SAND2021-4706C

kokkos

Kokkos Resilience

EXASCALE COMPUTING PROJECT

Nicolas Morales, Keita Teranishi, Bogdan Nicolae, Christian Trott, Franck Cappello

Sandia National Laboratories

Argonne NATIONAL LABORATORY

OAK RIDGE National Laboratory

BERKELEY LAB Lawrence Berkeley National Laboratory

Los Alamos NATIONAL LABORATORY EST. 1943

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineer- ing 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. SAND NO. 2021-XXXX

NNSA National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

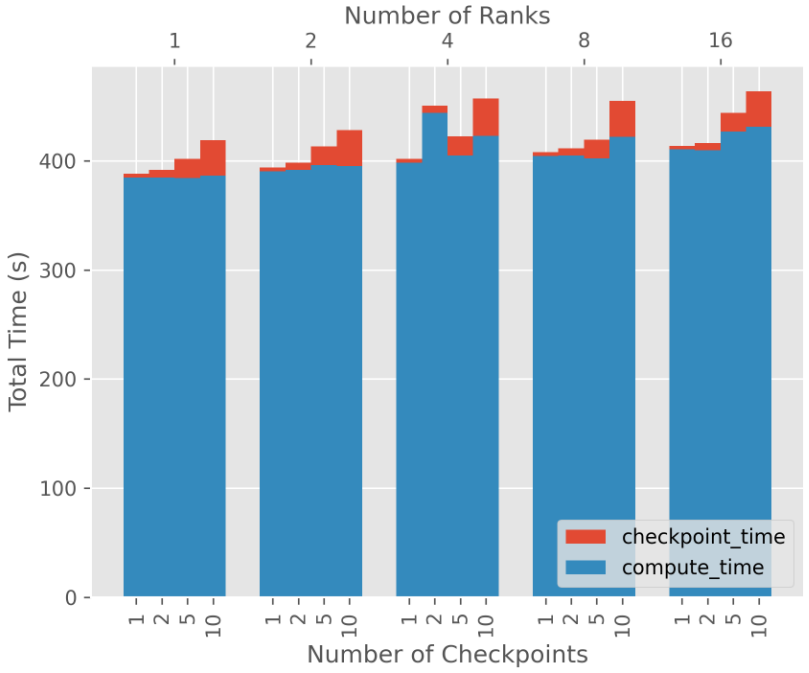
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Resilience in the Exascale Era

- As we enter the Exascale era, mean time between failures (MTBF) increase
  - Soft/hard failures
  - Fail slow
  - Errors may be a result of hardware problems, application bugs, system errors, etc.
- Can we take advantage of “Kokkos-ization” of programs to add in resilience capabilities?
  - Allow Kokkos::Views to easily be checkpointed
  - Make parallel constructs resilient

Checkpointing

- Goal: automatically checkpoint Kokkos::Views with little/no boilerplateTrack usage of Kokkos::View inside kernels
  - Compile time with limited run-time component: negligible performance impact (0.6s overhead with 256 views)
  - Automatically register views that are used inside of the functor passed to KokkosResilience::checkpoint
    - No explicit calls needed to bind views to checkpoint storage
    - Automatic detection of recovery conditions
  - Handle deep copy from device to host automatically
  - Minimize checkpoint by analyzing usage (e.g. const views)
  - Can group multiple kernels into one checkpoint region
- Efficient configurable backend for checkpointing
  - VeloC backend for efficient asynchronous checkpointing
    - Need to ensure that VeloC environment is set up (server process)
  - C++ IO for dependency-less checkpointing



Example timing on MiniMD application over 1000 timesteps

Adding Automatic Checkpointing

Without Checkpointing

```
1  int i = 0;
2
3  while(i < ITER_TIMES) {
4    le = doWork(np, rank, M, nbLines, g, h);
5    if ((i % REDUCED) == 0)
6      MPI_Allreduce(&le, &ge, 1, MPI_DOUBLE, MPI_MAX,
       ↪  MPI_COMM_WORLD);
7    if (ge < PRECISION)
8      break;
9    i++;
10 }
```

With KokkosResilience::checkpoint

```
1  auto i = 1 + KokkosResilience::latest_version( *ctx,
   ↪  "heatdis" );
2
3  while(i < ITER_TIMES) {
4    KokkosResilience::checkpoint(ctx, "heatdis", i,
5      [=, &le, &ge]() {
6        le = doWork(np, rank, M, nbLines, g, h);
7        if ((i % REDUCED) == 0)
8          MPI_Allreduce(&le, &ge, 1, MPI_DOUBLE, MPI_MAX,
           ↪  MPI_COMM_WORLD);
9    } );
10   if (ge < PRECISION)
11     break;
12   i++;
13 }
```

Resilient Execution Spaces

```
1  using view_type = Kokkos::View<double*, KokkosResilience::ResCudaSpace>;
2
3  view_type data("data", N);
4  Kokkos::RangePolicy<KokkosResilience::ResCuda> rp(0, N);
5
6  Kokkos::parallel_for(rp, KOKKOS_LAMBDA(const int i) {
7      data(i) = i;
8  });
```

- Execution and memory space for resilient kernel execution
- Mitigate soft errors in kernels by triplicating kernel execution
- Three step execution
  - Triplicate view data
  - Concurrent kernel execution for each copy
  - Recombine via voting; abort if all three differ

Conclusion

- Ongoing effort to add resilience to Kokkos programs
  - Goal is to mitigate effects of decreased MTBF as we move to exascale
- Capabilities
  - Automatic checkpoint/restart of views with minimal boilerplate
  - Resilient kernel execution
- One of our primary goals is ease of use and simplicity of adding resilient capabilities to existing Kokkos programs
- Please contact me (nmmoral@sandia.gov) or Keita Teranishi (knteran@sandia.gov) for questions