LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

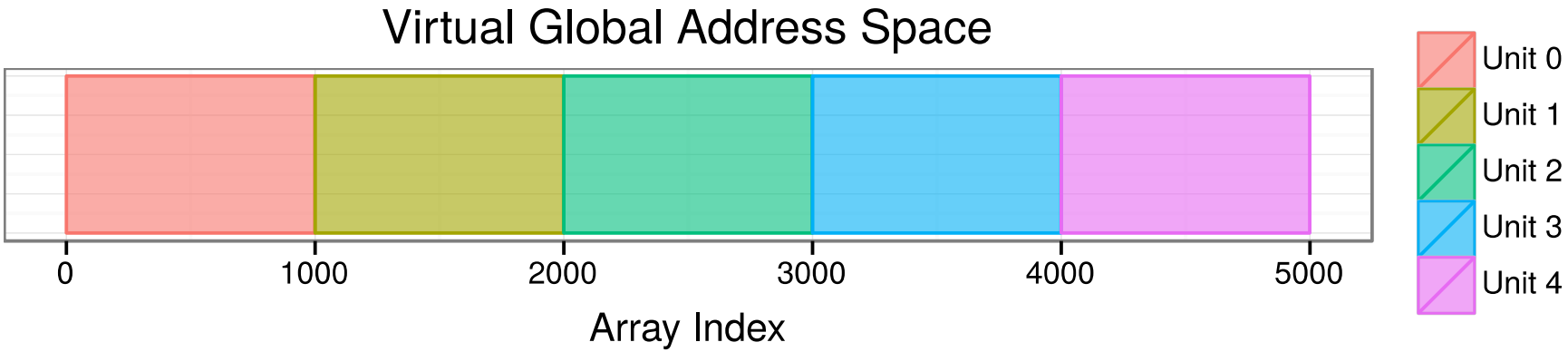
The DASH Coarray Abstraction

www.dash-project.org

Felix Mößbauer
felix.moessbauer@campus.lmu.de
Ludwig-Maximilians-Universität München, MNM-Team

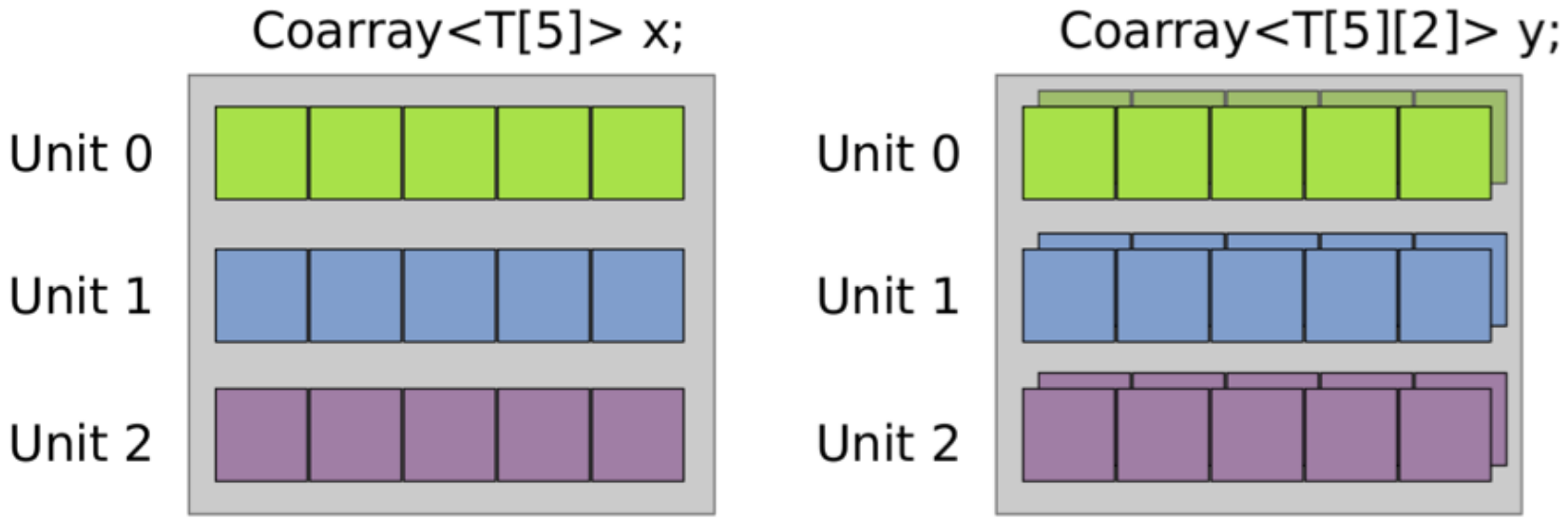
DASH - Overview

- PGAS
  - Data is distributed over units
  - Global data structures know its layout
- DASH
  - C++ library similar to UPC++
  - Implements PGAS approach
  - Provides distributed data structures and algorithms



Coarray - Overview

- Implements PGAS programming model
  - Part of Coarray Fortran (CAF 2008)
- CAF program executed as loosely coupled set of processes, following SPMD
- Extends array syntax with a remote dimension



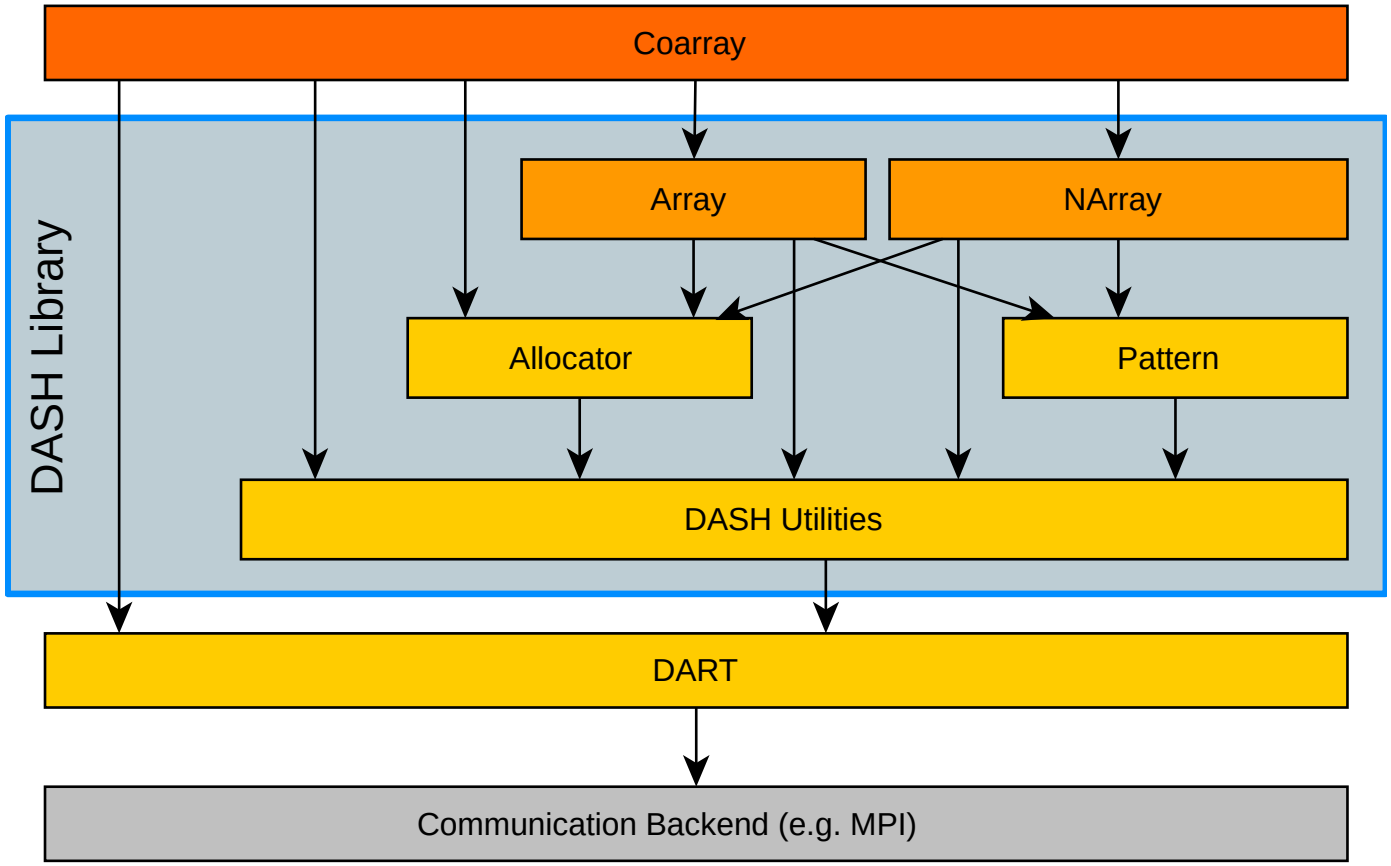
Coarray – Design Goals

- Interface similar to CAF
- Good integration with DASH + STL concepts
- Easy to use
- Reasonable performance

Is it useful?

- Yes, use rich C++ features
- Yes, easy integration

Coarray System Diagram



Arithmetic Operations on scalar Coarray

- Idea: Use scalar Coarray as number
  - Requires blocking gets
  - Globally overloading all arith. operators
- Scenarios
  - T + Coarray<T>
  - Coarray<T> + T
  - Coarray<T> + Coarray<T>
- Non-blocking put for performance

```
Coarray<int> x;
x += 1;
x = x(1) + 1;
x = 1 + x(1);
```

DASH Coarray Interface

- Implements container concept
- Provides linear access
  - To local and global elements
  - By using iterators

```
dash::CoArray<int[10][20]> x;

// global iterators to full range
GlobIter<int> gbegin = x.begin();
GlobIter<int> gend   = x.end();
// global iterator to range on single unit (might not be local)
GlobIter<int> gbegin = x(0).begin();
GlobIter<int> gbegin = x(0).end();
// local iterator to local range. Can be implemented using pointers
local_iterator<int> lbegin = x.lbegin();
local_iterator<int> lend   = x.lend();
```

Additional Structures

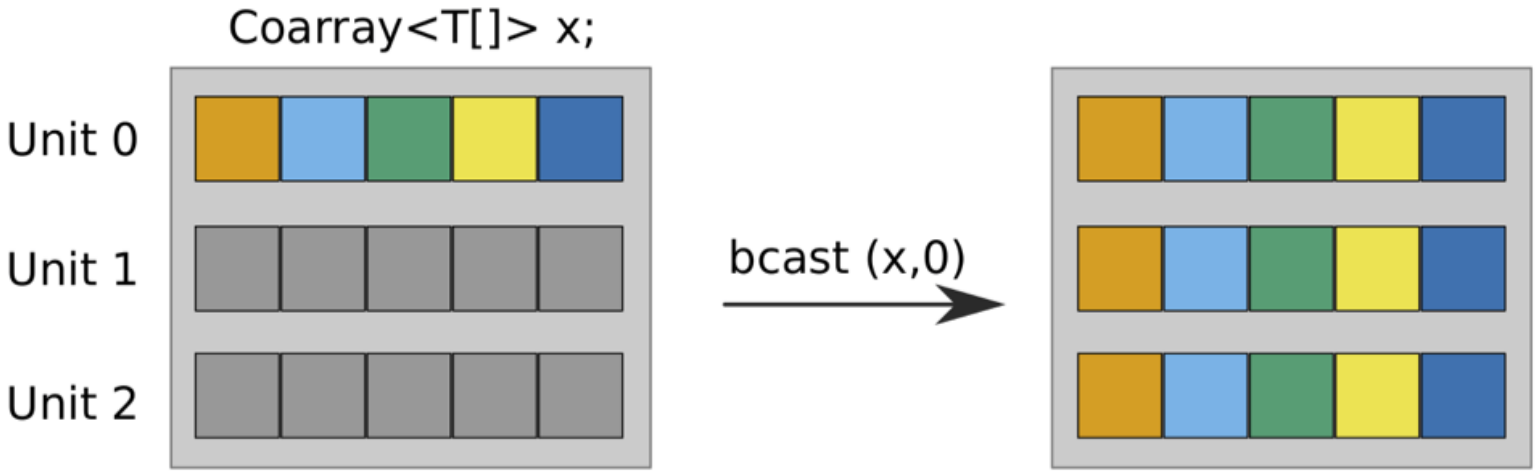
- Copointers => Implemented as GlobPtr
  - GlobPtr can be placed in Coarray
  - Point to arbitrary location in GAS
- Synchronization mechanisms
  - sync_all()
  - sync_images(list_of_images)
- Atomics
- Collective operations
- Comutex, Coevent
- Helpers for compound types

Atomic Operations

- Coarray<Atomic<int>[10]>
- Provides atomic access on elements, e.g
  - load, store
  - add, fetch_and_op
  - compare_exchange
- Implemented by specializing GlobRef<atomic<T>>
- Important: No local access using pointers

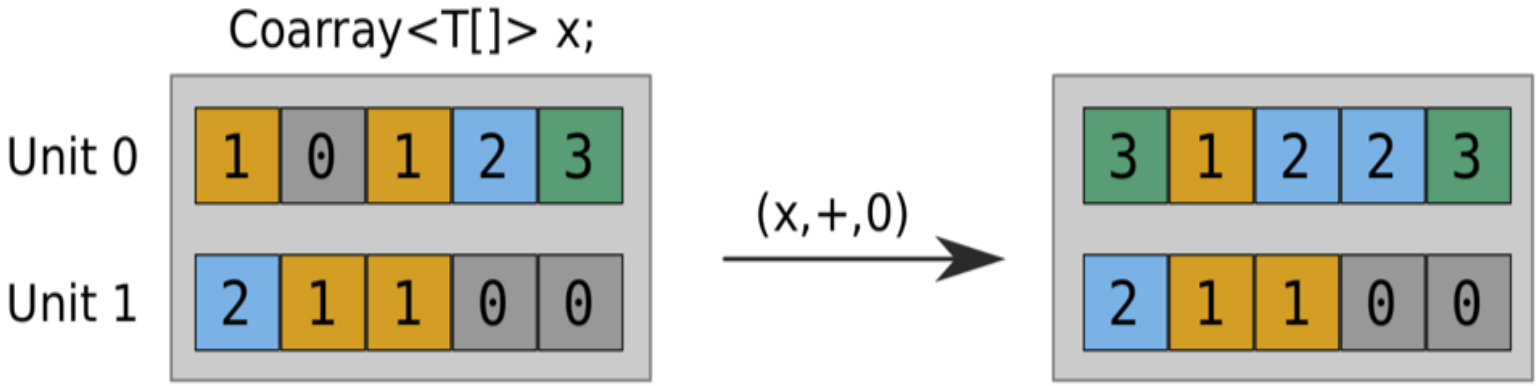
Collective Operations – Coarray Interface

- Broadcast (local element(s) to all units)
- Reduce (broadside reduction of local parts)



Collective Operations – Coarray Interface

- Broadcast (local element(s) to all units)
- Reduce (broadside reduction of local parts)



Collective Operations – Container Interface

- Supports all DASH + STL algorithms
  - copy, foreach, min, max, …
- Supports arbitrary ranges by using iterators
- No “range-references”

```
dash::Coarray<int[10][20]> x;
dash::fill(x.begin(), x.end(), 2);
dash::for_each(x(1).begin(), x(3).end(),...);
std::fill(x.lbegin(), x.lend(), 2);
```

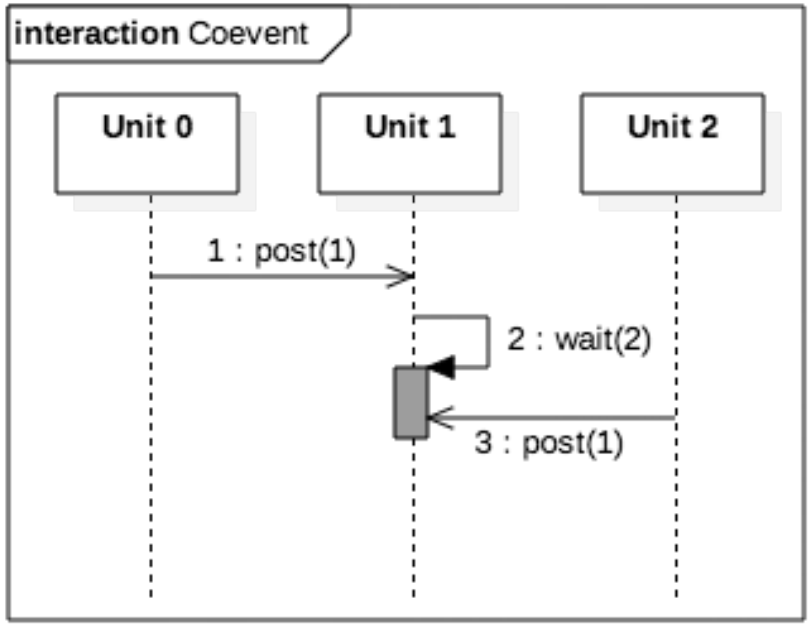
Comutex

- Mutual exclusion on per unit basis
- Compatible with std::lock_guard
- Internally uses dash::Mutex

```
Comutex comx;
{
    int unit = 1; // in [0,num_images()-1]
    std::lock_guard<dash::Mutex> lg(comx(unit));
    // do something critical
}
```

Coevent - Interface

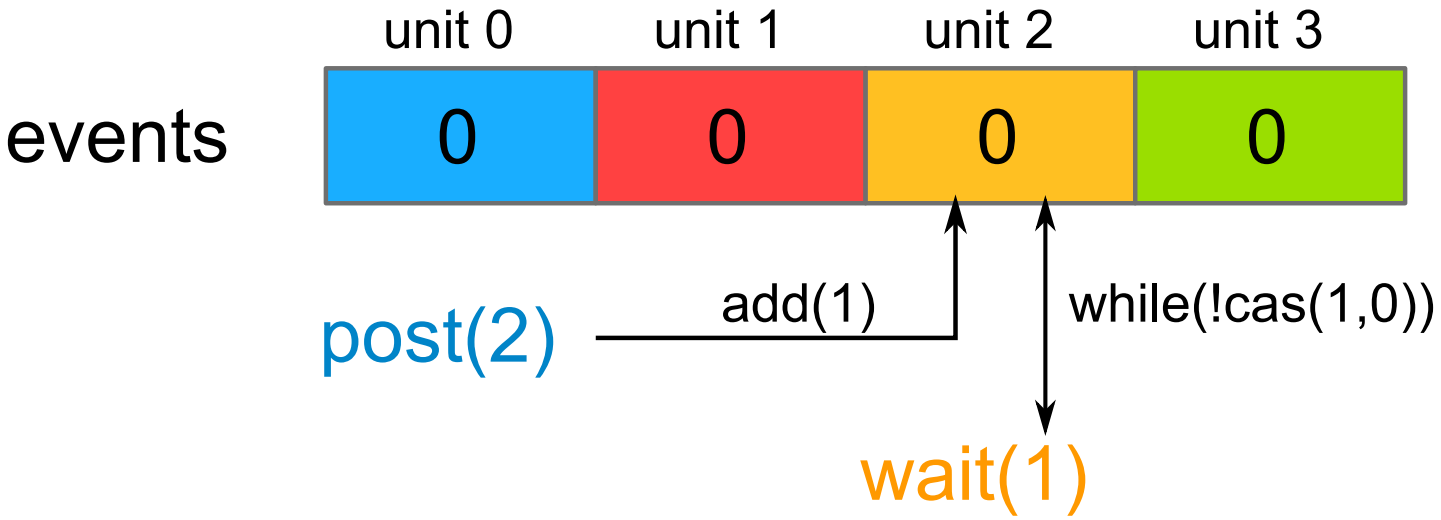
- Similar to proposed CAF Events
- Simplistic P2P synchronization
- Post(target) Wait(num)



Highly Scalable Ref. Implementation

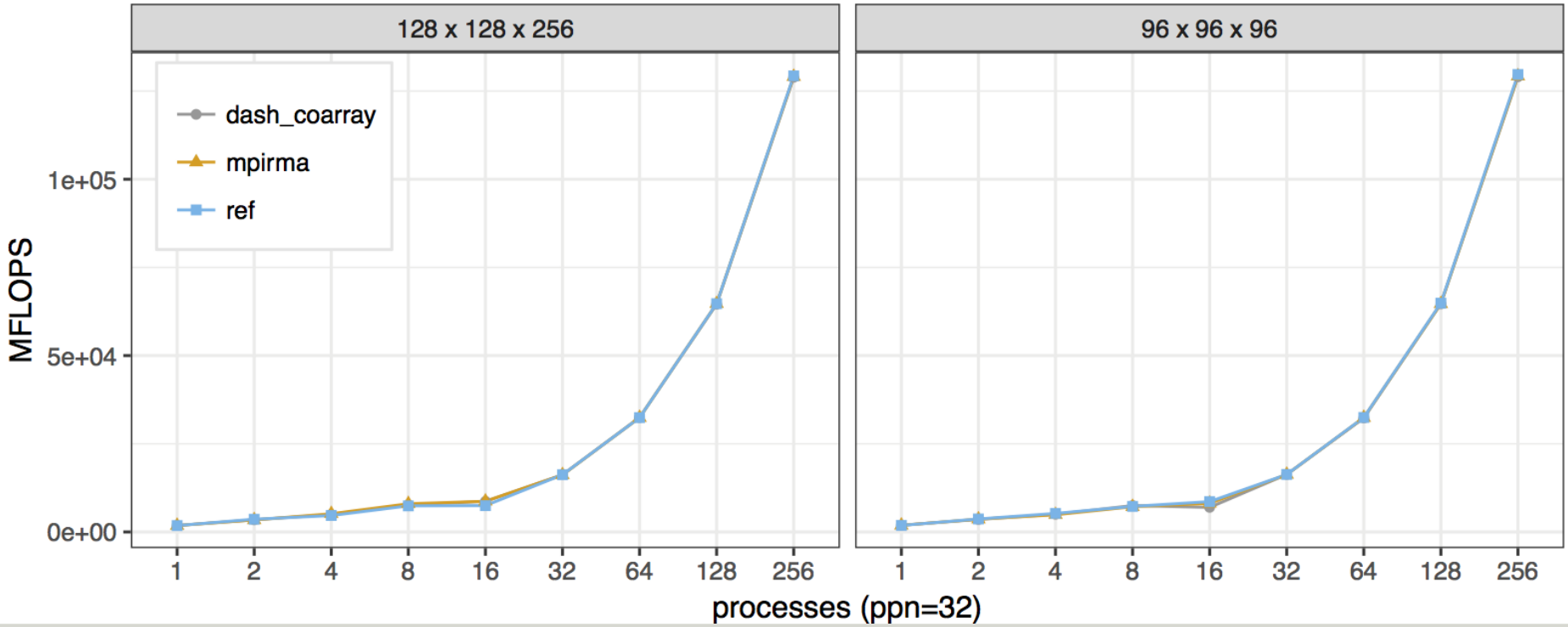
CoEvent - Implementation

- Pure one-sided communication
- Waiting by spinning on atomics



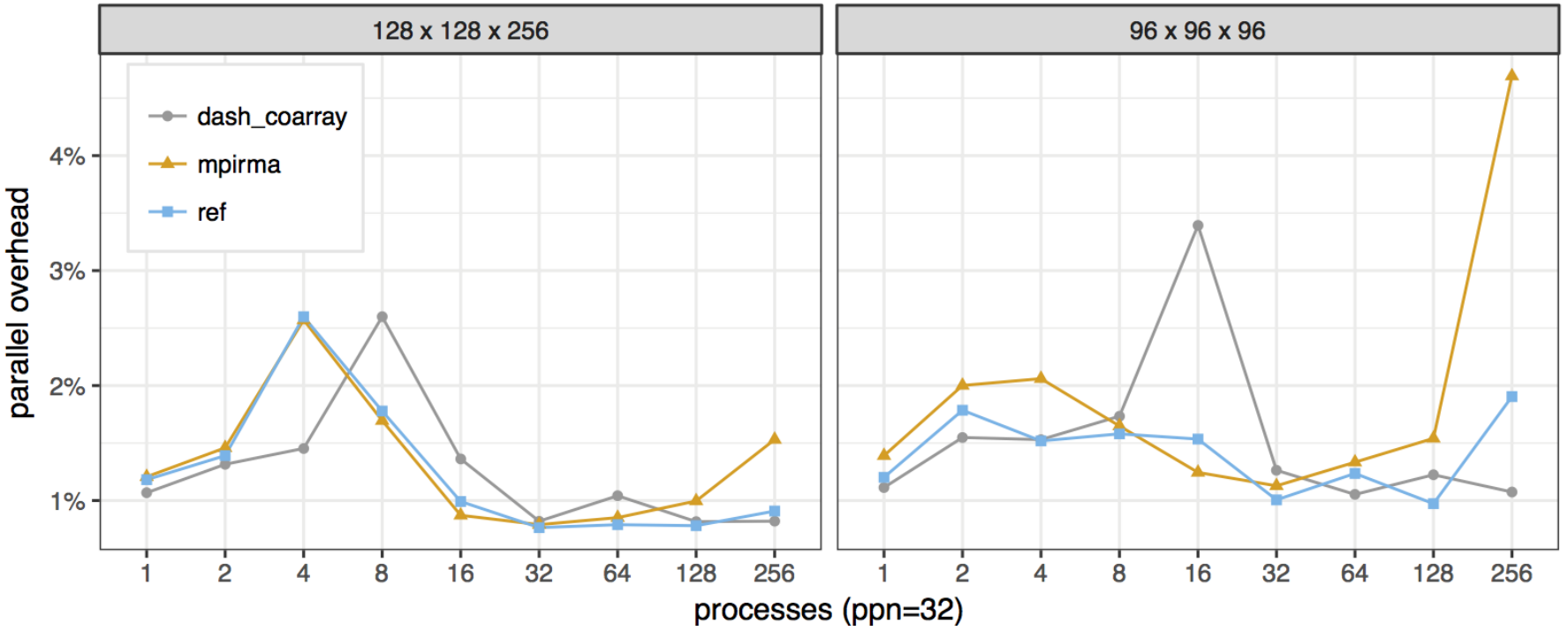
Evaluation

- HPCCG Benchmark (Cori - HW)
  - CG solver on 3D chimney domain
  - No communication / computation overlap
  - Coevent for signalling completion of data transfer
  - Blockwise data transfer



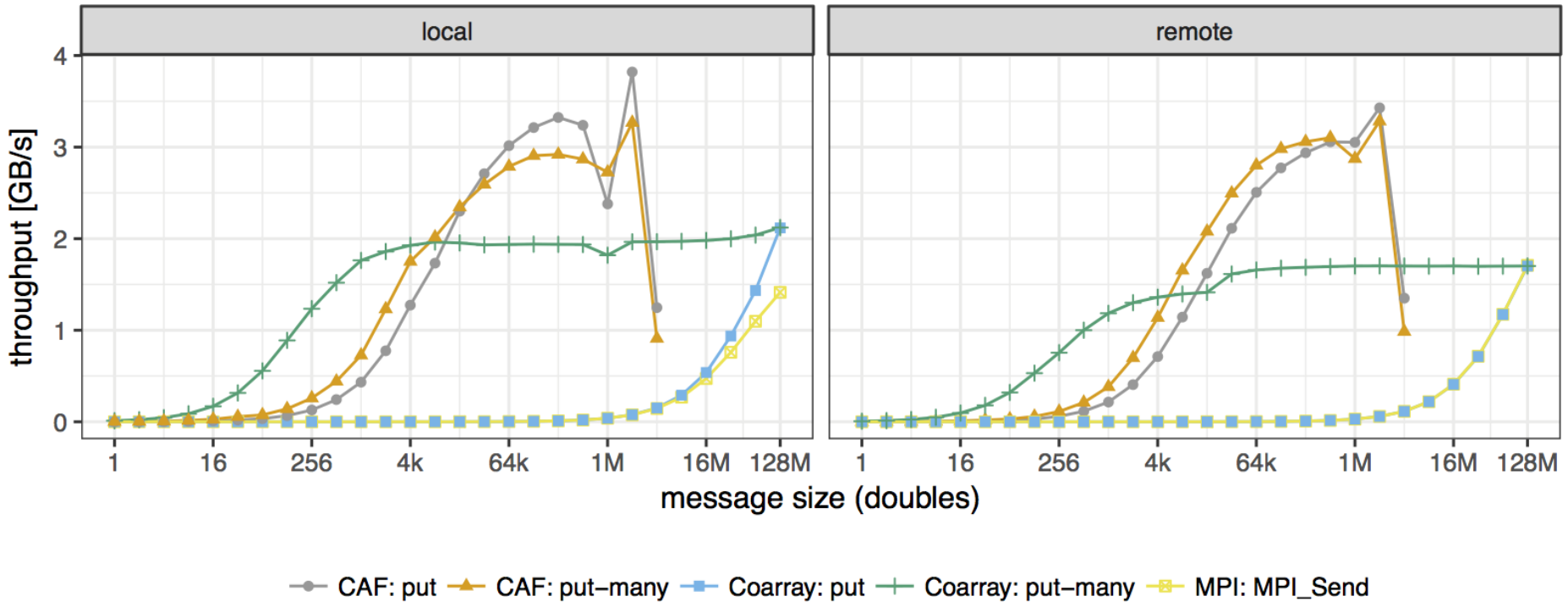
Evaluation

- HPCCG Benchmark (Cori - HW)
  - CG Solver on 3D chimney domain
  - No Communication / Computation overlap
  - Coevent for signalling compl. Of data transfer
  - Blockwise data transfer



Evaluation

- CAF-Bench (SuperMUC - IntelMPI)
  - Simulate communication patterns
  - Single element / ranges | Blocked / Async
  - Local / remote (various locality domains)



Summary

- CAF interface as C++ library
- Many shortcomings of CAF fixed
- Good integration in DASH
- Highly scalable with good performance

- Major challenges during the project
  - Full-stack guarantees (memory, progress)
  - Minimizing overhead

Acknowledgements

DASH on GitHub:
https://github.com/dash-project/dash

Coarray part of release 0.3.0
Paper accepted at PDP2018

- Funding

DFG Deutsche Forschungsgemeinschaft

Bundesministerium für Bildung und Forschung

- The DASH Team

  T. Fuchs (LMU), R. Kowalewski (LMU), D. Hünich (TUD), A. Knüpfer (TUD), J. Gracia (HLRS), C. Glass (HLRS), H. Zhou (HLRS), K. Idrees (HLRS), J. Schuchart (HLRS), F. Mößbauer (LMU), K. Fürlinger (LMU)