NUMA-aware OpenMP Programming

Dirk Schmidl
IT Center, RWTH Aachen University
Member of the HPC Group
schmidl@itc.rwth-aachen.de

Christian Terboven
IT Center, RWTH Aachen University
Deputy lead of the HPC Group
terboven@itc.rwth-aachen.de

Future OpenMP Directions

Future OpenMP Directions

- **Recent Technical Reports**
  - → TR3 (November 2014) provides current snapshot of OpenMP 4.1

- **OpenMP 4.1**
  - → Clarifications and errata to existing specification
  - → Refinements and minor extensions
  - → Do not break existing code
  - → Minimal implementation burden beyond 4.0
  - → Will be released by SC15

- **OpenMP 5.0**
  - → Address several major open issues for OpenMP
    - → Expect less significant advance than 4.0 from 3.1/3.0
  - → Do not break existing code unnecessarily
  - → Targeting release for SC17 (somewhat ambitious)
  - → Plan to release intermediate TR(s) annually

Device Constructs

- **Unstructured data movement**
  - → data regions need to start and end in the some scope
    - →complicates code structuring sometimes
    - →cannot be used in constructor and destructor
  - → now, mapping data to the device:

```
#pragma omp target enter data [clause [[,] clause] …]
```

  - → and back to the host:

```
#pragma omp target exit data [clause [[,] clause] …]
```

- **Device specific environment variables**
  - → specify OMP_NUM_THREADS,… for the Device
- **Multiple device types**

Device Constructs

- **Improved asynchronous execution**
  - → Currently nearly impossible to continue working on the host
  - → in 4.1 `target` regions become tasks
  - → can use `nowait` and `depend` clauses
- **Deep copy/map**
- **Interoperability with low level libraries**

Taskloops

- The `taskloop` directive allows to execute loops as tasks

```
#pragma omp taskloop [clause [[,] clause] …]
```

  - → Allows to execute loops with a subset of the threads in the team
  - → Provides better load balancing in some cases
  - → can be nested with other tasks and worksharing
- The task sizes can be optimized by the following clauses
  - → The `grainsize` clause specifies the number of iterations per task
  - → The `num_tasks` clause specifies an exact number of tasks to generate
- Most clauses of loops and tasks can be applied

Other topics

- **Parallelism without `parallel` regions (task-only threads)**
- **Memory affinity**
- **Transactional Memory**
- **General error model**
- **Interoperability e.g. with Pthreads**
- **Task priorities**
- **Task reductions**

Who works on OpenMP

Permanent Members of the ARB:

- AMD
- ARM
- Convey Computer
- Cray
- Fujitsu
- HP
- IBM
- Intel
- NEC
- NVIDIA
- Oracle Corporation
- Red Hat
- ST Microelectronics
- Texas Instruments

Auxiliary Members of the ARB:

- ANL
- ASC/LLNL
- BSC
- cOMPunity
- EPCC
- LANL
- LBNL
- NASA
- ORNL
- RWTH Aachen University
- SNL-Sandia National Lab
- Texas Advanced Computing Center
- University of Houston

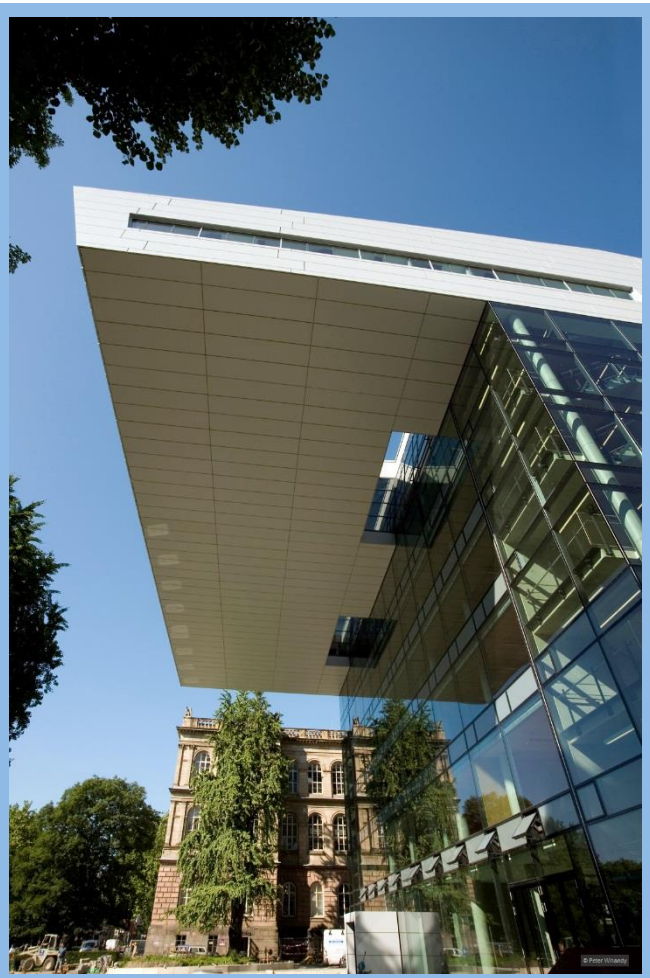
How to contribute?

- **Make your institution member of the OpenMP ARB. Cheap membership options for researchers/research groups in discussion.**
- **Attend IWOMP and become a cOMPunity member**
- **Provide feedback to technical reports**

IWOMP 2015

11th INTERNATIONAL WORKSHOP ON OPENMP
AACHEN, GERMANY

All topics related to OpenMP are of interest, including OpenMP performance analysis and modeling, OpenMP performance and correctness tools, proposed OpenMP extensions, and OpenMP applications in any domain (e.g., scientific and numerical computation, video games, computer graphics, multimedia, information retrieval, optimization, text processing, data mining, finance, signal and image processing and machine learning). Advances in technologies, such as multi-core processors and OpenMP devices (accelerators such as GPGPUs, DSPs or FPGAs), Multiprocessor Systems on a Chip (MPSoCs), and recent developments in OpenMP itself (e.g., tasking) present new opportunities and challenges for software and hardware developers. Recent advances in the C, C++ and Fortran base languages also offer interesting oppportunities and challenges to the OpenMP programming model. IWOMP 2015 particularly solicits submissions in these areas as well as ones that discuss how to apply OpenMP to additional models of parallelism such as event loops.



Important dates

Paper submission: May 3, 2015
Notification of acceptance: June 8, 2015
Camera-ready version: July 13, 2015
Registration deadline: t.b.d.

Q & A?