine Datei einzulesen.

Hilfe: man -s3C getopt

Seid nett!

- Gebt Prozessen, die lange laufen, eine niedrige Priorität.
  `nice +10 myprogram`

- Begrenzt den Speicher und die Laufzeit Eurer Programme.
  `limit`
  `limit cputime = 100` (Sekunden)

```
cputime         1:40
filesize        unlimited
datasize        2097148 kbytes
stacksize       2097148 kbytes
coredumpsize    0 kbytes
vmemoryuse      unlimited
descriptors     64
```

Skripte

- Iterativer Aufruf Eures Programms für viele Instanzen
- Auswertung von Ergebnisdateien
- Programmiersprache: Perl

```
#!/usr/local/bin/perl -w

open (INFILE, "j30opt.sm") || die "Can't open.";
  while(<INFILE>) {
    if (/(\S+)\s+(\d+)\s+(\d+)/) {
      $name=$1;
      $lb=$2;
      $ub=$3;
      system "nice myprogram -l$lb -u$ub $name";
    }
}
```

```
test1.sm 1 2
test2.sm 100 200
test3.sm 32 12
```

Buch: Learning Perl, R. Schwartz, T. Christiansen, L. Wall

CPLEX

Häufig transformiert man ein gegebenes Problem in ein Lineares Programm, welches dann mit Standardsoftware der Linearen Optimierung gelöst werden kann $\Rightarrow$ CPLEX.

```
\test.lp
Minimize
  obj: x1 + x2
Subject To
 c1: 2 x1 + x2 >= 5
 c2: x1 + 2 x2 >= 5
Bounds
  0 <= x1 <= 10
  0 <= x2 <= 8
End
```

1. Rechner: bond, owl, wandlitz
2. Aufruf: `cplex`
3. `read test.lp`
4. `opt`
5. Weitere Infos: `help`
6. Beenden: `quit`

Der Aufruf von CPLEX ist auch direkt aus C/C++ Programmen möglich!

Handbuch: unter anderem im MA608 und im MA708

Standard Template Library (STL)

- Es ist meist sinnlos, Standard-Datenstrukturen wie Listen, Priority Queues, etc. selbst zu implementieren.
- Die STL stellt derartige Datenstrukturen zur Verfügung.
- Die STL gehört zum C++ Standard.

> - Buch: STL Tutorial and Reference Guide, D. Musser and A. Saini
> - http://www.sgi.com/Technology/STL/

Verzeichnisstruktur

```
myproject/
      Makefile
      RCS/
      bin/
          myprogram
      doc/
          overview.html
          myclass.html
      include/
          RCS/
          myalgo.h
      obj/
          myalgo.o
          main.o
      results/
      src/
          RCS/
          myalgo.cc
          main.cc
      testinstances/
```