



Remote OpenMP Offloading

Atmn Patel (University of Waterloo)
Johannes Doerfert (Argonne National Laboratory)

Motivation

OpenMP remains widely used for intra-node parallelism in HPC.

It is non-trivial to extend OpenMP applications to distributed systems.

It requires the incorporation of additional programming models such as MPI, or the wholesale replacement of OpenMP via X10, Chapel, etc.



Motivation

With the introduction of accelerator offloading, OpenMP provides a natural transition from parallelism on CPUs to GPUs.

In comparison to vendor-specific languages such as CUDA, OpenMP has been historically slower due to the abstractions and portability that OpenMP provides.

Recently however, this has been changing, see the New Device Runtime in LLVM's OpenMP implementation.

PROJECT GOALS

Allow OpenMP programmers to use remote CPUs and GPUs as if they were local ones – transparently offload.

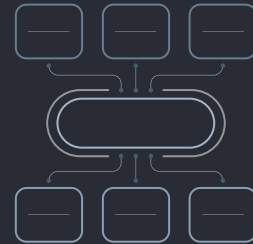
Developer Productivity

Programmers don't have to learn new models, nor change their workflows.



Performance

The resulting programs should be scalable.



Disclaimers

1. We are not trying to create a competitor to existing distributed programming models.
2. We are not experts in UCX, MPI, distributed programming, etc.
3. This is a proof-of-concept implementation.

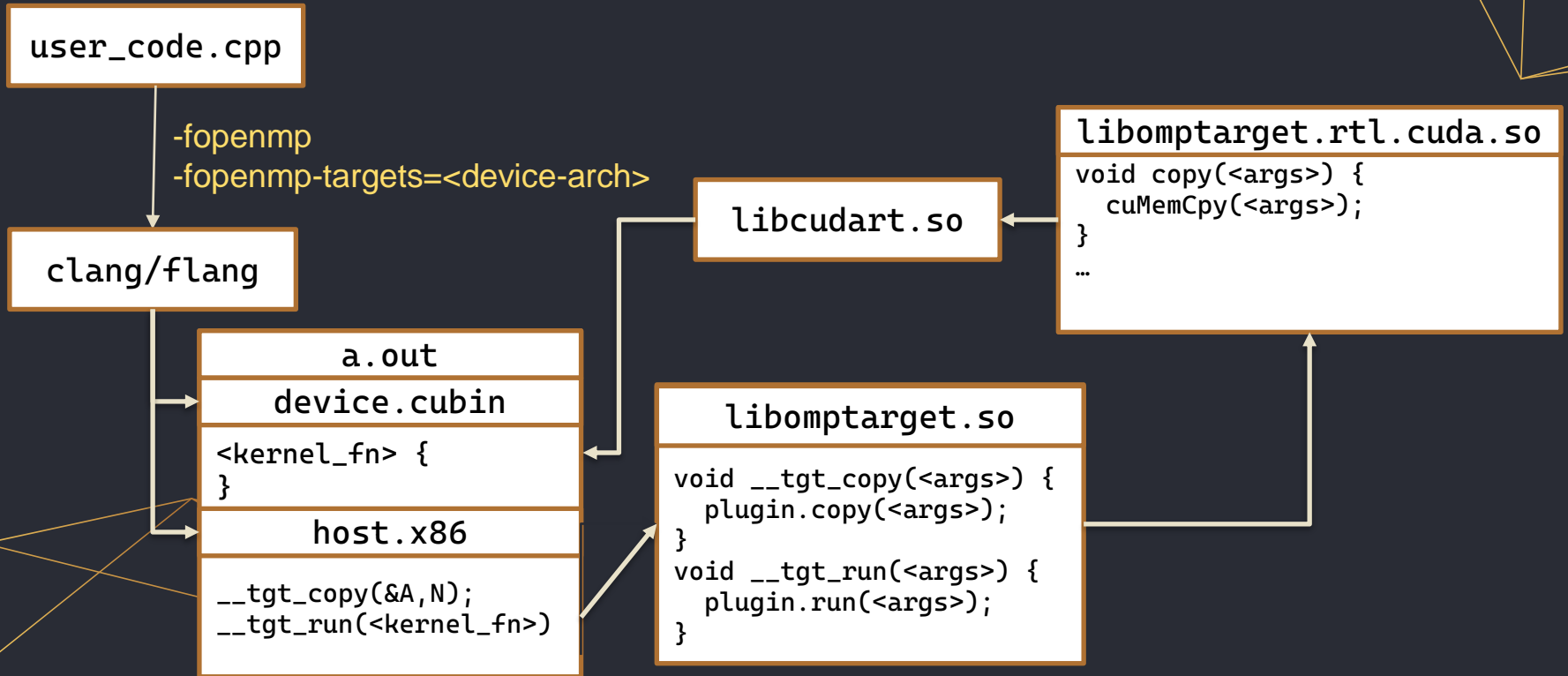
Basic Use Case – Distributed Merge Sort

```
    if (N <= Threshold) {
#pragma omp target depend(out:A[0]) device(getDev()) map(tofrom: A[:N])
        sort(A, &A[N]);
    } else {
#pragma omp task depend(out:A[0])
        mergesort_rec(A, N/2);

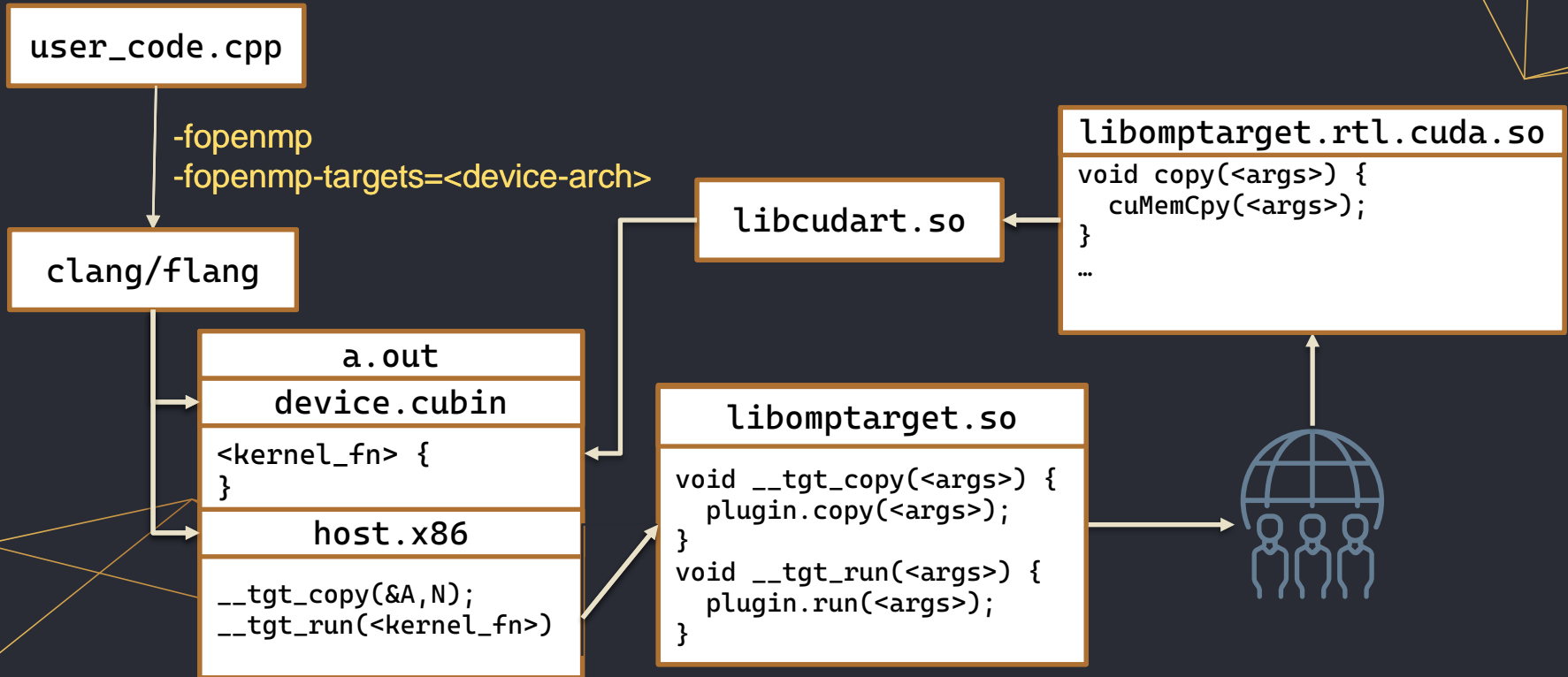
#pragma omp task depend(out:A[N/2])
        mergesort_rec(&A[N/2], N - N/2);

#pragma omp target depend(in:A[0], A[N/2]) device(getDev()) map(tofrom: A[:N])
        merge(&A[0], &A[N/2], &A[N]);
    }
```

OpenMP Target Offloading in LLVM



OpenMP Target Offloading in LLVM





Supported Device Architectures

AArch64

AMDGPU

CUDA

X86_64

PPC64/PPC64LE

VE



Backends



- Commodity-grade hardware support
- Protocol Buffers Serialization
- Available since LLVM 12

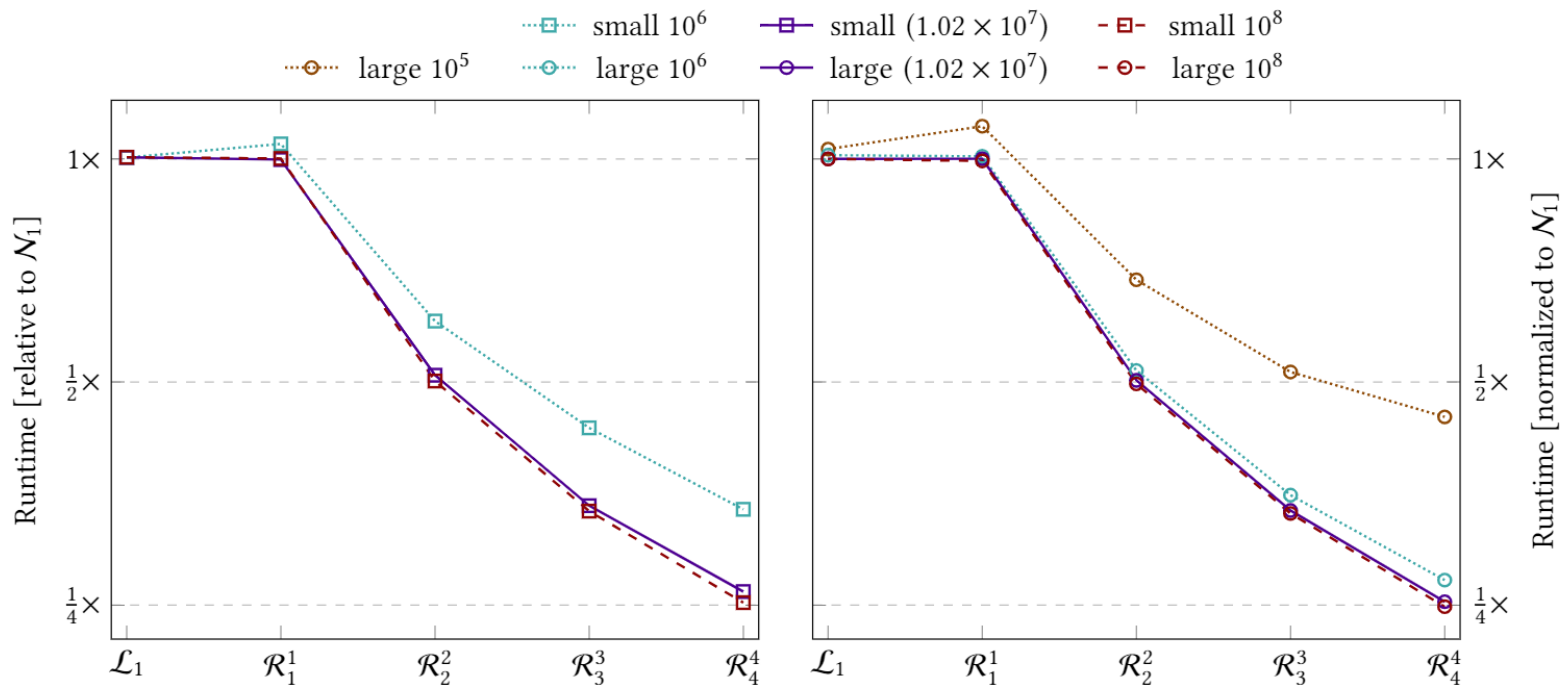


- HPC-grade hardware support
- Custom Serialization
- In development

gRPC

- RPC framework developed by Google
 - Support for load balancing, tracing, health checking and authentication
- Limitations:
 - Uses protocol buffers for Serialization
 - Optimized for small messages
 - Large Dependencies
 - Limited Purpose Zero-Copy Semantics
 - Limited to TCP/IP
- Tested on Google Cloud – Excellent Weak Scaling Results On Commodity Hardware

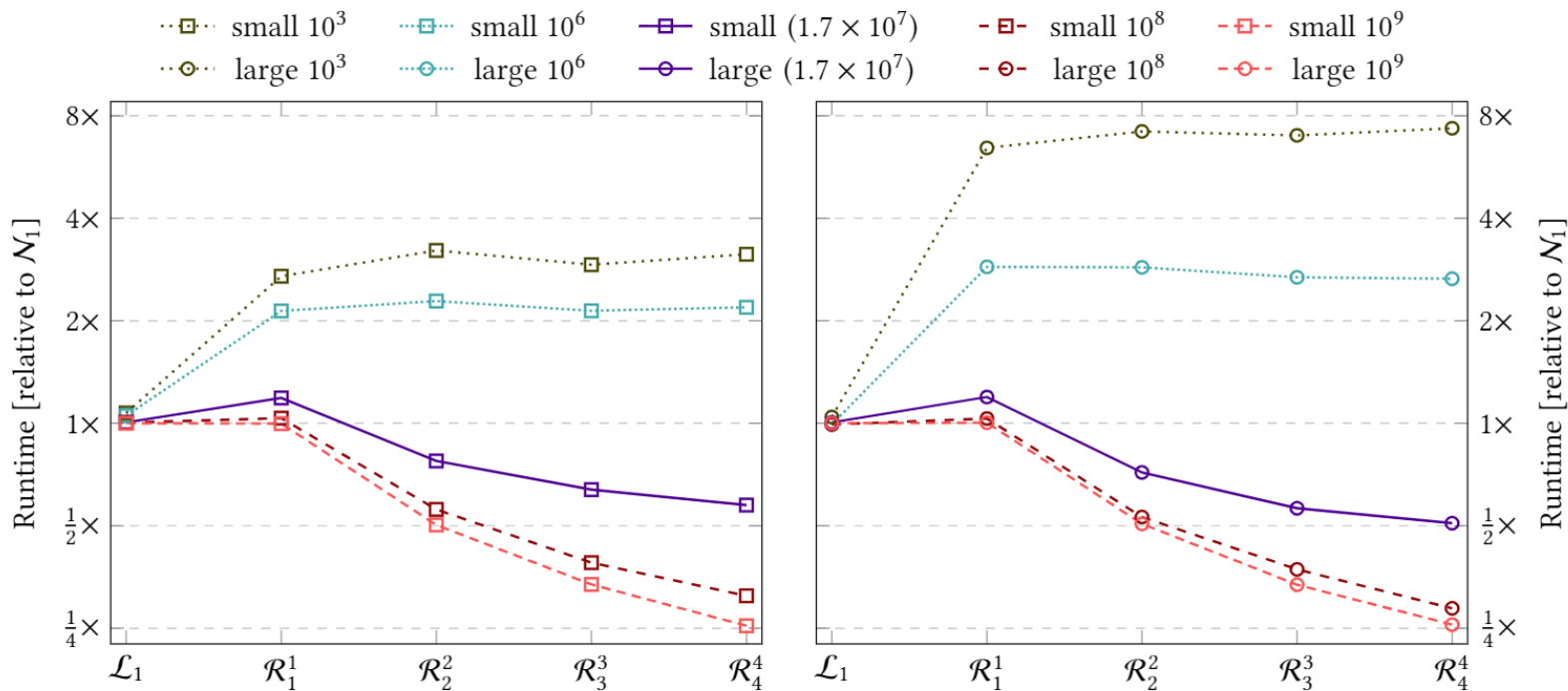
Google Cloud Evaluation (RSBench) – Weak Scaling



(a) Configurations running the Small problem size.

(b) Configurations running the Large problem size.

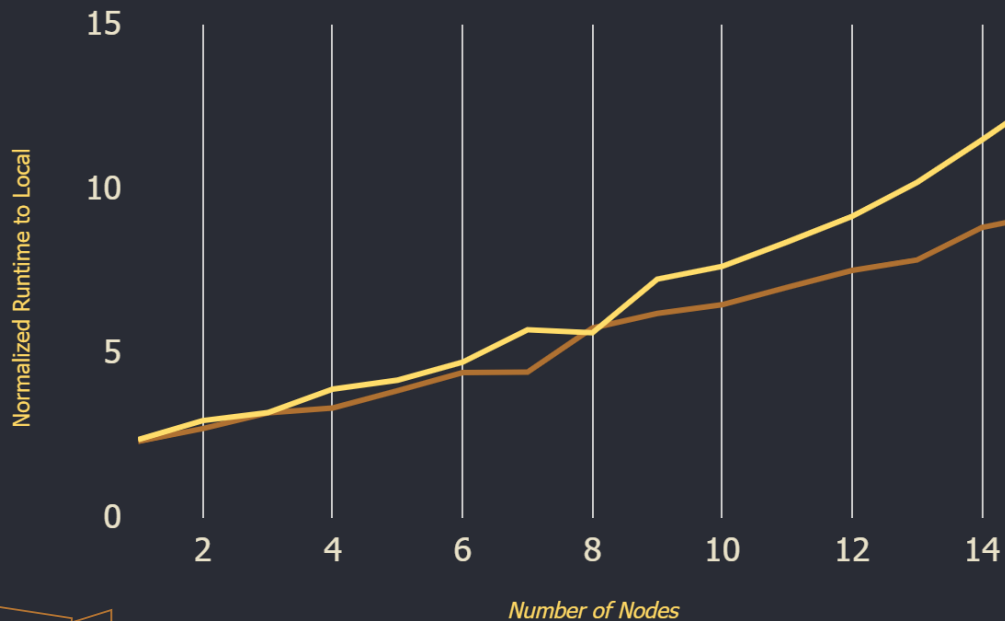
Google Cloud Evaluation (XSbench) – Weak Scaling



(a) Configurations running the Small problem size.

(b) Configurations running the Large problem size.

ThetaGPU Evaluation (XSBench) – Strong Scaling



— Small — Large

120+ A100 GPUs
With no source modifications

Largest Overhead
Data Transfers required for
initialization

Limitations

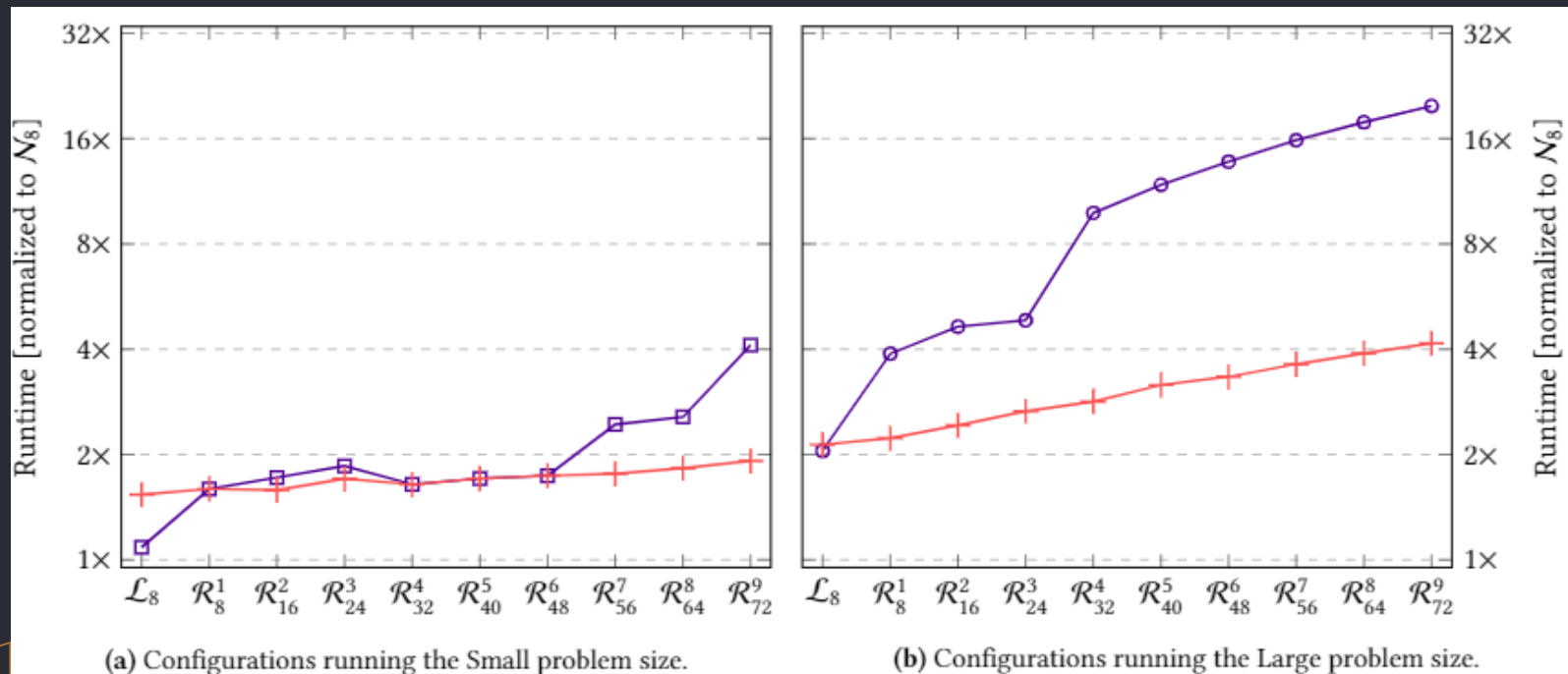
- UCX Implementation uses Tag-Matching Only; No RDMA, GPU-to-GPU Copy, etc.
 - Our implementation has high network costs in comparison to mature UCX users
 - Lots of further tuning required.
- Evaluation has Chicken-Egg Problem - OpenMP applications are typically not multi-GPU
- Applications need a sufficiently long compute time to rationalize the network costs
- Limitations of OpenMP Accelerator Device Model

OpenMP Device Model

- Each accelerator is independent.
- No concept of hierarchical offloading
- Leads to possibly redundant network operations
- Limited to Star-Topology



ThetaGPU Evaluation (gRPC-RSBench) – Strong Scaling




Advantages

- Reasonably efficient and highly scalable on commodity and HPC
- Requires no source modifications to scale from 1 GPU to 100+ GPUs
- Support for many architectures, as well as across architectures
 - i.e. X86 ⇔ ARM offloading with SmartNICs

Vision



- Incorporate more UCX features into Remote OpenMP Offloading
 - Explore compression
 - Explore hierarchical offloading opportunities within OpenMP
 - Find applications within SmartNICs and beyond that can use Remote OpenMP Offloading
- 



THANKS!

QA

CREDITS: This presentation template was created by [Slidesgo](#), including icons by [Flaticon](#), infographics & images by [Freepik](#).