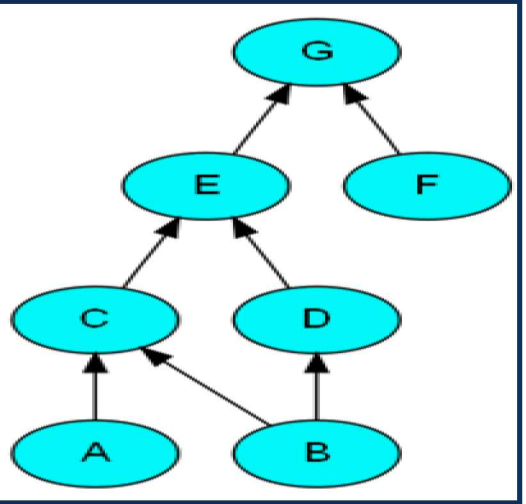
SAND2018-13112C



OpenMP Tasks: New Features in 5.0

SC18 OpenMP BoF

Stephen Olivier
Center for Computing Research
Sandia National Labs (NM)

November 14, 2018

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Some 5.0 Tasking Features

Sandia National Laboratories

- Iterator syntax for clauses and its use in depend clause
  - Expands to multiple values in a range
  - Expected to have more uses in future
- Task affinity based on data locations, similar to dependences
  - Unlike dependences, is a hint and does not impose ordering constraints
  - Can use iterators in this clause also
- The `mutexinoutset` dependence type
  - Enables a set of inout tasks to commute but not execute concurrently
- Allow the depend clause on the `taskwait` construct
- Allow any l-value in the depend clause

More 5.0 Tasking Features

Sandia National Laboratories

- Depend objects
  - Portable representation of a task dependence
  - New construct to initialize, update, and destroy
  - Can then be supplied to the depend clause
- Detached tasks
  - Decouples completion of a task from completion of its structured block
  - Creates an event handle that can be passed in function calls
  - Calling `omp_fulfill_event` routine on a handle completes the task
  - Enables asynchronous interoperation with other programming models like CUDA and MPI

Reductions and Scans

Sandia National Laboratories

- Reductions over tasks
  - Scoped using `task_reduction` clause on `taskgroup` construct or `reduction` clause on `taskloop` construct
  - Can also specify a `task` modifier in the `reduction` clause on a `parallel` or worksharing region to reduce over tasks in the region
  - Specify `in_reduction` clause on any `task`, `target` or `taskloop` construct to participate in the reduction
- Bonus: Parallel prefix scan (not a tasking feature)
  - `scan` directive in a loop / loop nest
  - `inscan` modifier on the reduction clause
  - Can specify inclusive or exclusive

Example: Target Task in Reduction

Sandia National Laboratories

```
int x = 0;

#pragma omp taskgroup task_reduction(+:x)
{
   #pragma omp target in_reduction(+:x) nowait   // offload
      ...
   #pragma omp target in_reduction(+:x) nowait   // offload
      ...
   #pragma omp task in_reduction(+:x)            // exec. on host
      ...
}
// combined value of x available at this point
```