Kokkos-HClib: enabling high-performance and resiliency for HPC systems

Akihiro Hayashi, Sri Raj Paul, Matthew Whitlock (Georgia Tech)
Nicolas Morales, Jeff Miles, Keita Teranishi (Sandia National Laboratories)
Vivek Sarkar (Georgia Tech)

Sandia report SAND2020-8692 C Unclassified Unlimited Release

Kokkos

- What's Kokkos?
  - Kokkos enables single performance portable codes
    - ✓ CPUs, NVIDIA GPUs, …
  - Kokkos provides data abstractions critical for performance portability not available in OpenMP or OpenACC (Kokkos::View)
  - Essential parallel constructs
    - ✓ Kokkos::parallel_for();
    - ✓ Kokkos::parallel_reduce();
    - ✓ Kokkos::parallel_scan();

Georgia Tech Sandia National Laboratories

Kokkos Example

```
1  View<int*> x("x", N);                              // Constuct a View (length: N)
2  // Parallel For
3  parallel_for(RangePolicy<>(0, N),                  // 1D Range (0 to N)
4               [=] (int i) {                         // C++11 lambda
5                 x(i) = i;                           // Computational Body
6               });
7
8  int result = 0;
9  // Parallel Reduce
10 parallel_reduce(RangePolicy<>(0, N),               // 1D Range (0 to N)
11                 [=] (int i, int &update) {         // C++11 lambda
12                   update += x(i);                  // Computational Body
13                 }, result);                        // Value to update
14 std::cout << result << std::endl;
```

Georgia Tech

Sandia National Laboratories

Enabling Resiliency in Kokkos

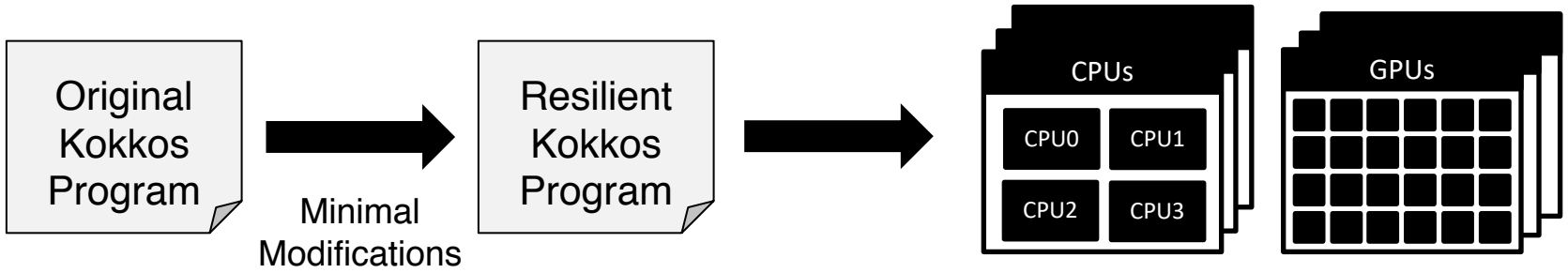
- Motivation
  - Failure probability “significantly” increases as the number of nodes increases [1]
- Observation
  - User-level failure mitigation mechanisms will become more important in the next generation systems
    - ✓ Checkpoint/Restart (C/R)
    - ✓ Task Replay
    - ✓ Task Replication
    - ✓ Algorithm-based Fault Tolerance (ABFT)

Georgia Tech Sandia National Laboratories

[1] C. D. Martino et al. 2015. Measuring and Understanding Extreme-Scale Application Resilience: A Field Study of 5,000,000 HPC Application Runs. (DSN2015)

Enabling Resiliency in Kokkos

- Research Question
  - When enabling resiliency in Kokkos, what is a desirable point in the performance-portability-productivity (P3) space?
  - What is a desirable user-facing parallel-resilience API that enables good productivity-portability?

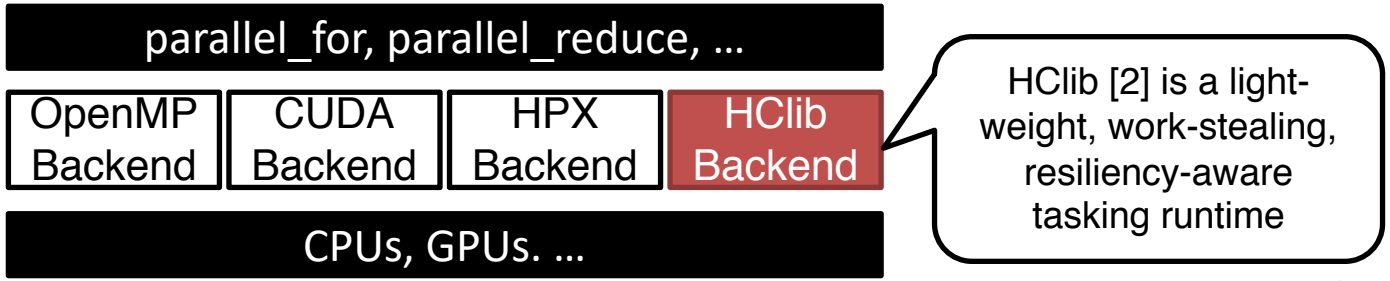


Resilient Execution on different platforms

Georgia Tech
Sandia National Laboratories

Resilient Kokkos

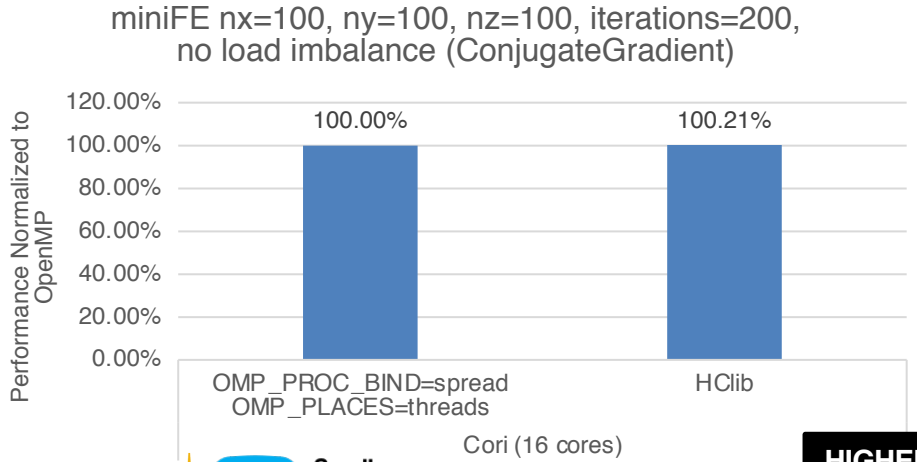
- Overview
  - Provides Kokkos with resilient execution capability
  - Build on top of Resilient HClib [2]
    - ✓ Provides various runtime APIs for task-level resiliency
- Our Current Focus
  - Implementation of Kokkos-HClib Backend
  - Design and implementation of Kokkos Task Replay/Replication API

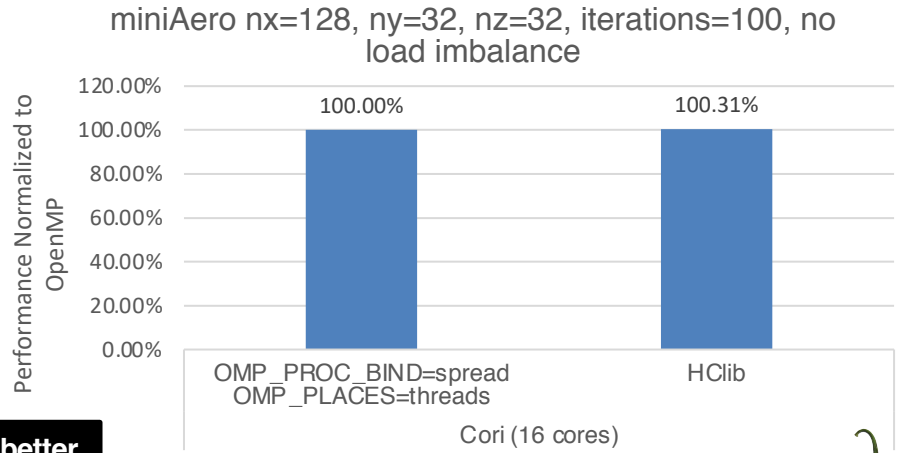


[2] Sri Raj Paul et al. 2019. Enabling Resilience in Asynchronous Many-Task Programming Models. (EuroPar2019)

Building Kokkos-HClib backend

- Non-resilient Execution of miniFE and miniAero of the Mantevo benchmark on Cori@NERSC
- The HClib backend is comparable to the OpenMP backend





HIGHER is better

CPU: 16-core 2.3 GHz Intel Xeon Processor E5-2698 v3 (Haswell), Compiler: gcc 8.3

Resilient Kokkos API Design

- Key Idea: Provide resilient execution spaces

Original parallel_for

```
parallel_for(RangePolicy<ExecutionSpace>(0, N), lambda);
```

- Task Replication
  - Semantics: Replicate a task, and check for the equality of output variables

Replicate parallel_for

```
parallel_for(RangePolicy<ReplicateSpace>(0, N), lambda);
```

- Task Replay
  - Semantics: Execute a task, then invoke a user-provided error checking function, and replay the task if the error checking is failed

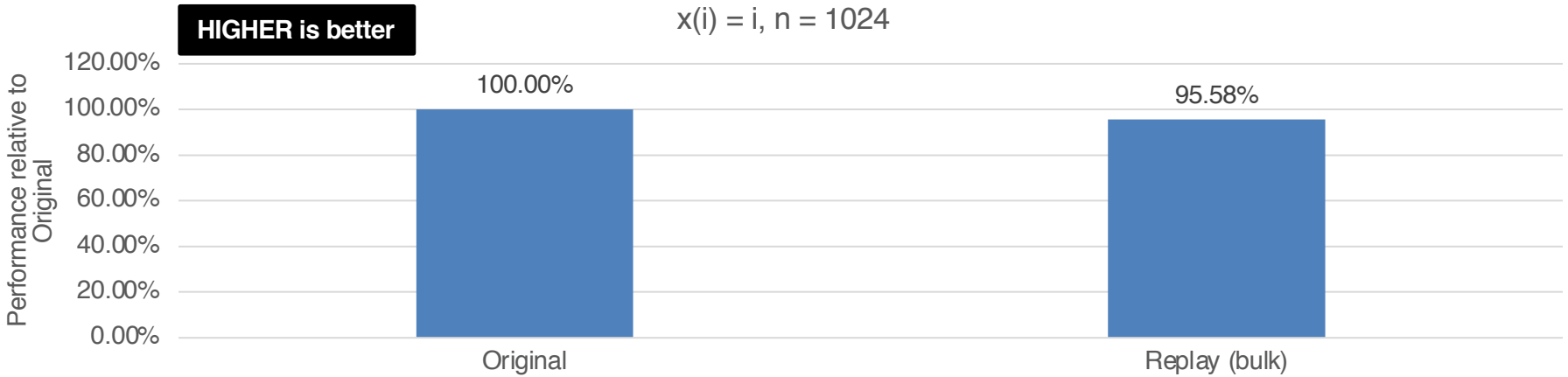
Replay parallel_for

```
parallel_for(RangePolicy<ReplaySpace>(0, N), lambda, lambda_check);
```

Georgia Tech

Sandia National Laboratories

Preliminary Performance Evaluation (Overhead of Error Checking)



- Variants
  - Original: parallel_for only
  - Replay (bulk): parallel_for w/ a single error checking (no replay)

Georgia Tech Sandia National Laboratories

CPU: 4-core 3.5 GHz Intel(R) Core(TM) i5-7600 CPU, Compiler: gcc 7.5

Summary & Future Work

- Summary
  - The Kokkos Resilient API enables resiliency in Kokkos in a portable and productive manner
    - ✓ Essentially, the user is only supposed to 1) replace an execution policy with a resilient version and 2) provides an error checking function if necessary
  - The developed Kokkos-HClib backend is comparable to existing OpenMP backend for non-resilient execution
    - ✓ Verified with two of the Mantevo benchmark (miniFE/miniAero)
- Future Work
  - Continue the implementation and discuss the performance impact of the Resilient API using different applications
  - Support the all the Kokkos constructs in the Kokkos-HClib backend

Georgia Tech

Sandia National Laboratories