SnuCL D

A Distributed OpenCL Framework using Redundant Computation and Data Replication

Junghyun Kim, Gangwon Jo, Jaehoon Jung, Jungwon Kim, and Jaejin Lee

Center for Manycore Programming
Department of Computer Science and Engineering
Seoul National University, Korea

Center for Manycore Programming
매니코어 프로그래밍 연구단

Multicore Computing Research Laboratory
멀티코어 컴퓨팅 연구실

Outline

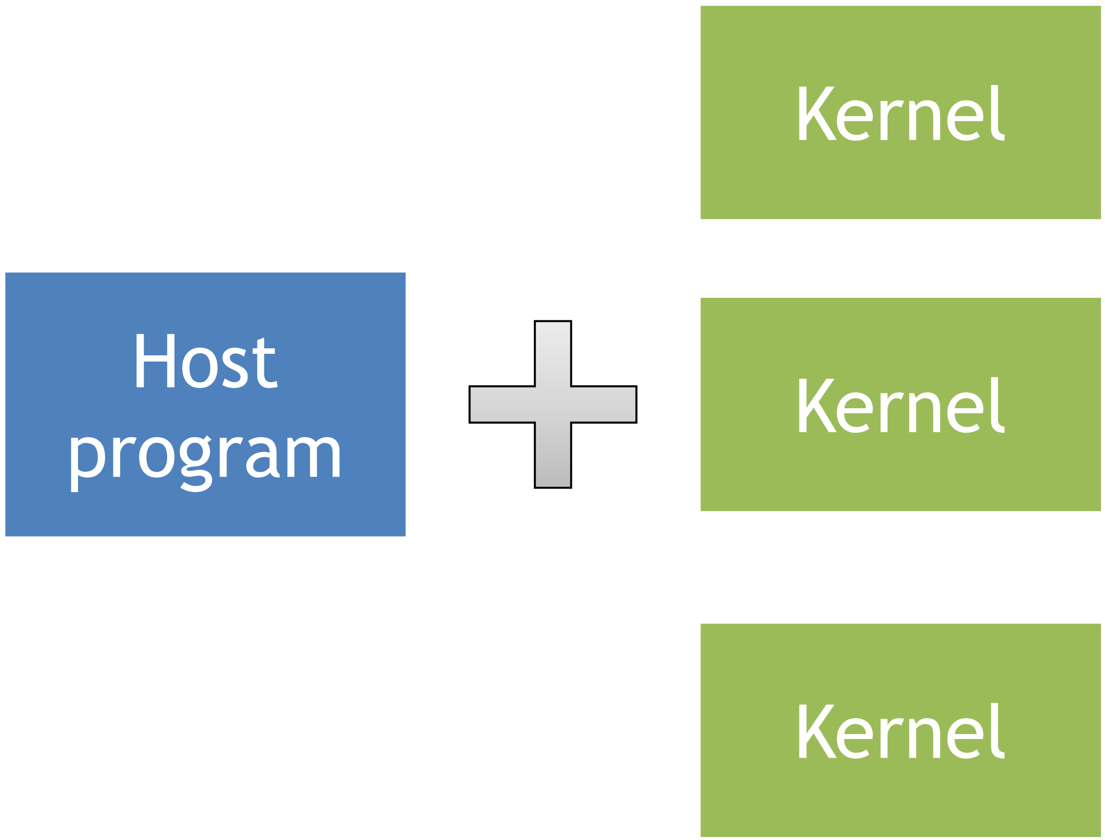
- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

Introduction

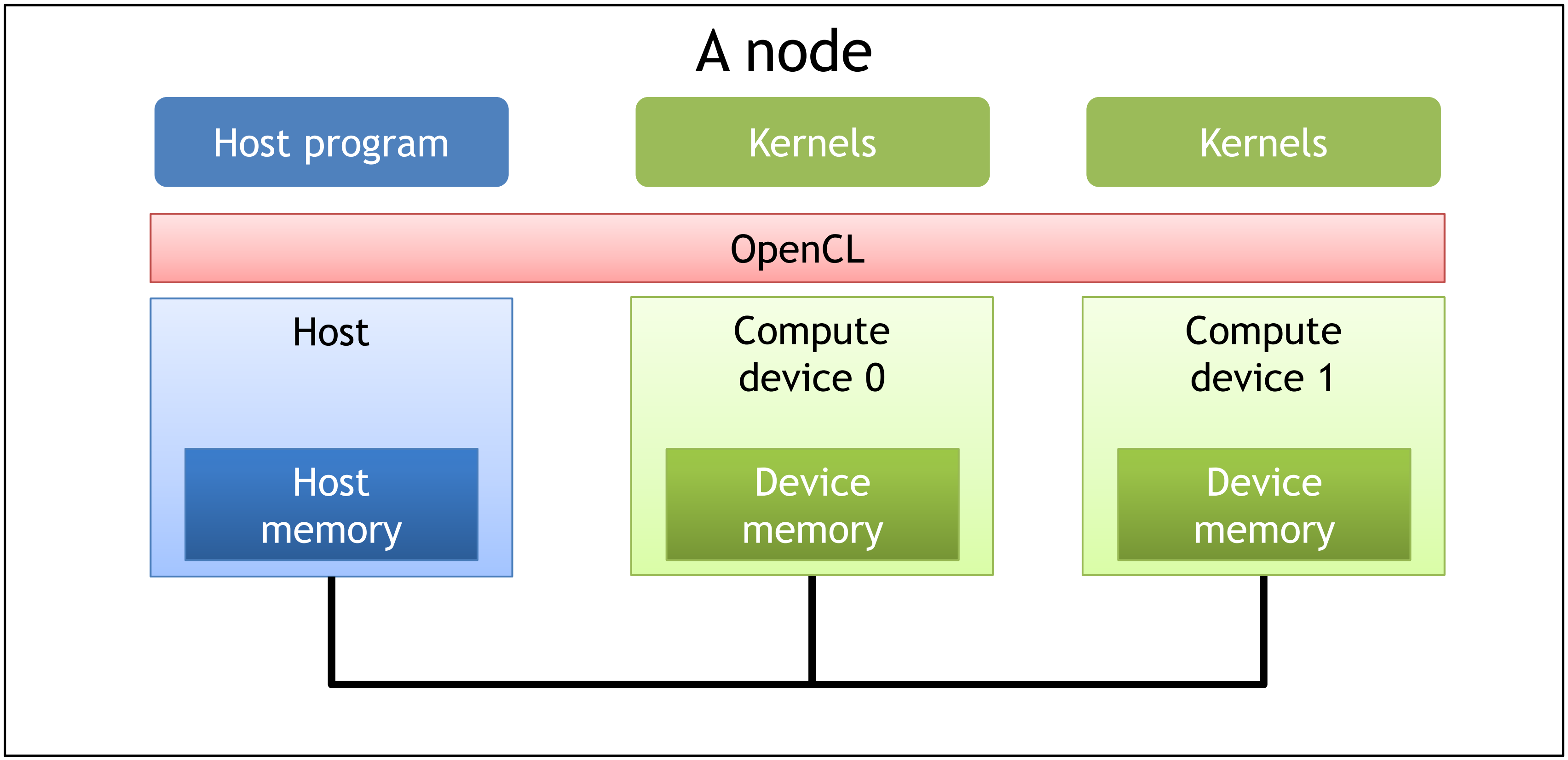
- Heterogeneous systems
  - Different types of processors
    - E.g., CPUs+GPUs

- Major programming models
  - CUDA and OpenCL

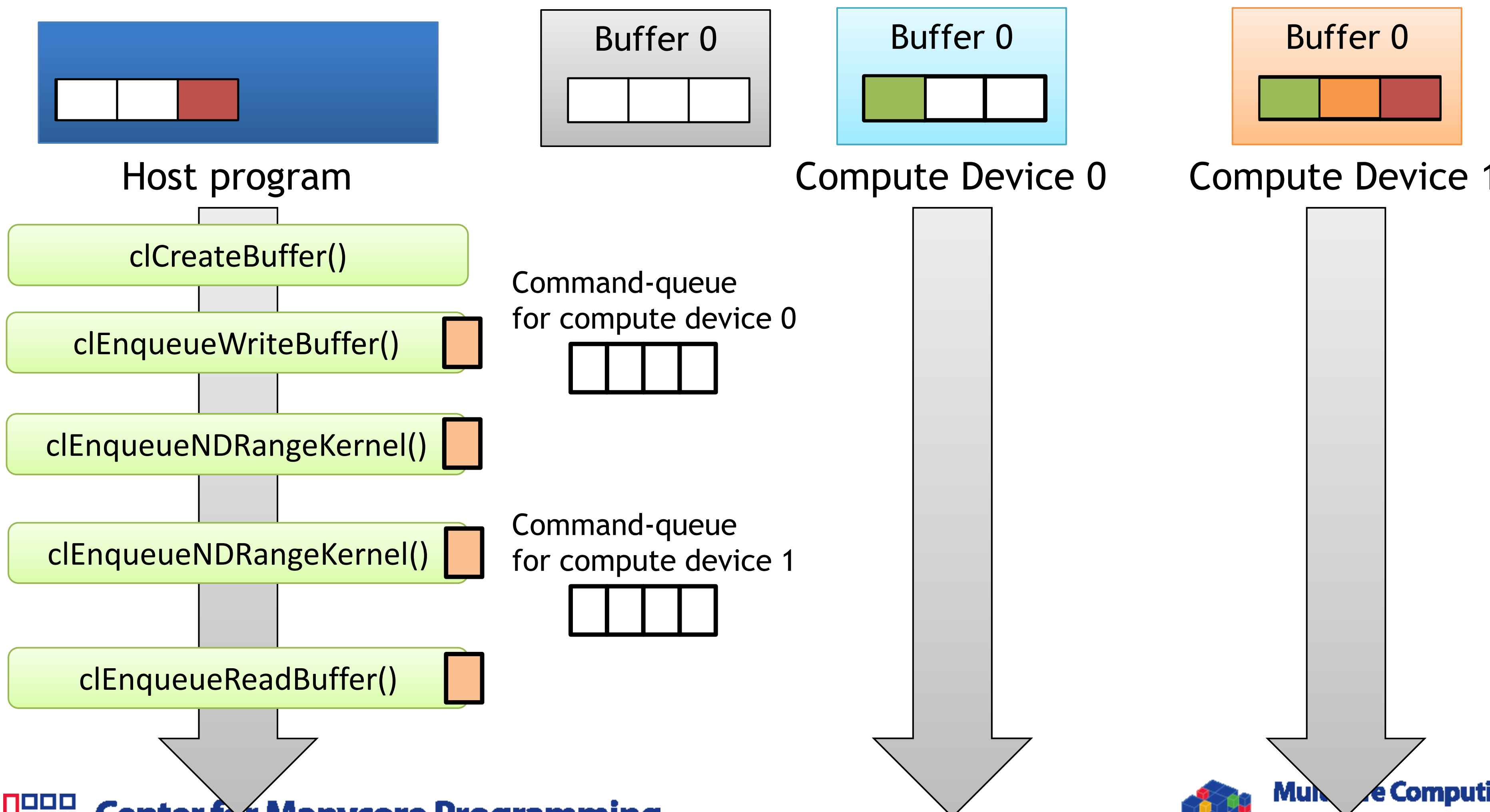
OpenCL Program



OpenCL Platform Model

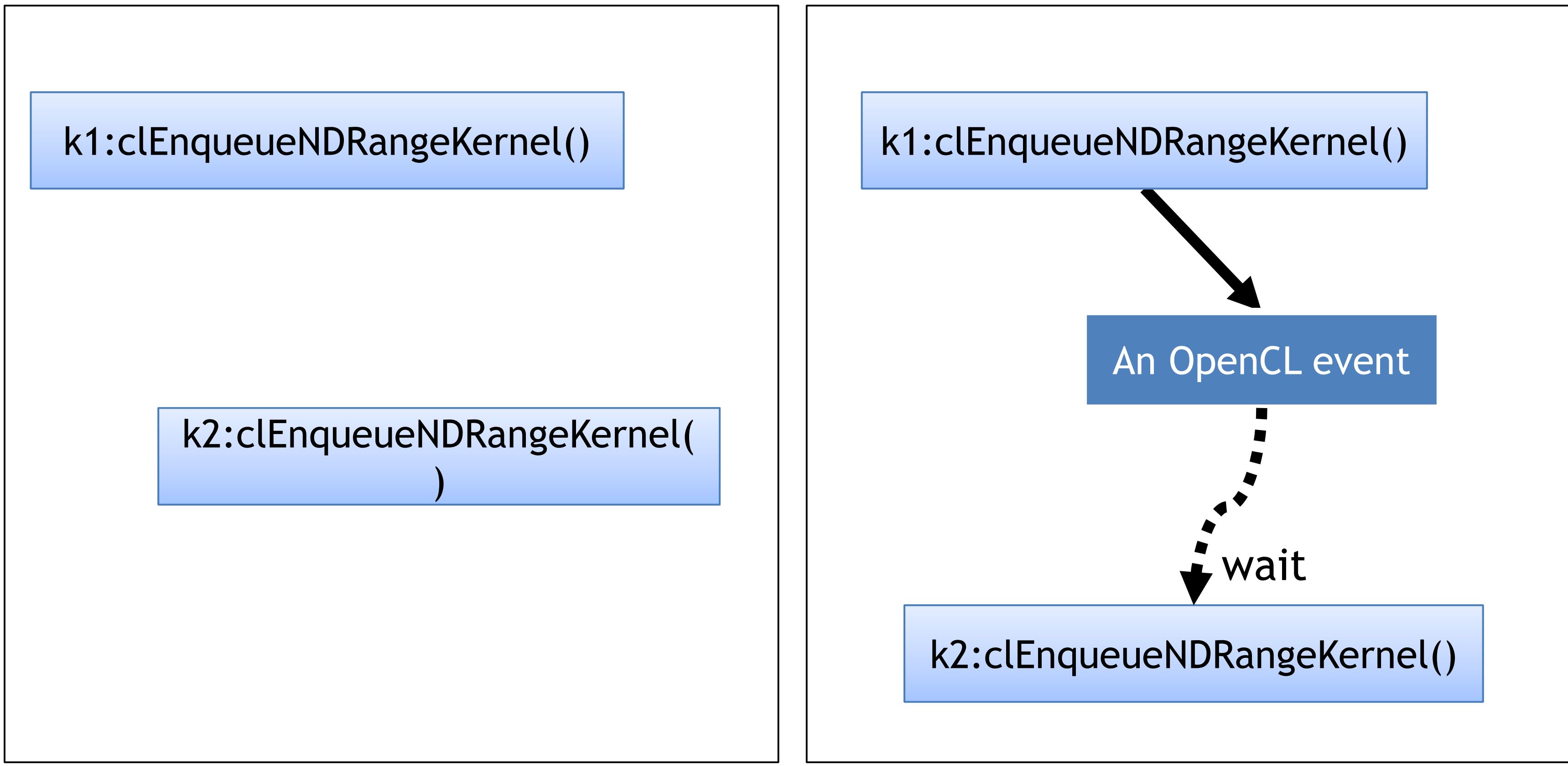


OpenCL Programming Model

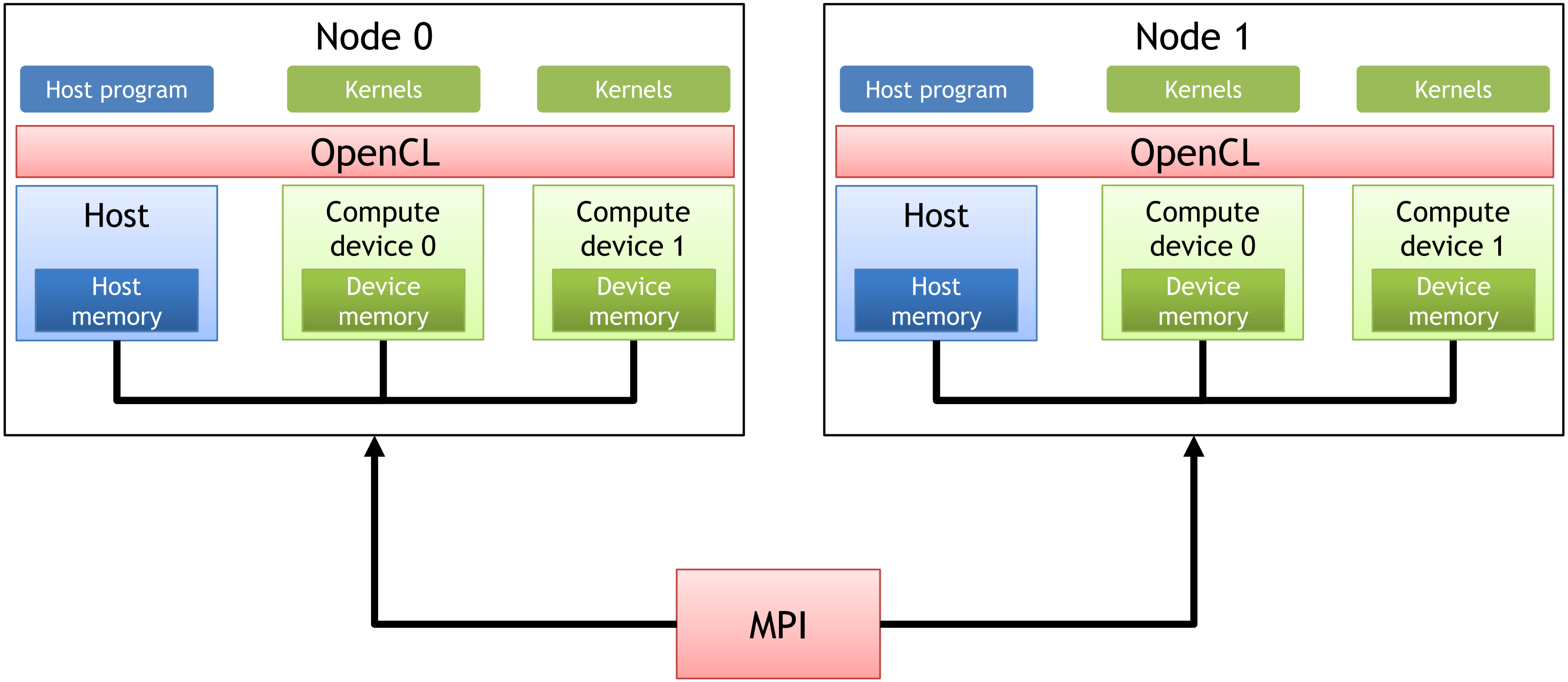


Order of Commands

Unordered **Ordered**



OpenCL for Heterogeneous Clusters



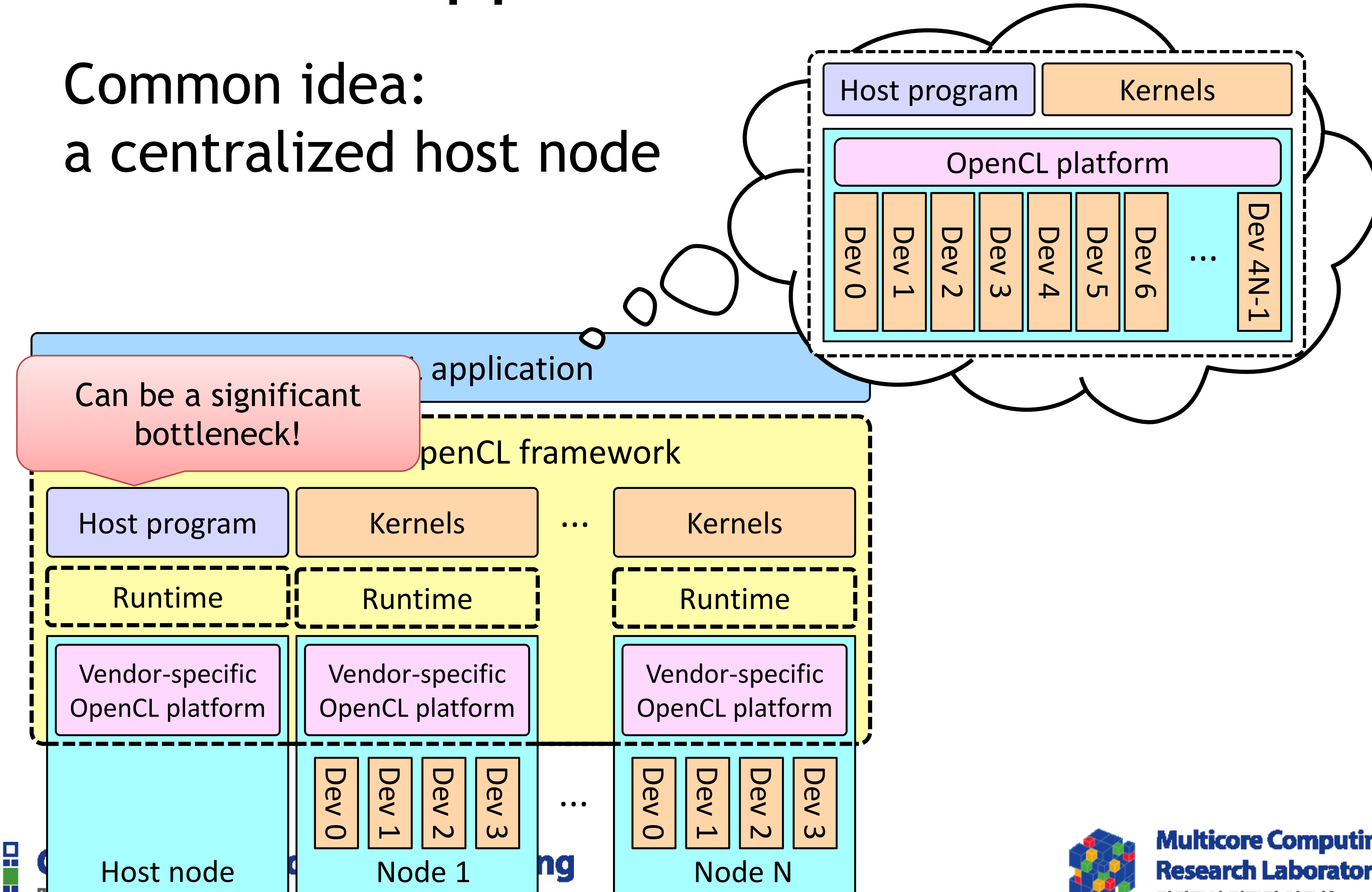
MPI+OpenCL: cumbersome and error-prone

Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

Previous Approaches for Clusters

Common idea:
a centralized host node



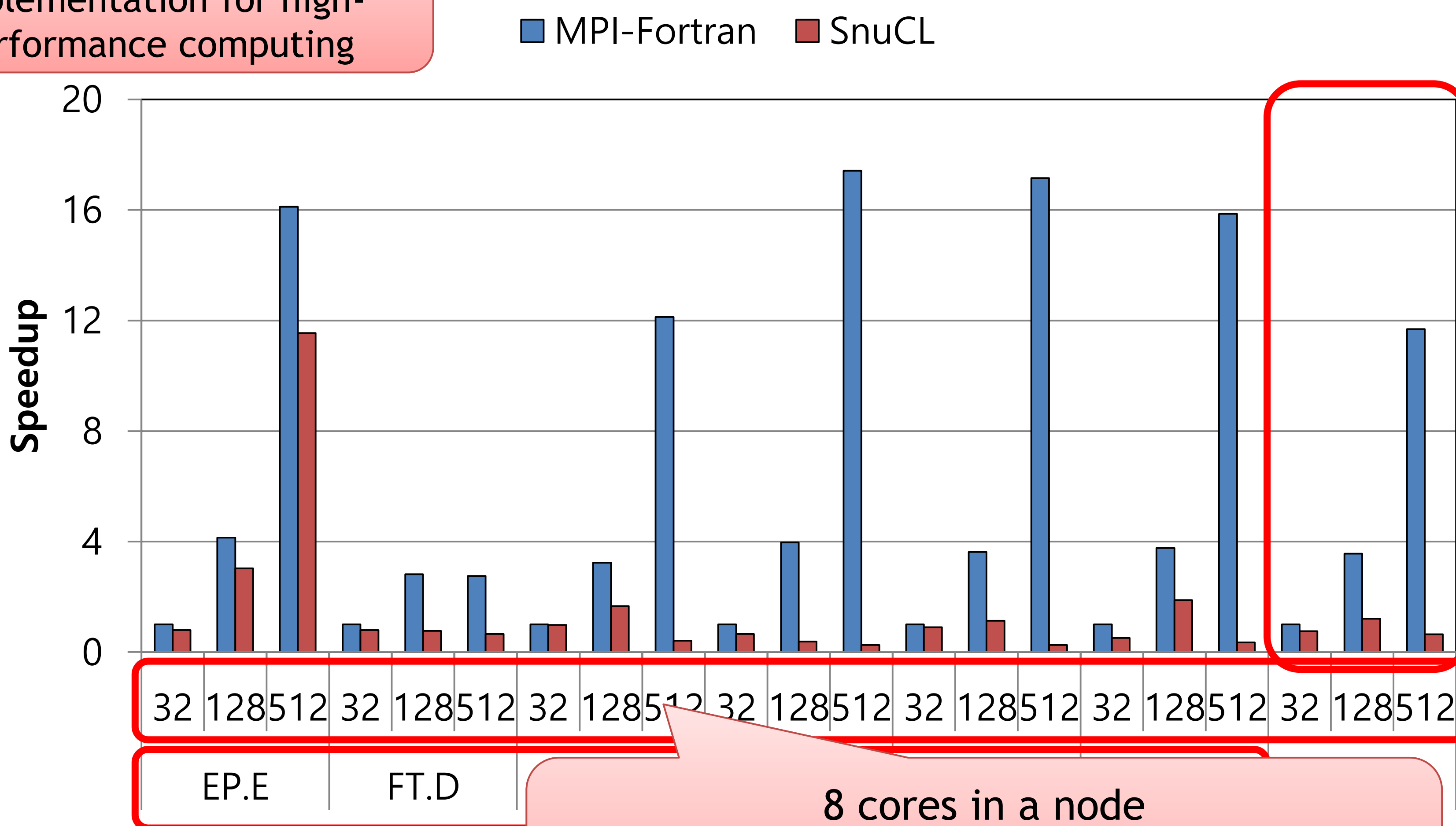
Centralized Approaches

- clOpenCL
- Hybrid OpenCL
- SocketCL
- dOpenCL
- CLara
- SnuCL
- DistributedCL
- CLuMPI
- rCUDA
- DS-CUDA

Evaluates it on a large number of nodes (256 nodes)

MPI-Fortran vs. SnuCL

The most efficient implementation for high-performance computing



8 cores in a node
1 core / MPI process
A set of 4 cores / OpenCL compute device

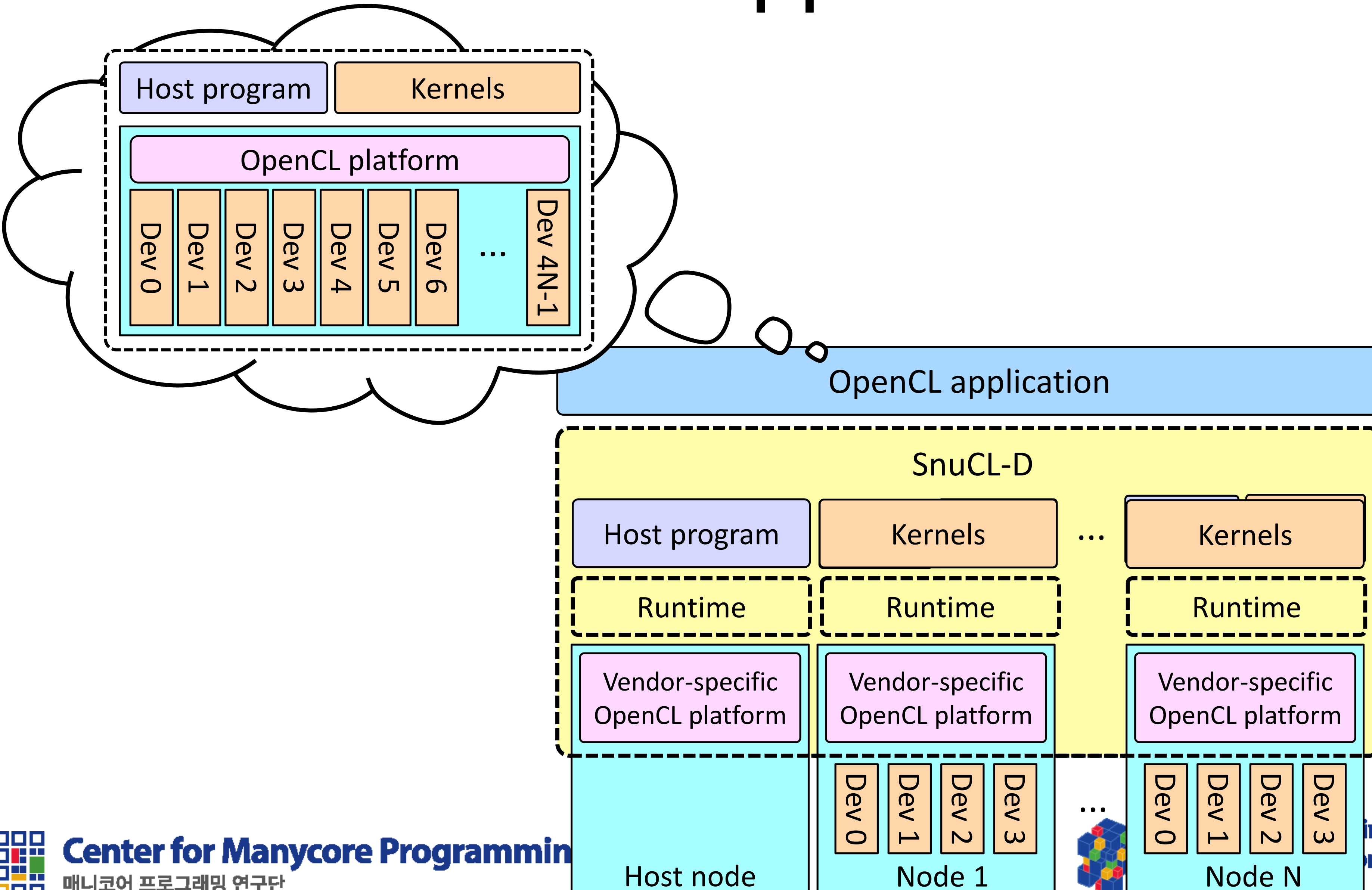
Goals

- Develop a scalable OpenCL framework for clusters
  - Comparable to MPI-Fortran
  - Achieve ease of programming with high performance
- Key idea
  - Eliminate the centralized host node
    - Redundant host computation
    - Data replication
- SnuCL-D
  - The successor of SnuCL
  - A distributed OpenCL framework

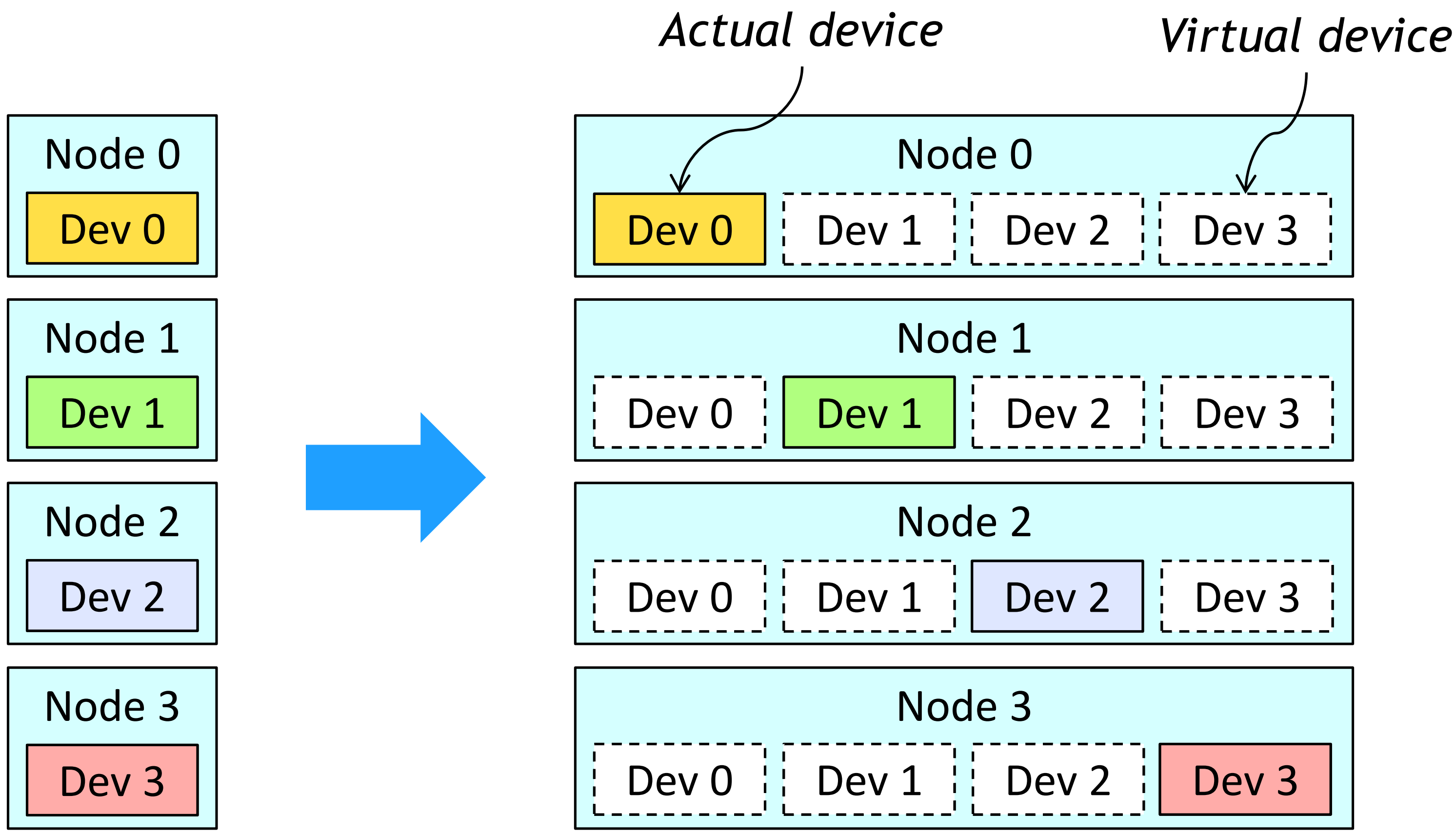
Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

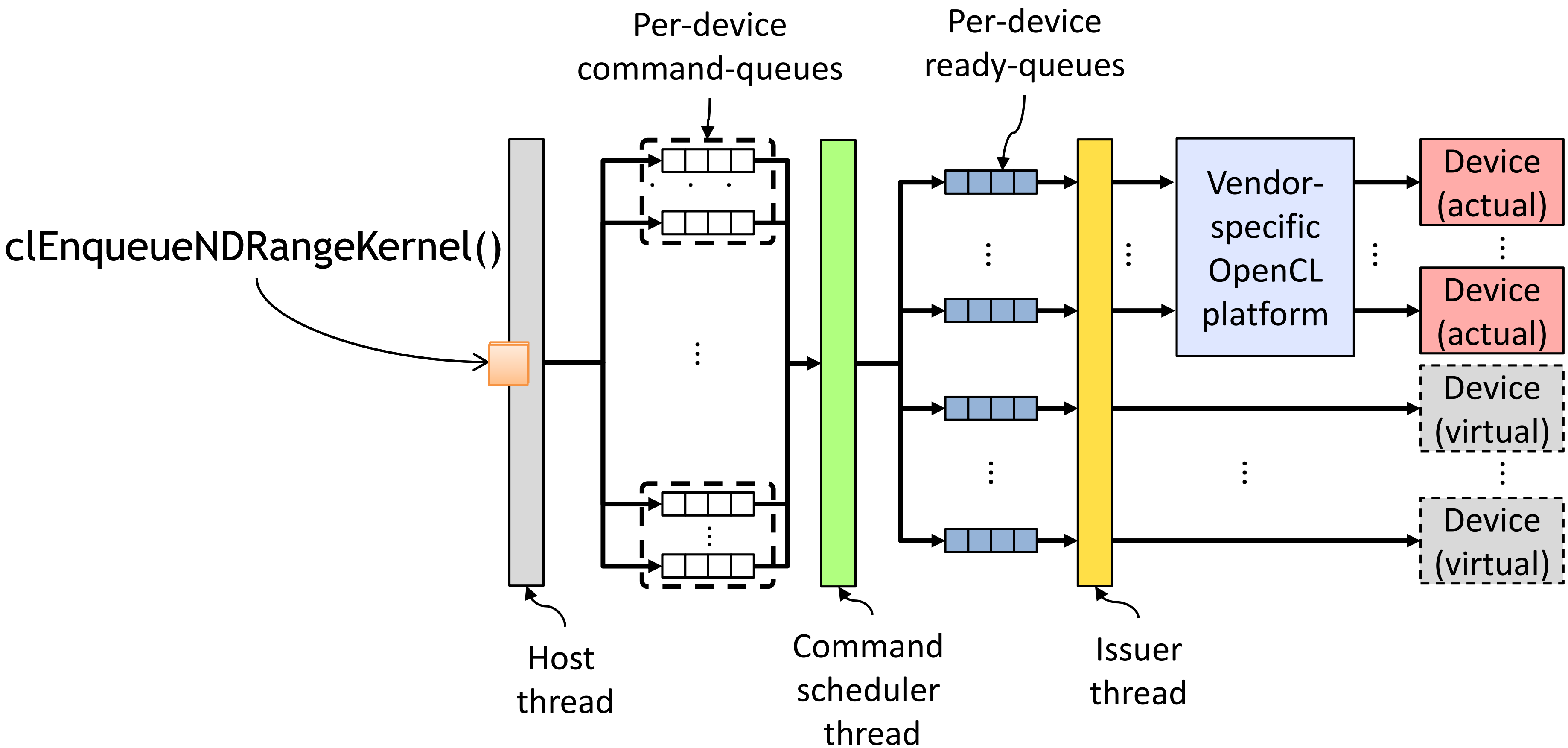
SnuCL-D's Approach



Remote Device Virtualization



SnuCL-D Runtime



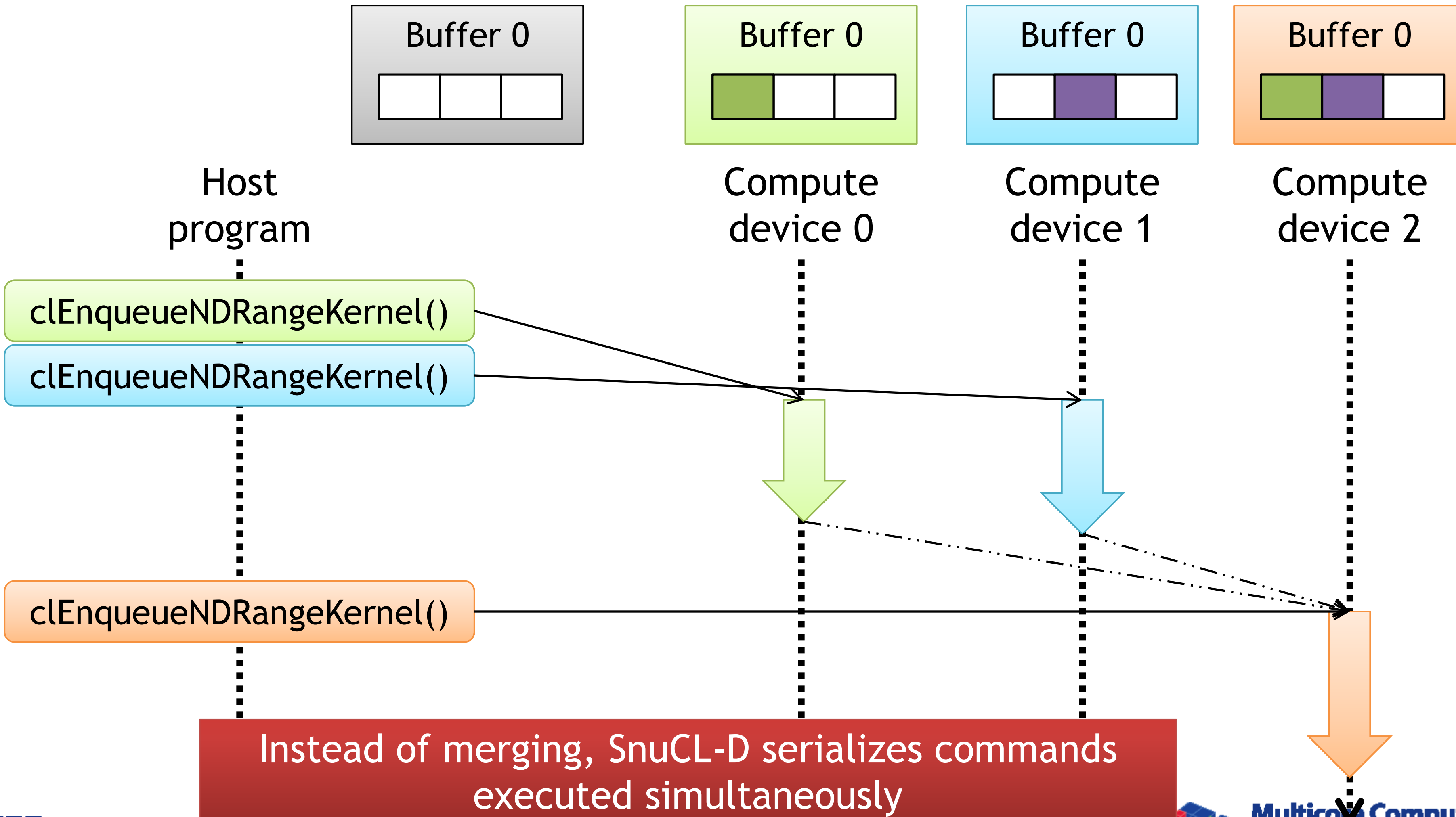
Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

Correctness Problems

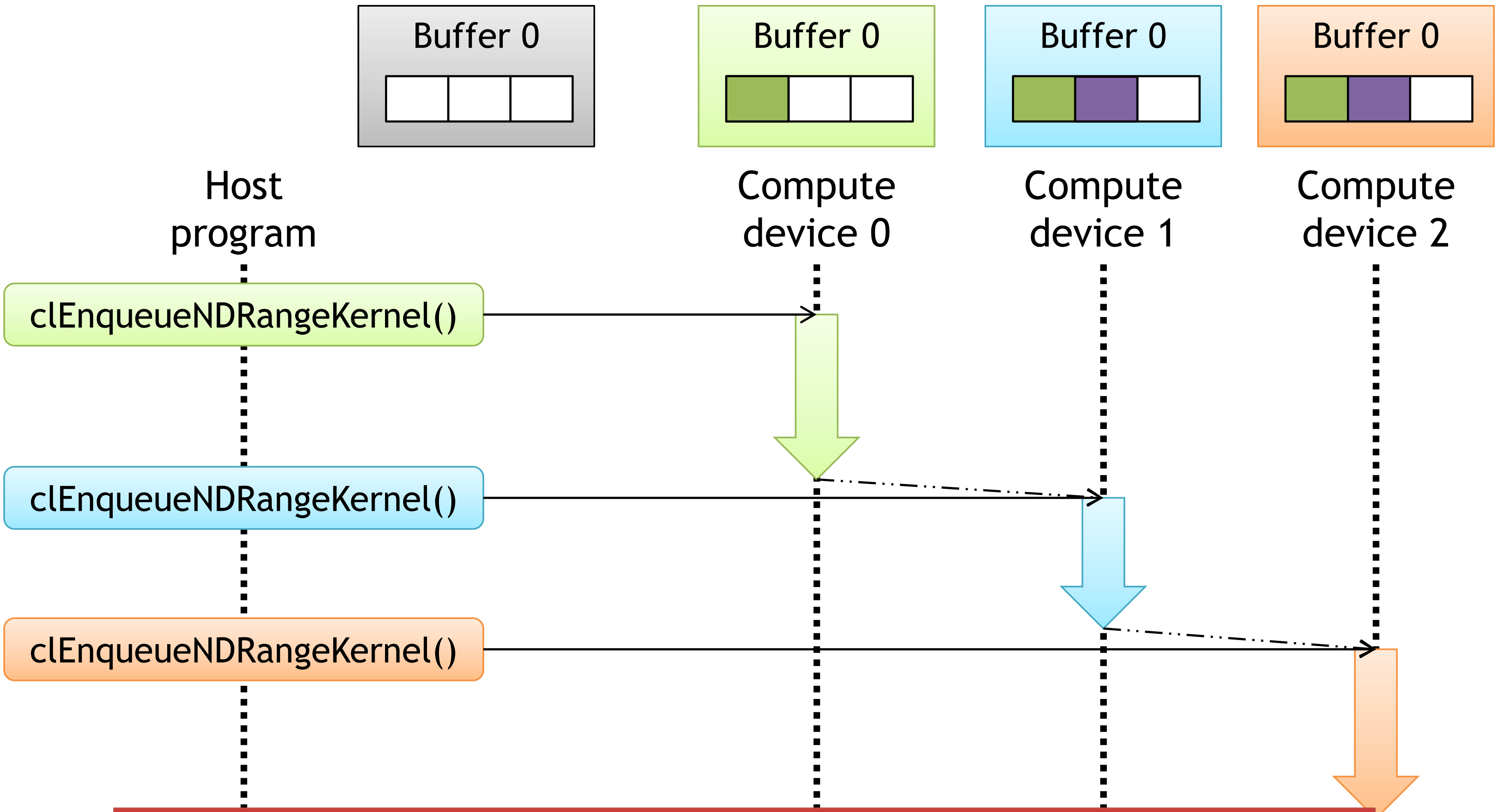
- Consistency problem
- Non-determinacy problem

Consistency Problem (Simultaneous Accesses)



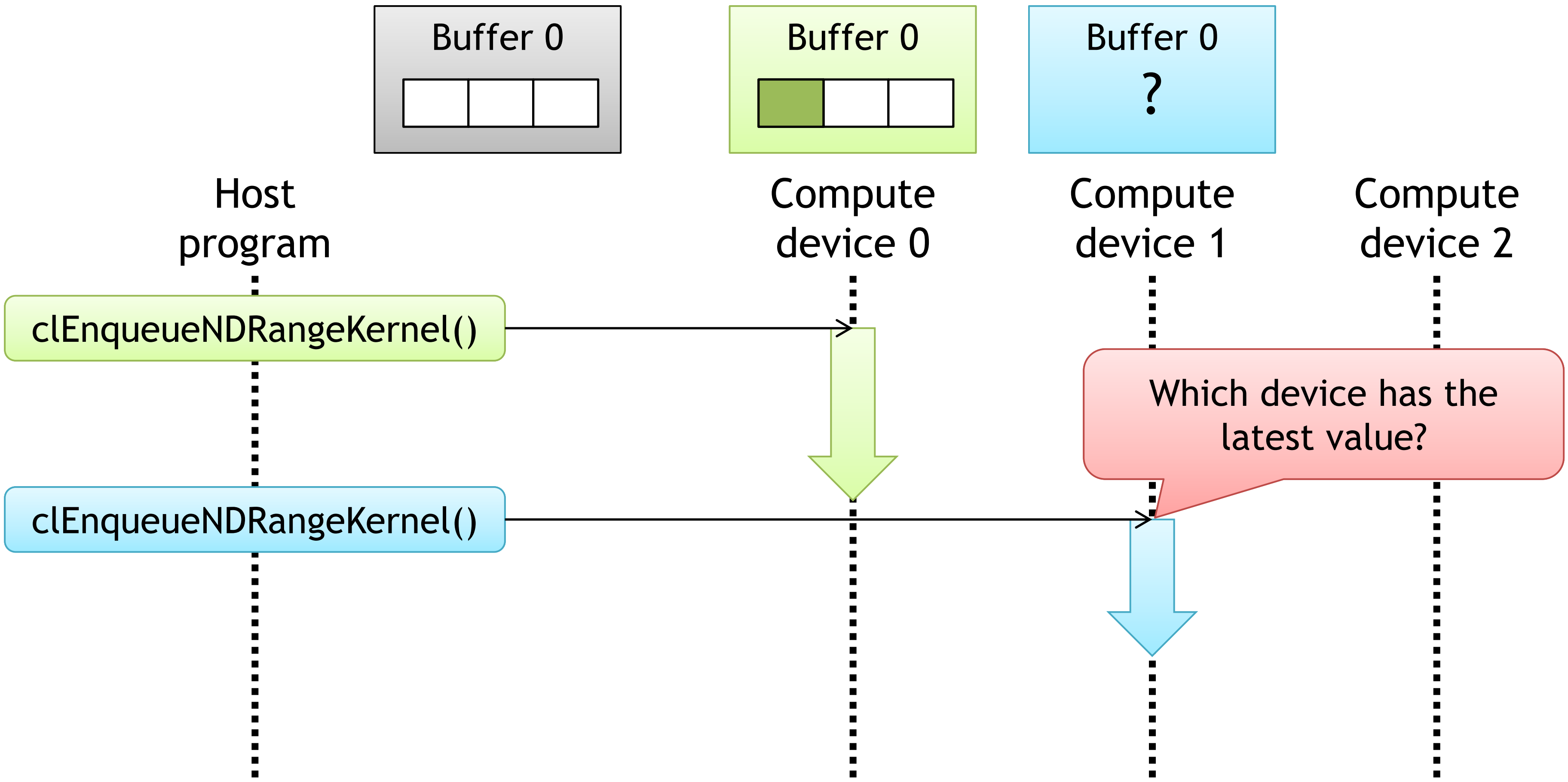
Instead of merging, SnuCL-D serializes commands executed simultaneously

Consistency Problem (Simultaneous Accesses)



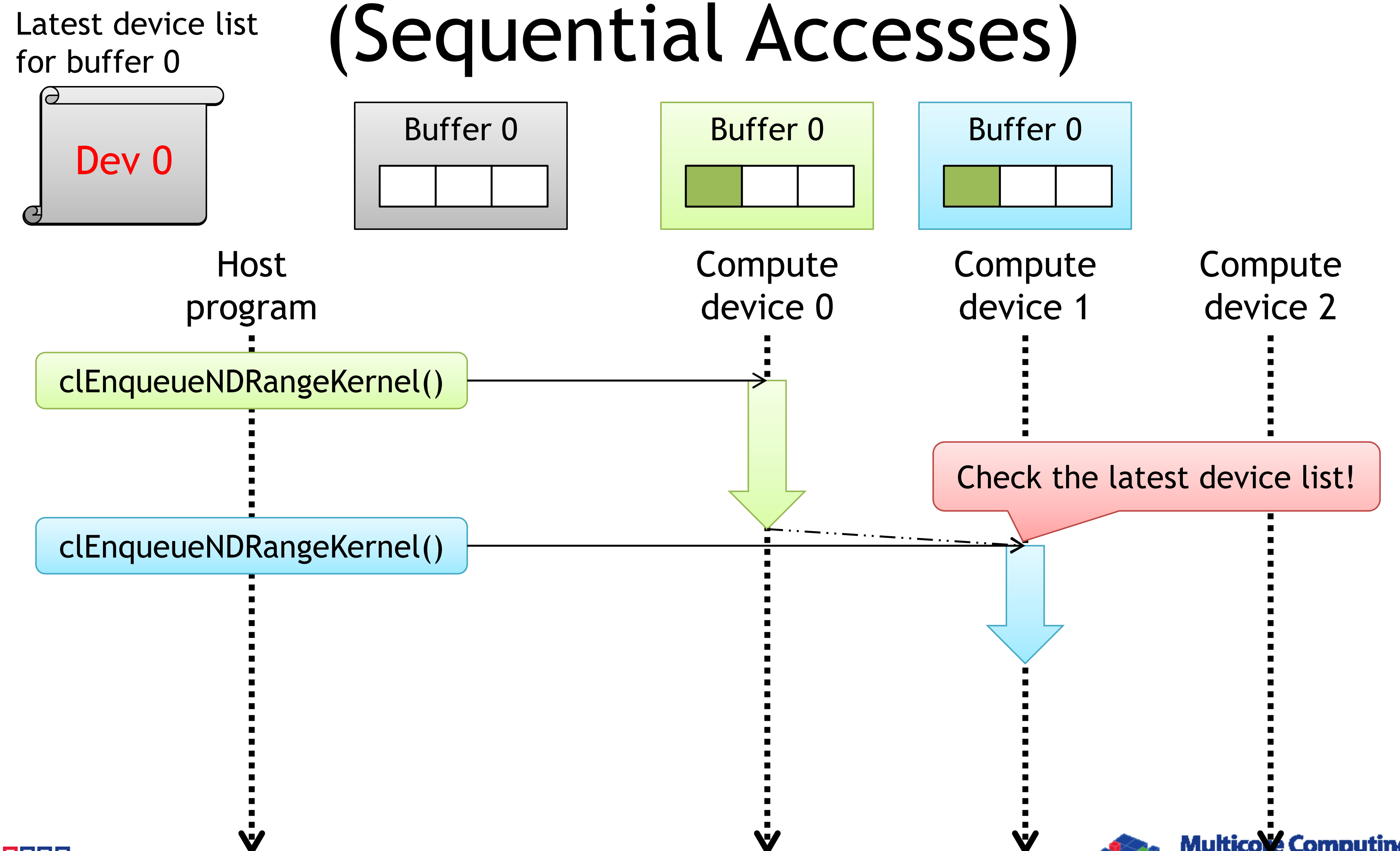
There is no simultaneous accesses by multiple commands in SnuCL-D

Consistency Problem (Sequential Accesses)



SnuCL-D maintains the latest device list for each memory object

Consistency Problem (Sequential Accesses)



Correctness Problems

- Consistency problem
  - Solved by serialization and latest device lists
- Non-determinacy problem

Non-deterministic Command Scheduling (Single-threaded Host Programs)

- Problem
  - Redundant command scheduling on every node
  - If there is no enforced order between commands,
    - Command execution order can be different across nodes
    - May cause a deadlock, data inconsistency, etc.
- Solution
  - The enqueueing order is fixed
    - The order of clEnqueue…() calls
  - SnuCL-D enforces the enqueueing order

Non-deterministic Command Scheduling (Multi-threaded Host Programs)

- Problem
  - Even the enqueueing order is not guaranteed

- Solution
  - Can be solved by deterministic multithreading
    - E. D. Berger [OOPSLA 09], C. Bienia [PACT 08], T. Liu [SOSP 11], M. Olszewski [ASPLOS 09]
  - Using this, we can make the enqueueing order deterministic

- Single-threaded host programs are more common
  - SnuCL-D assumes single-threaded host programs

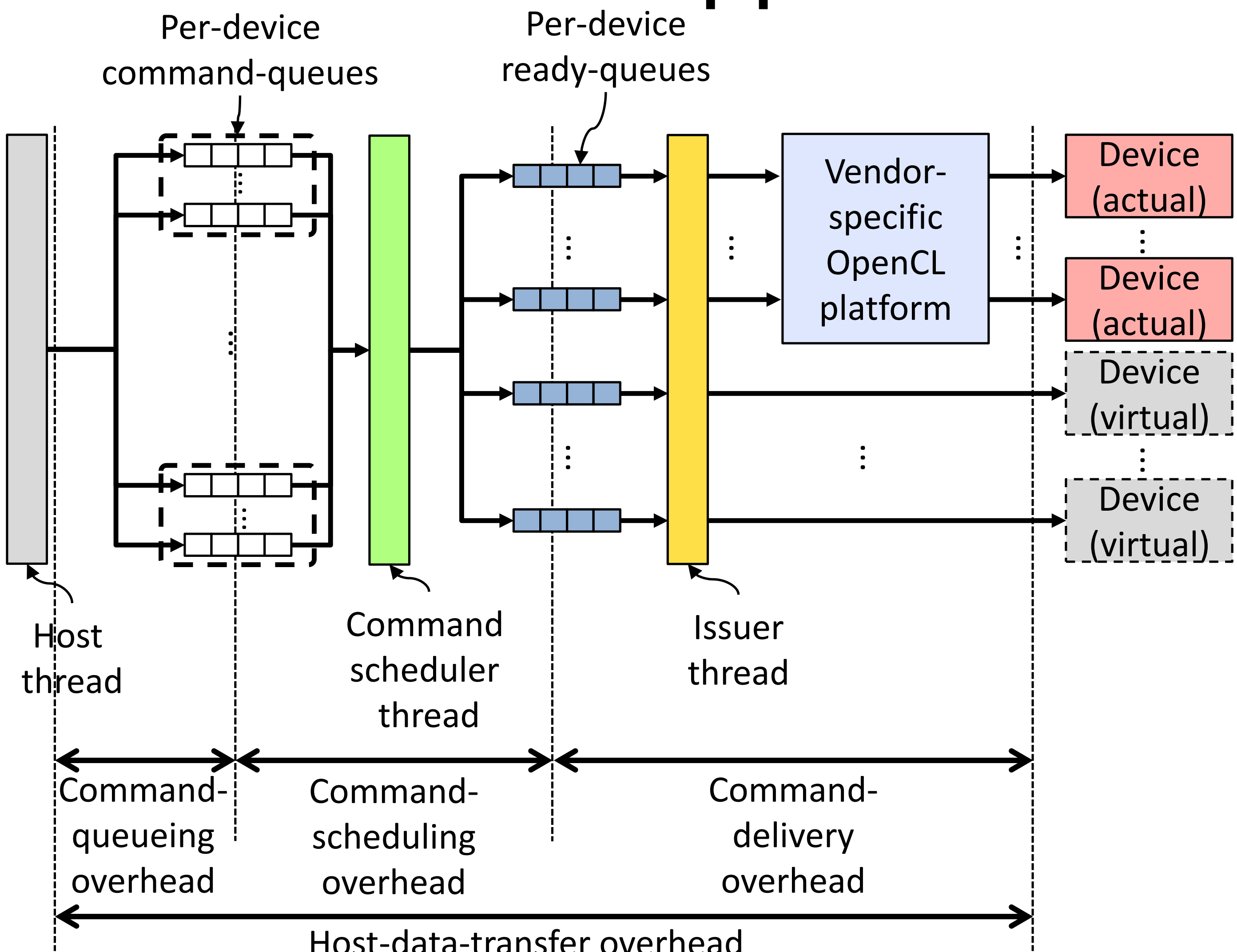
Non-deterministic Result of a Function Call

- Problem
  - The result of a function call can be different across nodes
  - E.g., file I/O and srand()
- Solution
  - Use global synchronization between nodes
  - After designating a root node, the root node performs the call in the host program
  - The others receive the result from the root

Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

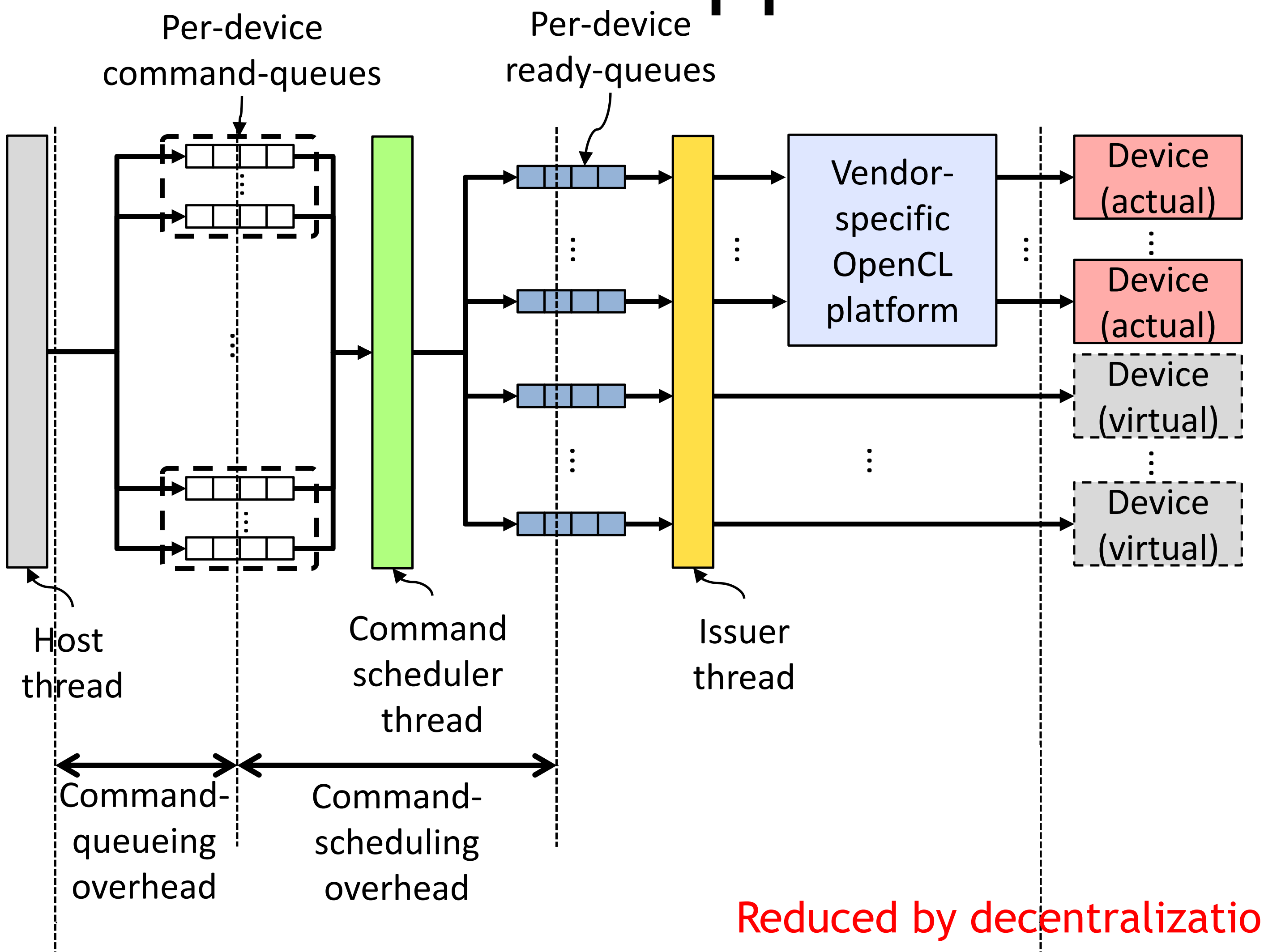
Runtime Overheads of Centralized Approaches



Decentralization Technique

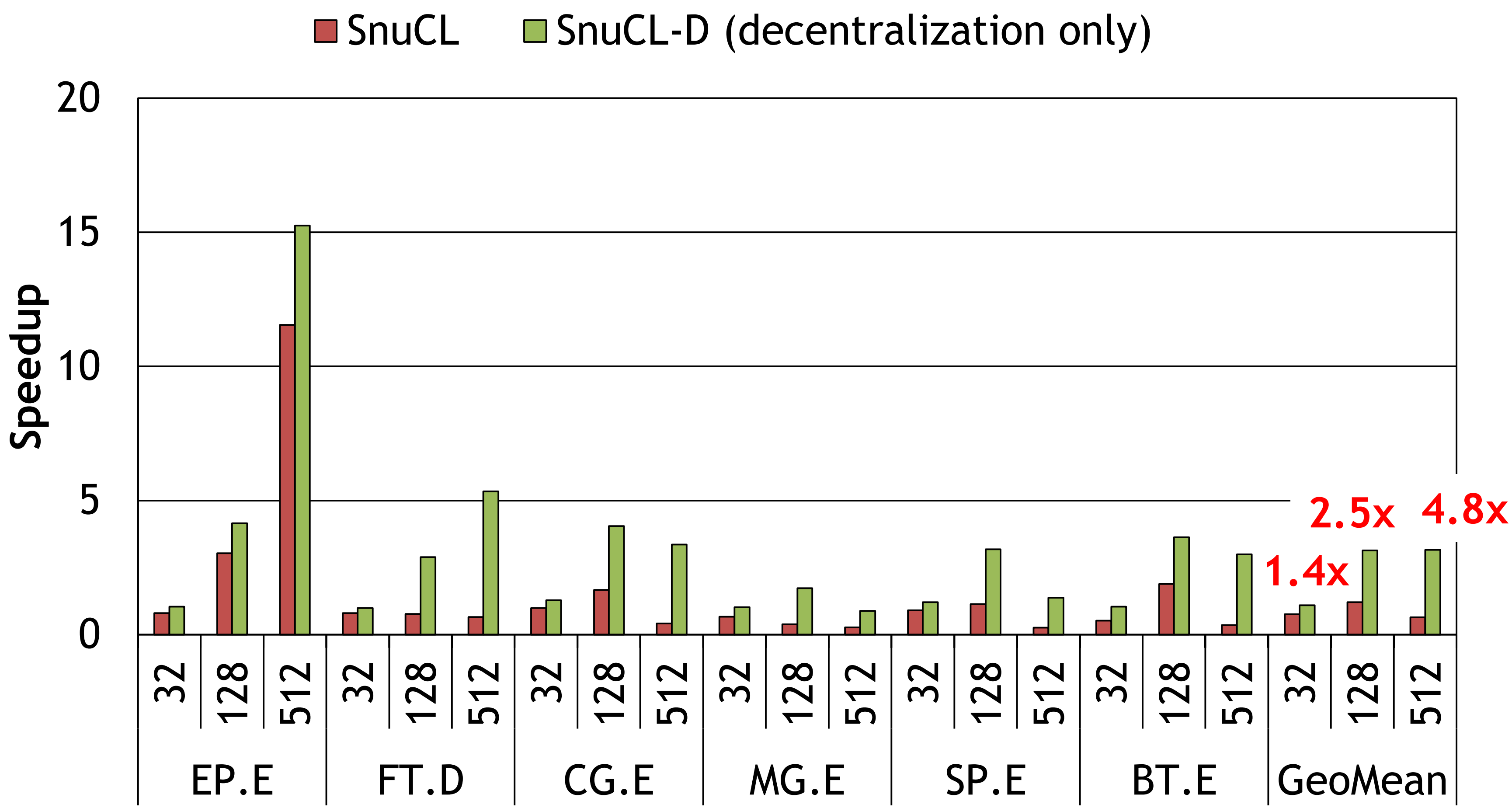
- To reduce
  - Command-delivery overhead
  - Host-data-transfer overhead
- The host program is executed on every node
  - Redundant computation
  - Data replication
- Remote device virtualization
  - Deliver commands to only actual devices

Runtime Overheads of Centralized Approaches

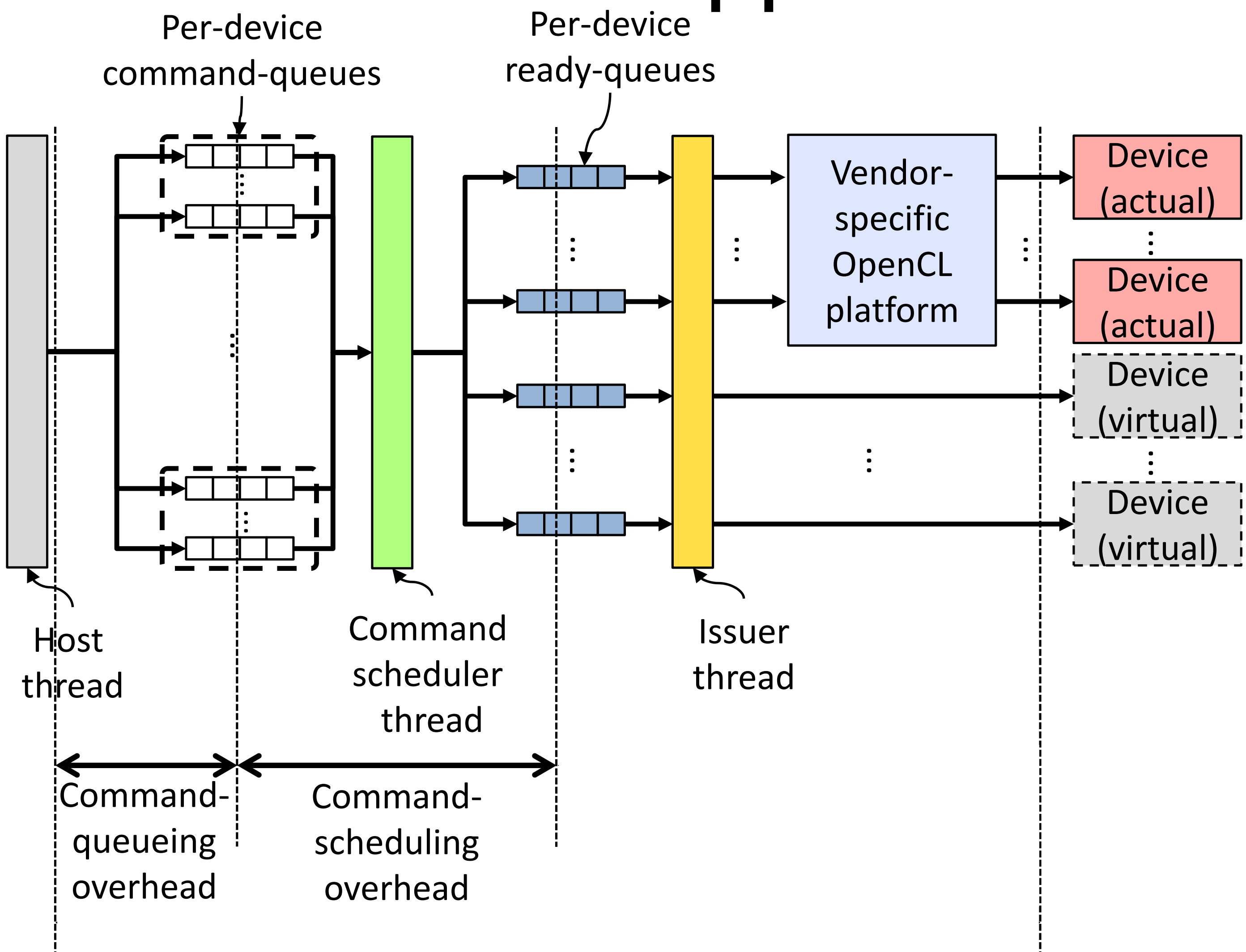


Reduced by decentralization

Performance



Runtime Overheads of Centralized Approaches

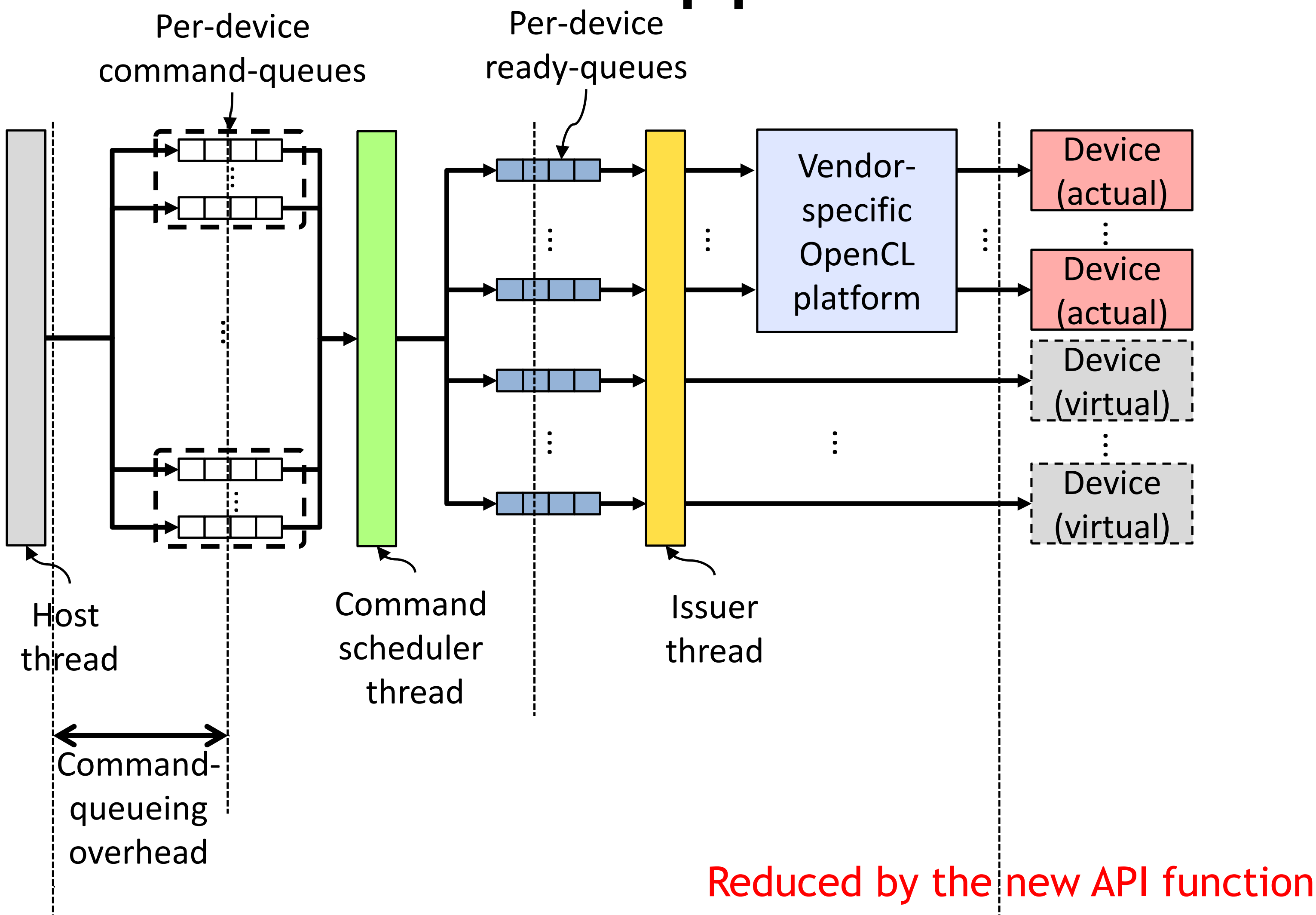


New API Function

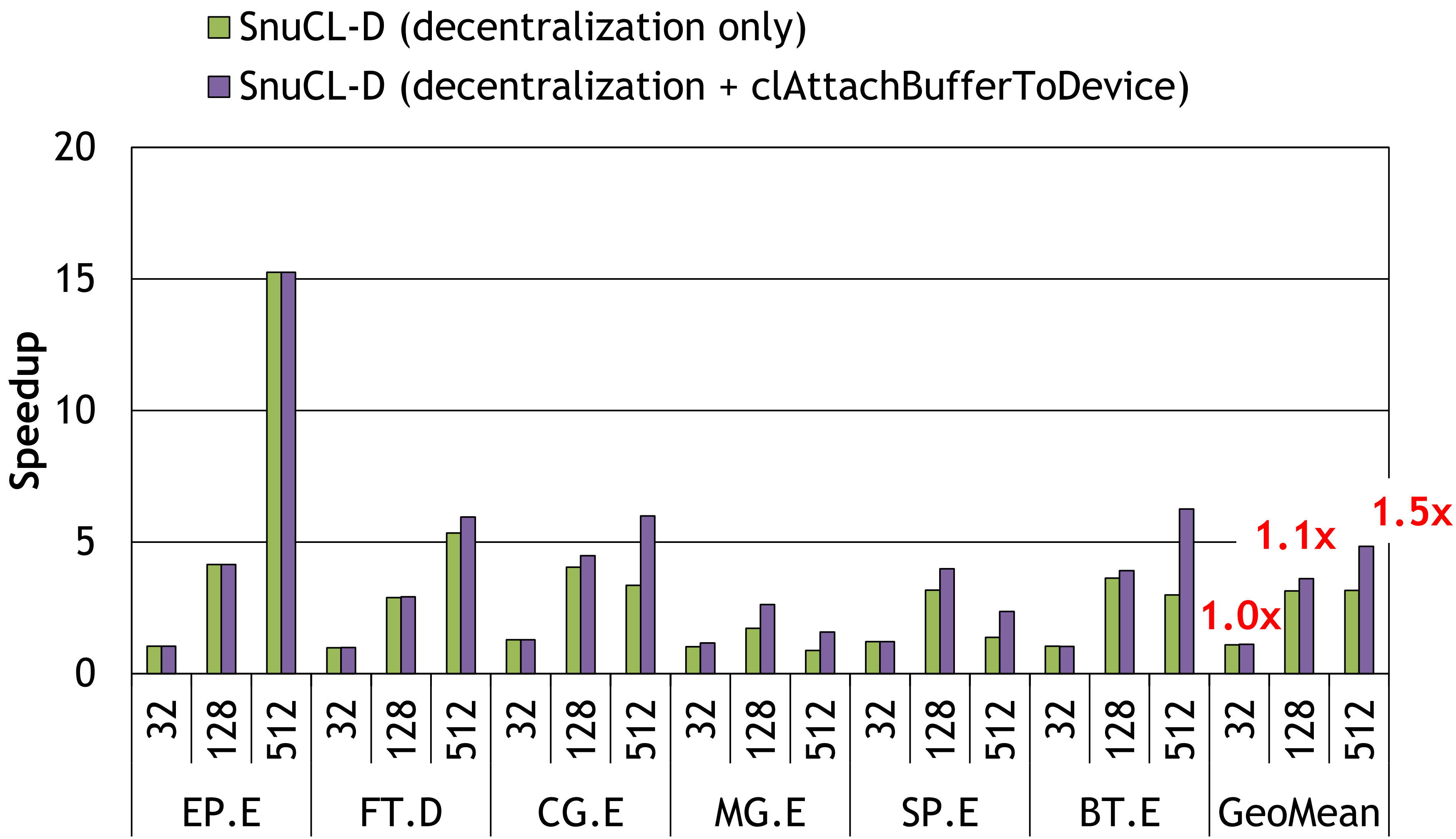
```
void clAttachBufferToDevice(cl_mem m, cl_device_id d);
```

- In OpenCL,
  - Memory object ***m*** is not bound to any devices
  - Scheduling overhead
    - Need to maintain latest device lists (consistency management)
- If this function is called,
  - SnuCL-D assumes
    - ***d*** always has the latest copy of ***m***
  - Scheduling overhead reduced
    - No need to maintain latest device lists

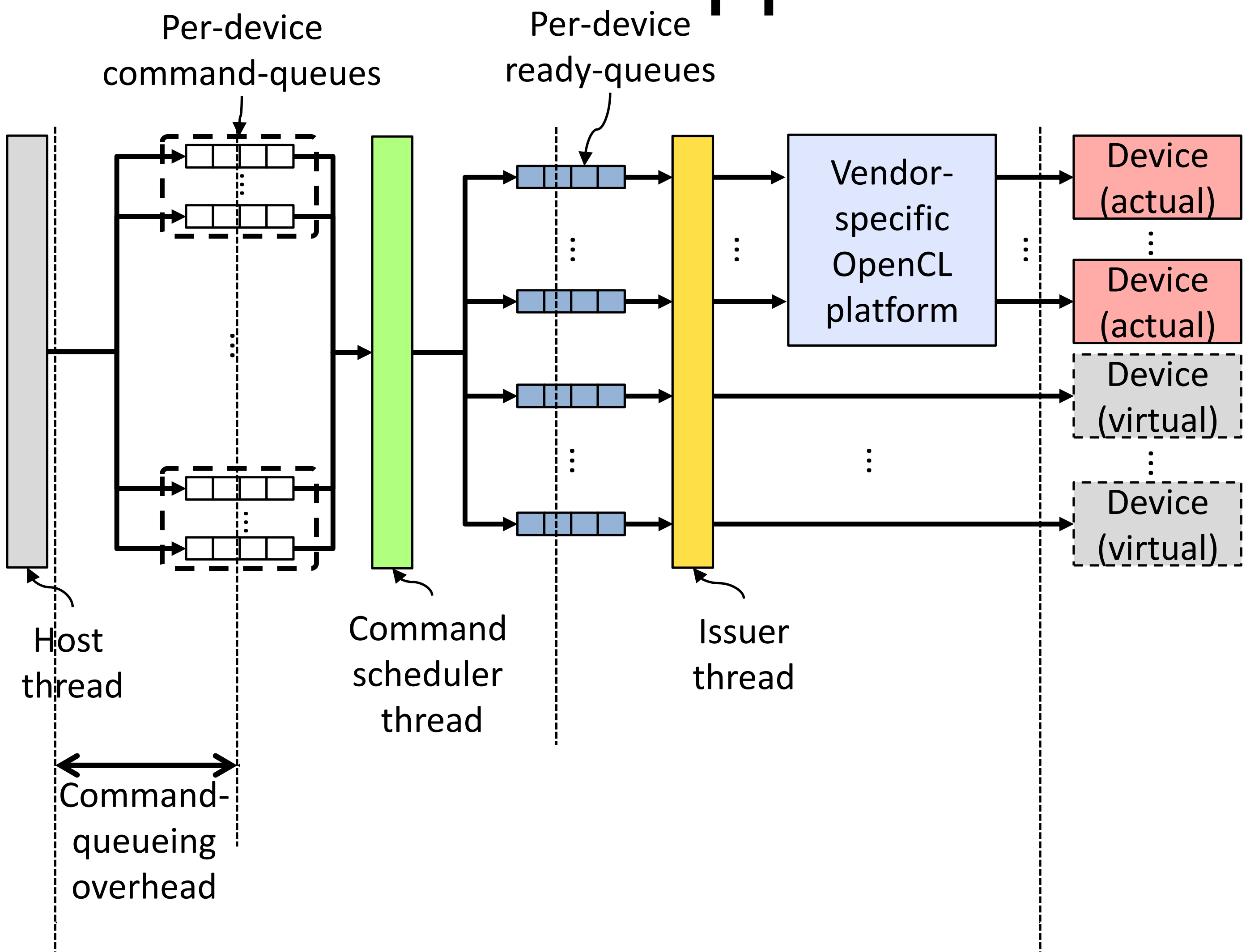
Runtime Overheads of Centralized Approaches



Performance



Runtime Overheads of Centralized Approaches



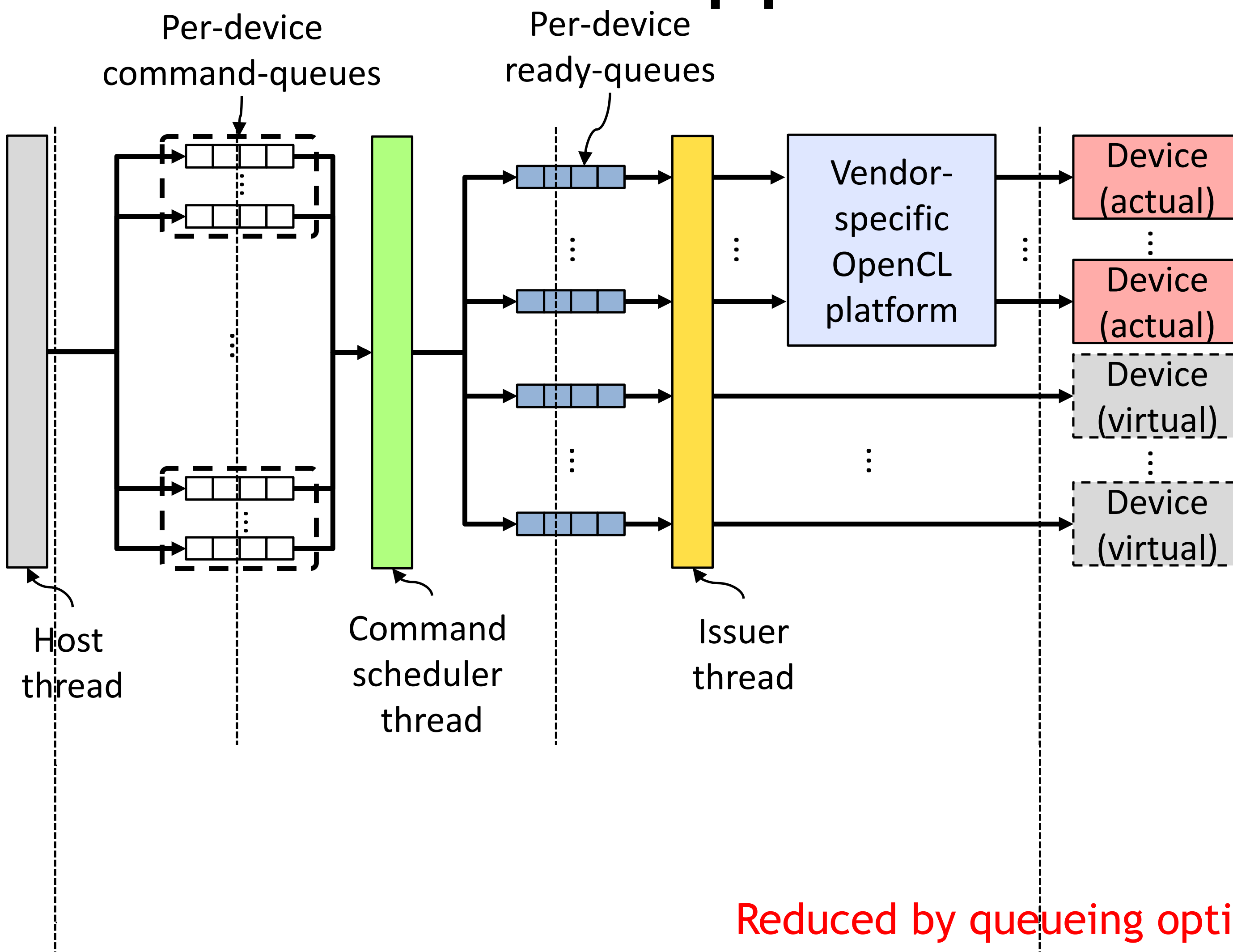
Queueing Optimization

- Two conditions under which commands for a virtual device do not need to be enqueued
  - No events in the event wait list
  - Each memory object is attached

- If the two conditions are met by a command
  - Discarding it does not affect correctness

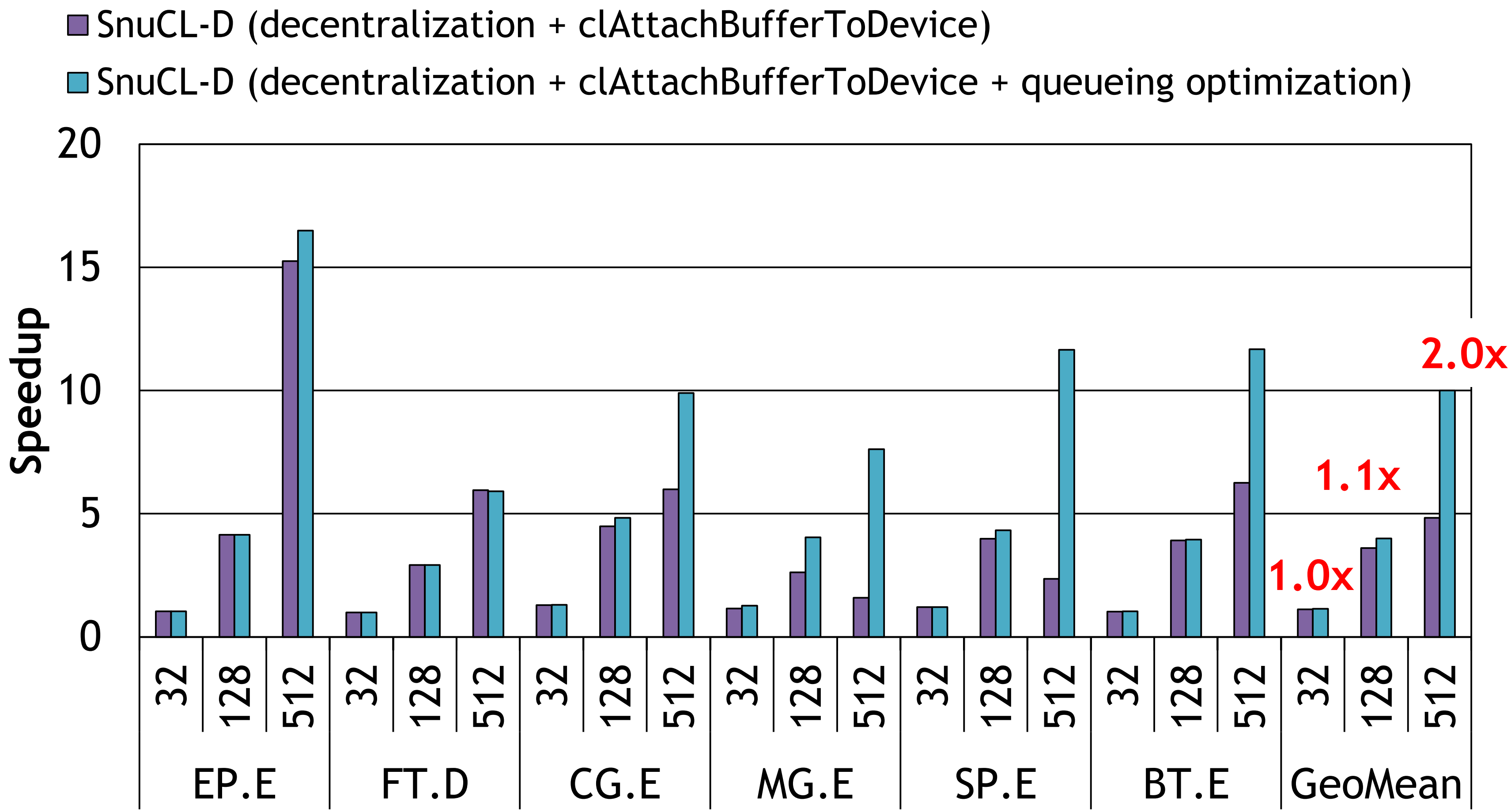
- The commands can be safely discarded when enqueued

Runtime Overheads of Centralized Approaches

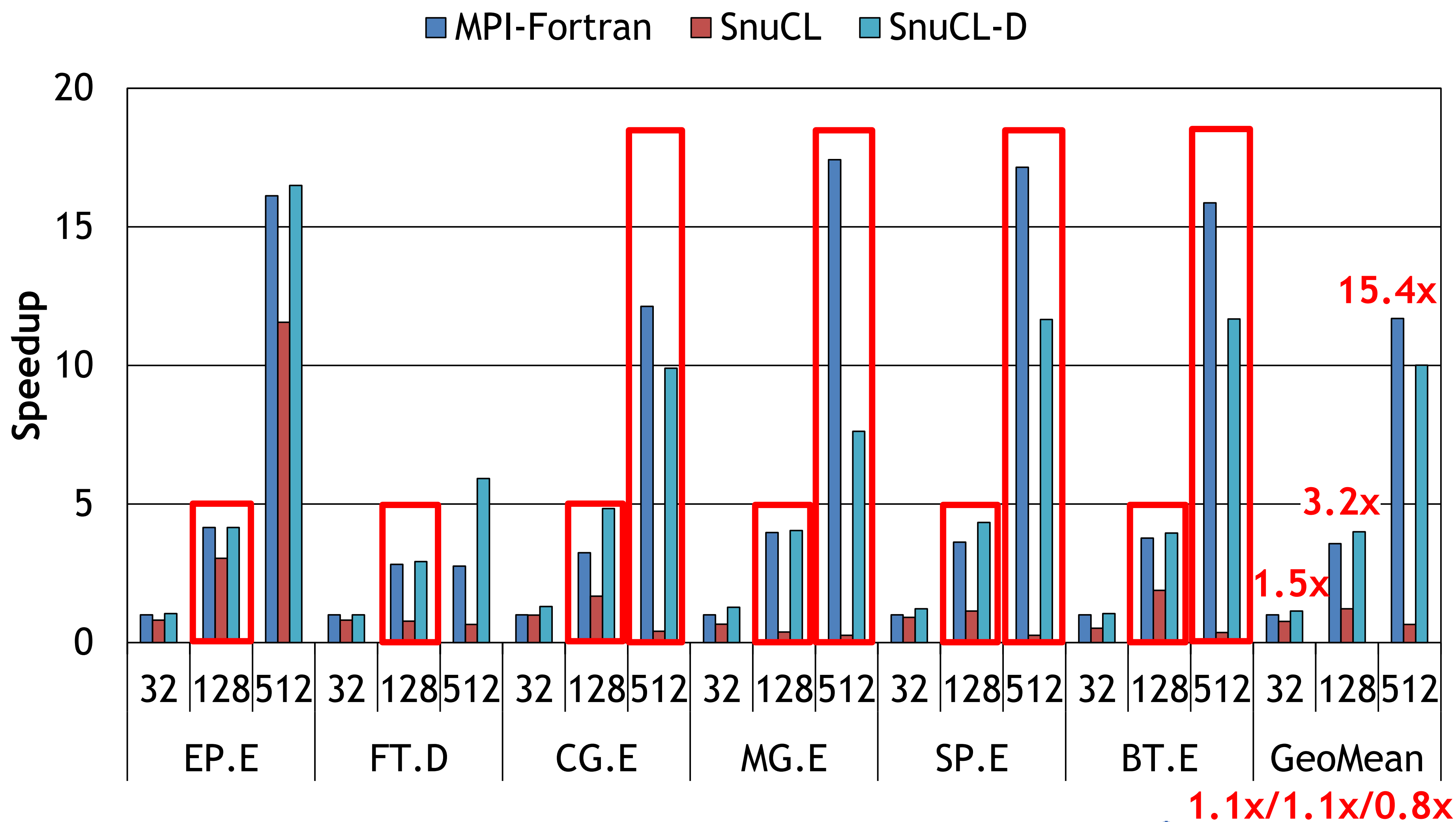


Reduced by queueing optimization

Performance



Performance



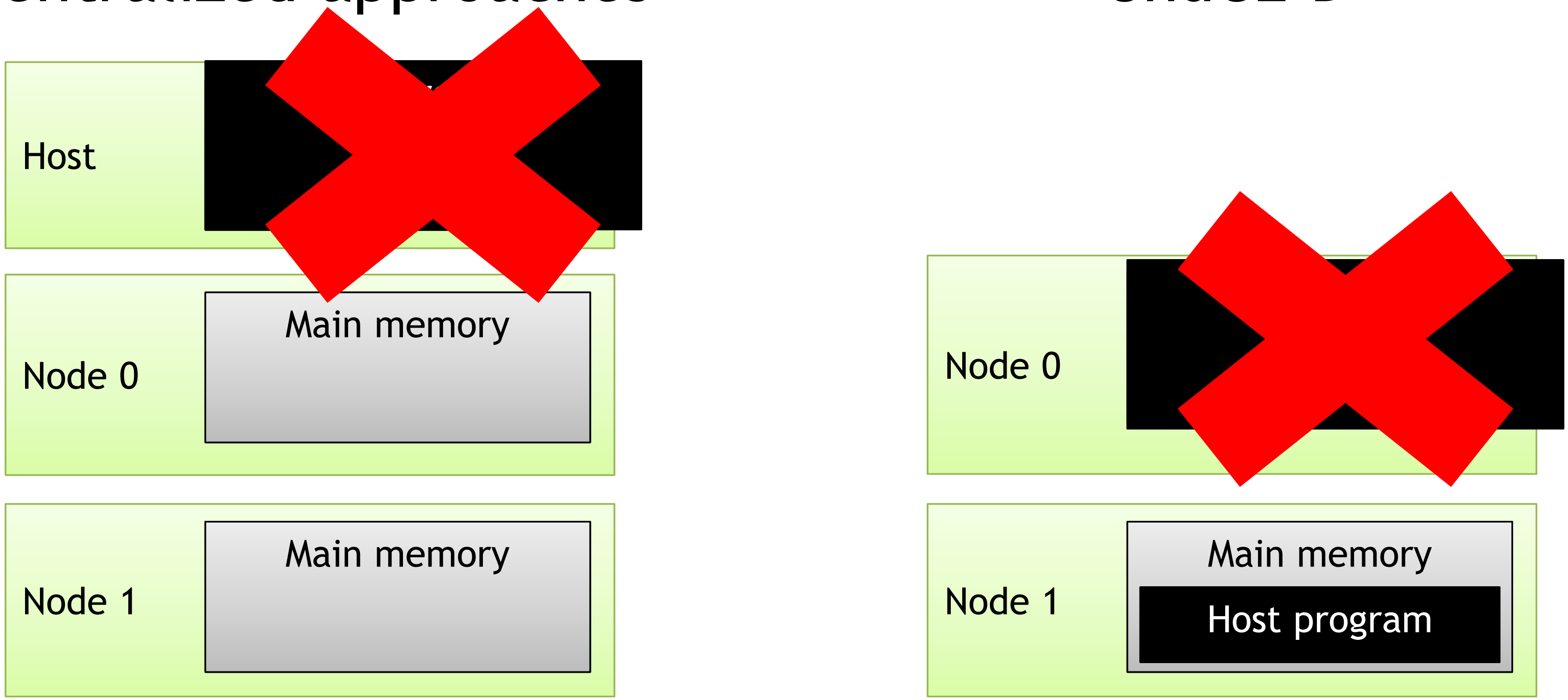
Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

Limitation: Memory Footprint

Centralized approaches

SnuCL-D



Outline

- OpenCL programming model
- Previous approaches for clusters
- Overview of SnuCL-D
- Correctness problems
- Optimization techniques
- Limitation
- Conclusion

Conclusion

- SnuCL-D
  - A scalable and distributed OpenCL framework for clusters
  - OpenCL programs for multiple devices
    - Efficiently executed on a large-scale cluster
- Correctness Problems
  - Consistency problem and non-determinacy problem
- Three optimization techniques
  - Decentralization, new API function, and queueing optimization
- Available at http://snucl.snu.ac.kr
  - July 11, 2016

Thank you