Parallel Debugging

Matthias Müller, Pavel Neytchev

University of Stuttgart
High-Performance Computing-Center Stuttgart (HLRS)
www.hlrs.de

Outline

• Motivation
• Approaches and Tools
• Totalview
Problems of Parallel Programming

- All problems of serial programming
- Additional problems:
  - Increased difficulty to verify correctness of program
  - Increased difficulty to debug N parallel processes
  - New parallel problems:
    - deadlocks
    - race conditions
    - irreproducibility

What is a debugger?

- Common Misconception: A debugger is a tool to find and remove bugs
- A debugger does:
  - tell you where the program crashed
  - help to gain a better understanding of the program and what is going on
- Consequence:
  - A debugger does not help much if your program does not crash, e.g. just gives wrong results
  - Avoid using a debugger as far as possible.
  - Use it as last resort.
Avoiding debuggers

- Think about a verbose execution mode of your program
- Use a careful/paranoid programming style
  - check invariants and pre-requisites
    (assert(m>=0), assert(v<c) )
- Use the debugging/assertion techniques of the compiler
  - use debug flags (-g), warnings (-Wall)
  - array bound checks in Fortran
  - use memory debug libraries (-lelfence)

Avoiding Debuggers

- Write portable programs
  - it avoids future problems
    - architectures/platforms have a short life
    - all compilers and libraries have bugs
    - all languages and standards include implementation defined behavior
  - running on different platforms and architectures significantly increases the reliability
- Use verification tools for parallel programming like assure
Parallel Debuggers

- Most vendor debuggers have some support
- gdb has basic support for threads
- Debugging MPI programs with a “scalar” debugger is hard but possible
  - mpich supports debugging with gdb attached to one process
  - manual attaching to the processes is possible
- totalview is a good but expensive tool

TOTALVIEW

Matthias Müller

University of Stuttgart
High-Performance Computing-Center Stuttgart (HLRS)
www.hlrs.de
What is TotalView?

- Parallel debugger
- Source level debugging for C, C++, F77, F90, HPF
- MPI, OpenMP, Pthreads, PVM, shmem
- SMPs, MPPs, PVPs, Clusters
- Available on all major Unix Platforms and most Supercomputers
- GUI (independent of platform, exception Cray T3E)
  - totalview 4.x based on tcl/tk
  - totalview 5.x based on Motif

Availability of TotalView

- Compaq Digital Alpha
  - HP-UX
  - IBM RS6000 and SP Power
  - SGI MIPS
  - Sun SPARC SunOS 5
  - Linux Intel IA32 (RedHat)
  - Linux Alpha (RedHat)
- Cray T3E by Cray
- Hitachi SR2201 by SofTek, SR8000
- NEC SX-4 (V 3.x) by SofTek, SX-5 in beta-test
## Availability of TotalView at HWW

<table>
<thead>
<tr>
<th>Platform</th>
<th>Availability</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volvox</td>
<td>Yes</td>
<td>V5 and V4</td>
</tr>
<tr>
<td>Hitachi SR2201</td>
<td>Yes</td>
<td>V3.7.9</td>
</tr>
<tr>
<td>Hitachi SR8000</td>
<td>Yes</td>
<td>V4.0</td>
</tr>
<tr>
<td>Cray T3E</td>
<td>Yes</td>
<td>Cray 3.0.0</td>
</tr>
<tr>
<td>NEC SX-4</td>
<td>Yes</td>
<td>V3.7.9</td>
</tr>
<tr>
<td>NEC SX-5</td>
<td>Yes</td>
<td>Betatest</td>
</tr>
<tr>
<td>SGI Onyx</td>
<td>No</td>
<td>Use cvd</td>
</tr>
<tr>
<td>HP N-Class</td>
<td>Yes</td>
<td>V4.1 and V5</td>
</tr>
<tr>
<td>IBM SP</td>
<td>Yes</td>
<td>V4.1</td>
</tr>
<tr>
<td>Cray SV1</td>
<td>Yes</td>
<td>Cray 3.0.0</td>
</tr>
</tbody>
</table>

More information:

www.hlrs.de/organization/par/services/tools/debugger/totalview

## Availability of TotalView at University Stuttgart

2 user 8 CPU Floating License for University Stuttgart:

1. Download Software from http://www.etnus.com
2. Set environment variable for license.
   
   LM_LICENSE_FILE=7244@servint1.rus.uni-stuttgart.de

More information about campus licenses available at

http://www.hlrs.de/organization/par/services/tools/campus
TotalView usage at HLRS

- set USE_TOTALVIEW in your login scripts
- CRAY T3E: set USE_PROG_ENV
- Compile with -g compiler switch
  CRAY T3E: compiler switch -G
- command name: totalview

Starting TotalView

On a new process:

% totalview myprog -a arguments to myprog

To debug MPI programs:

% totalview mpirun -a -nprocs 3 myprog
% mpirun -tv -np 3 myprog

To debug IBM POE programs:

% totalview poe -a myprog [args]

To debug CRAY T3E programs:

% totalview -X #procs myprog [args]
Totalview on Hitachi SR8000

- Compilation:
  - f90 -g
  - cc -g
  - KCC -g --backend -tv
- OpenMP
  - f90 -g -omp -procmnum=8
  - cc -g -omp -parallel=1 -O2
- MPI
  - mpirun -tv

Totalview on HPN

- Compilation:
  - f90 -g
  - cc -g
  - KCC -g
- OpenMP
  - guif90 -g
  - guidec -g
  - guidec++ -g
- MPI
  - mpirun -np #procs -tv ./a.out
Totalview Exercise: Basic Look & Feel

- Log into hwwhpn.hww.de
- use bash as shell
- Change into directory
  ~/TOTALVIEW/#NR/TOTALVIEW/SIMPLE
- Compile calc_pi_{f90,c,cc}.f90,cc
- Start totalview with totalview executable

TotalView Windows

Root Window

Process Window

Data Windows
Totalview Mouse Buttons

- **Left button is Select:**
  - Chooses an item of interest,
  - Starts editing an item

- **Middle button is Menu:**
  - Raises a menu of actions
  - All menus have a Help (^?) entry

- **Right button is Dive:**
  - Gets more information about an item
  - Shift+Dive forces open a new window

Totalview Main Window

- **Process/thread status**
- **Process ID**
- **Number of threads**
- **Process name**
- **Thread list**
- **tid/systid**
- **Function or PC value**
TotalView Process Window

- Stack Trace pane
- Source pane
- Thread pane
- Action Points pane
- Local variables for the selected frame
- Process/thread motion buttons

TotalView Source Pane

- Gridded box is a possible site for a breakpoint
- Current function and source file
  - Select to set one
  - Current point of execution
    - Dive on a source word to get more information
    - Select a line to use Run to selection command
    - Select or dive on a line number to set an action point
Parallel Debugging - Philosophy

- By default, TotalView places processes in groups
  - Program Group - Includes parent and all related processes
  - Share Group - Only processes that share the same source code
- Command can act on single process or share group
  - halt process (h), halt group (H)
  - next step process (n), next step group (N)
  - go process (g), go group (G)

Totalview Exercise: Debug simple program

- Run calc_pi inside totalview:
  - Check where the program crashes
- analyze core file with totalview
  - run calc_pi
  - execute totalview calc_pi core
- for advanced users: choose another programming paradigm:
  - MPI, OpenMP, MPI+OpenMP
TotalView support for debugging MPI

- Special support for MPI is available depending on your MPI library:
  - display message queue state of a process
- Supported MPI implementations:
  - mpich v1.1.0 or later (use -debug in configure)
  - HP MPI v1.6
  - Compaq MPI >v1.7
  - IBM, release >2.2 of Parallel Environment, threaded version of MPI
  - SGI MPI v1.3 or later

TotalView MPI Message Queue Window

- Communicator name and info
- Non-blocking receive operations
- Unmatched incoming messages
- Non-blocking send operations
- Dive on source or target to refocus Process window
- Dive on buffer to see message contents
Totalview Exercise: Parallel program

- Example in TOTALVIEW/MPI:
  - deadlock_{c,cc,f90}.{c,cc,f90}
  - start program with mpirun -tv -np 2 a.out
  - interrupt execution after “deadlock”
  - try to find the reason for the deadlock and fix it

- For advanced users:
  - pending_{c,cc,f90}.{c,cc,f90}
  - try to find pending message by setting breakpoint at MPI_Finalize

TotalView more information

- http://www.hlrs.de/organization/par/services/tools/debugger/totalview
  - User Guide
  - Installation Guide
  - CLI Guide
  - Powerpoint Tutorial
- CRAY T3E: Online Documentation at http://www.hlrs.de/platforms/crayt3e