Debugging of thread-based task-parallel Programs in StarSS

5th Parallel Tools Workshop, 27th of September 2011
Rainer Keller, HLRS
Steffen Brinkmann, HLRS
Jose Gracia, HLRS
Christoph Niethammer, HLRS
Overview

• Overview of Task-Based Parallel Programming Models

• The StarSs programming model

• The Temanejo Debugger

• Demo

• Conclusion
Overview of Task-based Parallelism

• Parallel Programming has come a long way (P4, PVM, MPI)

• Fine-grained parallelism for upcoming many-core nodes:
  – Pthreads (DIY),
  – OpenMP (for loops),
  – Task-based parallelisation (OpenMP, Cilk, HMPP, StarSS)

•Pragma-based compiler extensions have nice features:
  – Portability of code onto various machines / compilers,
  – On top of existing language (OpenMP, HPF)
  – Assumption of incremental parallelization (OpenMP),
  – i.e. parallel version is “similar” to existing sequential code.
TEXT Project 1/3

• Towards EXascale Applications (TEXT)
• EU-funded CP & CSA in FP7-Infrastructures-2010-2
• 9 Partners:
  4 HPC Centers, 4 Universities, 1 Industrial

• Centered around StarSS programming model by BSC:
  #pragma css task input(v1, v2, len) output(v3)
  void vadd (float *v1, float *v2, float *v3, int len)
  { ... }

• Project Aim: To apply the StarSs at various HPC Centers, various real-world applications and extend and optimize the compiler in the process.
• Parallelization using SmpSs on the:
  – BEST / LBC Lattice Boltzmann codes
    Jose Gracia
  – LS1-Mardyn MD code
    Christoph Niethammer

• Develop a small testsuite to find implementation bugs and tools capabilities (valgrind-like errors)

• Develop debugger & techniques of threaded debugging
• Ease thread programming with graphical debugger:

Contact: Steffen Brinkmann
Task-parallel programming

• Task-Parallel programming similar in several aspects:
  – Allow expression of dependencies for parts of code,
  – Provide means to synchronize,
  – Possibly schedule tasks

• StarSs: Pragma-based model on top of C/Fortran
• Framework from BSC: Cell-SuperScalar CSS, SmpSs

```c
#pragma css task input(A, B) inout(C)
void matmul (float A[64][64], float B[64][64], float C[64][64])
{
    for (int i=0; i < 64; i++)
        for (int j=0; j < 64; j++)
            for (int k=0; k < 64; k++)
                C[i][j] += A[i][k] * B[k][j]
}
...
for (int bi = 0; bi < SIZE; bi++) matmul (A[bi * SIZE][0]);
```
Task-parallel programming

• These tasks are
  – Generated (StarSs: on the fly during runtime),
  – Scheduled onto Unix Threads and
  – Executed on a core.

• The Task-Graph
  – is a huge DAG,
  – is created on-the-fly,
  – and multi-threaded execution poses questions to programmers

• We need a debugger supporting task-parallel execution
Task-parallel programming

- Traditional debuggers by design are:
  - Line-based,
  - focus on one thread,
  - focus on serial correctness,
  - Code-generators obfuscate the code

- So, in conjunction with StarSs:
  - Want to Visualize large DAGs,
  - Step through execution,
  - See data dependencies
  - Debug the actual CSS tasklets using traditional debugger.
Why use Temanejo?

- Debugging means **seeing** and intercepting the execution:
  - **Tasks:**
    - Status of Tasks
    - Which Function
    - On which Thread/Core
  - **Dependencies:**
    - Which memory address
    - Which Tasks inter-depend
Why use Temanejo?

• Debugging means seeing and **intercepting** the execution:

• **Tasks:**
  – Stop at a specific task
  – Attach debugger (gdb)
  – Force specific execution & Prioritize specific tasks

• **Dependencies:**
  – Resolve Dependency
  – Through debugger intercept actual parameters
The aim is to allow:
- MPI + StarSs debugging,
- Scalable display,
- Different task-parallel models:
Semantics of Temanejo display

• Temanejo is very flexible, one may set the semantics of
  – Colors of node (default: executing tasklet / function),
  – Shape of node (default: executing thread)
  – Border’s color (default: Task’s state)
• Task’s state during it’s lifetime as follows:

```
Not Queued

Queued

Running

Finished
```

Dependencies fulfilled?

Dequeued by worker thread?

When finished?

Notify dependent tasks!
Temanjo Demo

:: Debugging w/ Temanejo ::

::

14
Conclusion

- Ayudame is a library to hook into task-parallel runtimes (currently: SmpSs, and OMPS compiler of BSC)
- Temanejo is an extensible, scalable debugger for StarSs and other task-based programming models.

Thank You very much!

Acknowledgements:
- Rosa Badia, Jesus Labarta, BSC