https://astra-sim.github.io

https://github.com/mlcommons/chakra

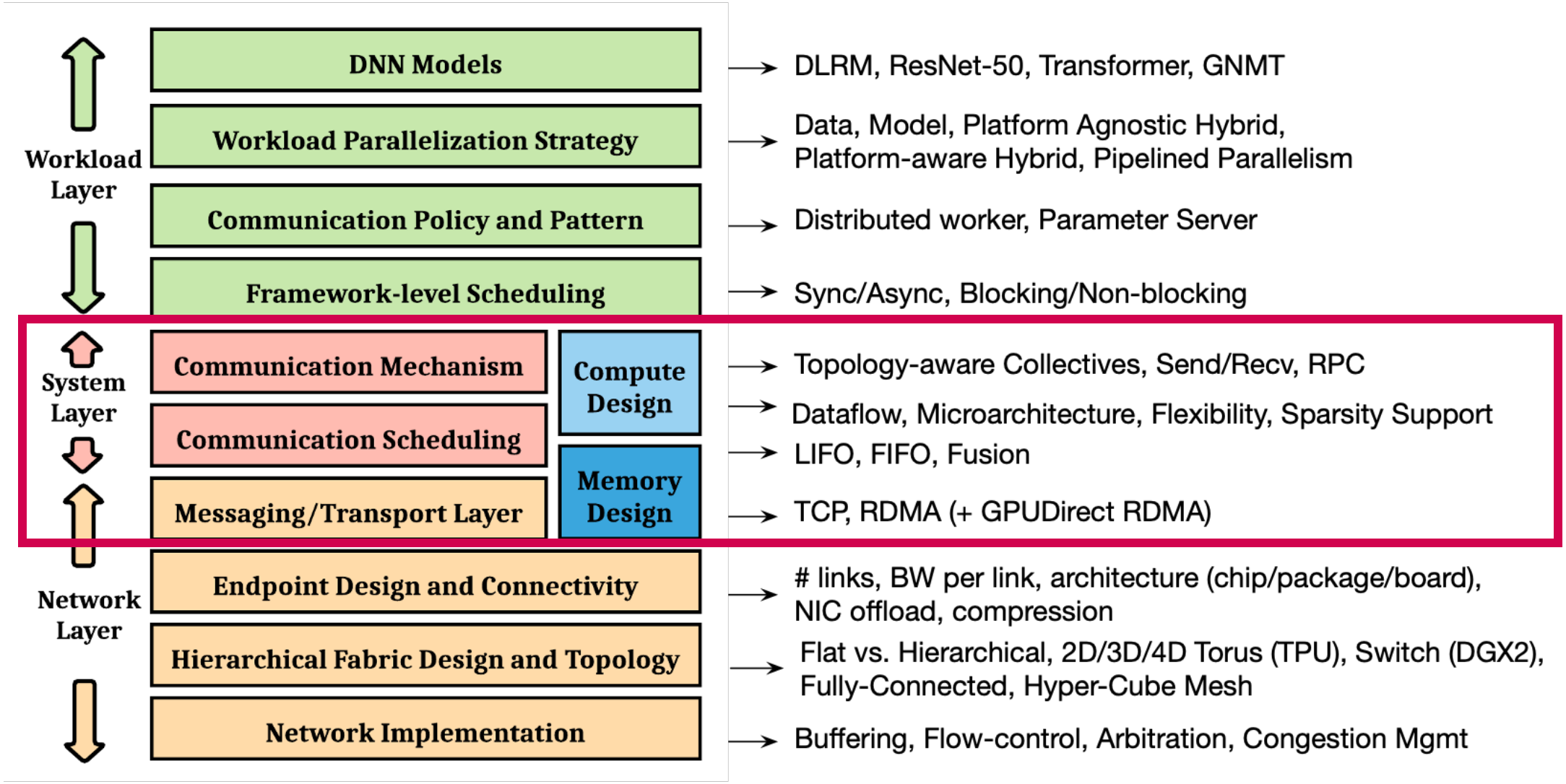
ASTRA-sim Tutorial
MICRO 2024
Nov 3, 2024

ASTRA-sim and Chakra Tutorial:
System Layer

Will Won
Ph.D. Candidate
School of CS, Georgia Institute of Technology
william.won@gatech.edu

Slide courtesy: Saeed Rashidi
<rashidi1saeid@gmail.com>

Design Space of Distributed ML

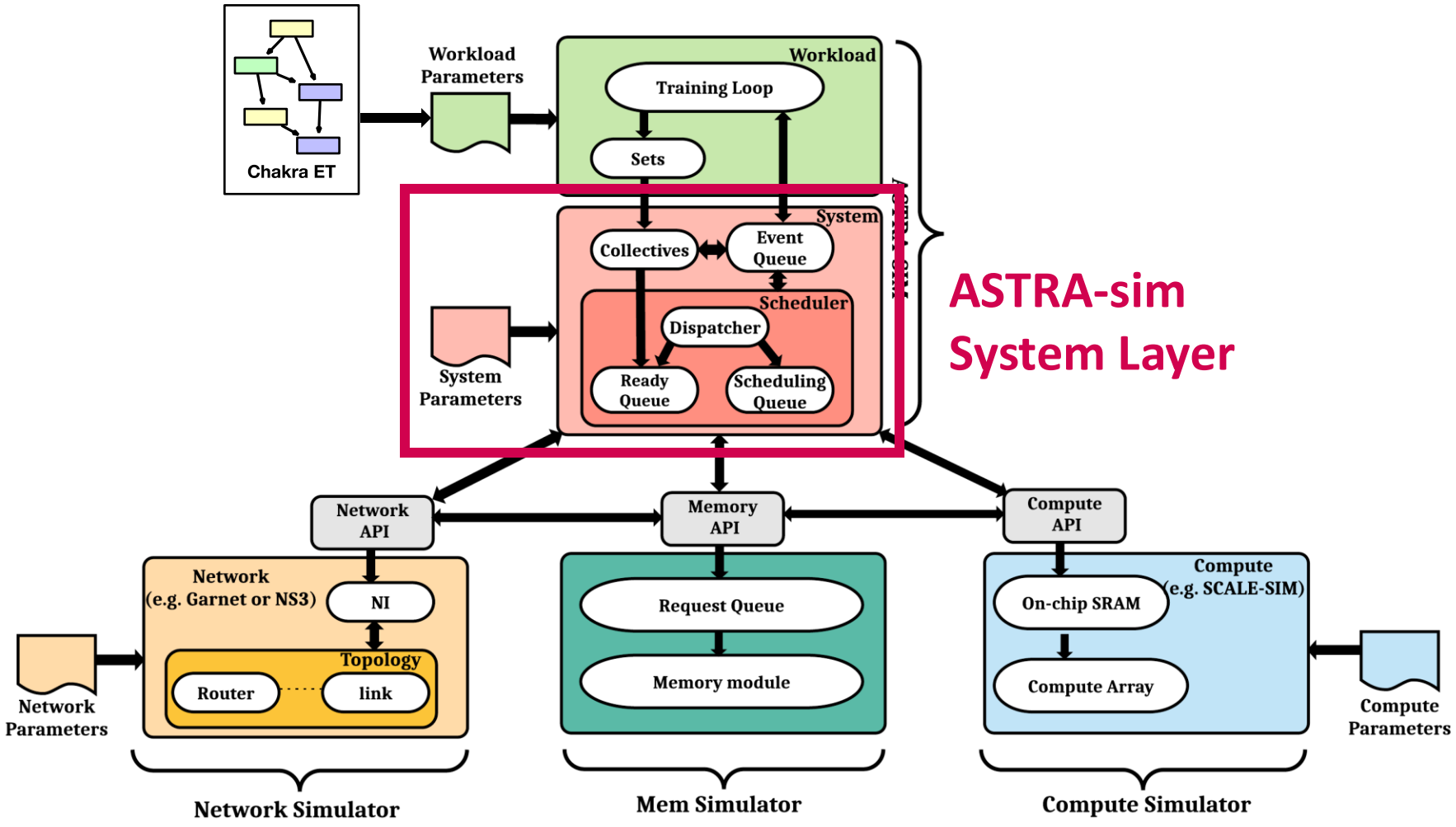


System Layer

- Manages intra- and inter- **collective scheduling**
- Implements textbook (basic) **collective communication algorithms**
- **Dispatches actual send/recv requests** to the network layer

- Mimics the **Collective Communication Libraries (CCLs)**
  - e.g., NCCL, RCCL, oneCCL

ASTRA-sim: System Layer



Recall: Workload Layer

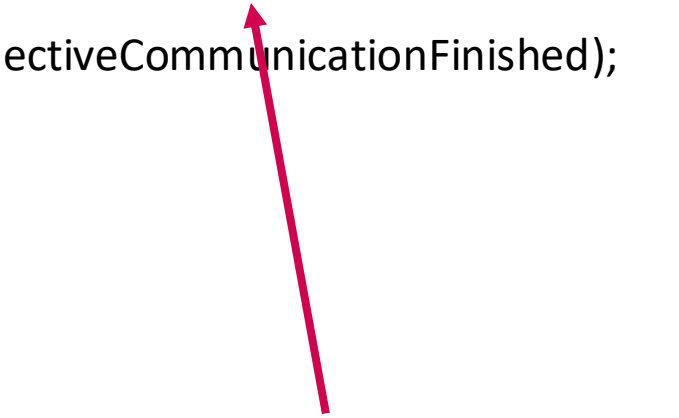
```
void Workload::issue_comm(node) {
        hw_resource->occupy(node);

        if (node->comm_type() == ChakraCollectiveCommType::ALL_REDUCE) {
                DataSet* fp = sys->generate_all_reduce(node->comm_size( ), ...)

                fp->set_notifier(EventType::CollectiveCommunicationFinished);

        }

        (...)
}
```



Managed by the System Layer

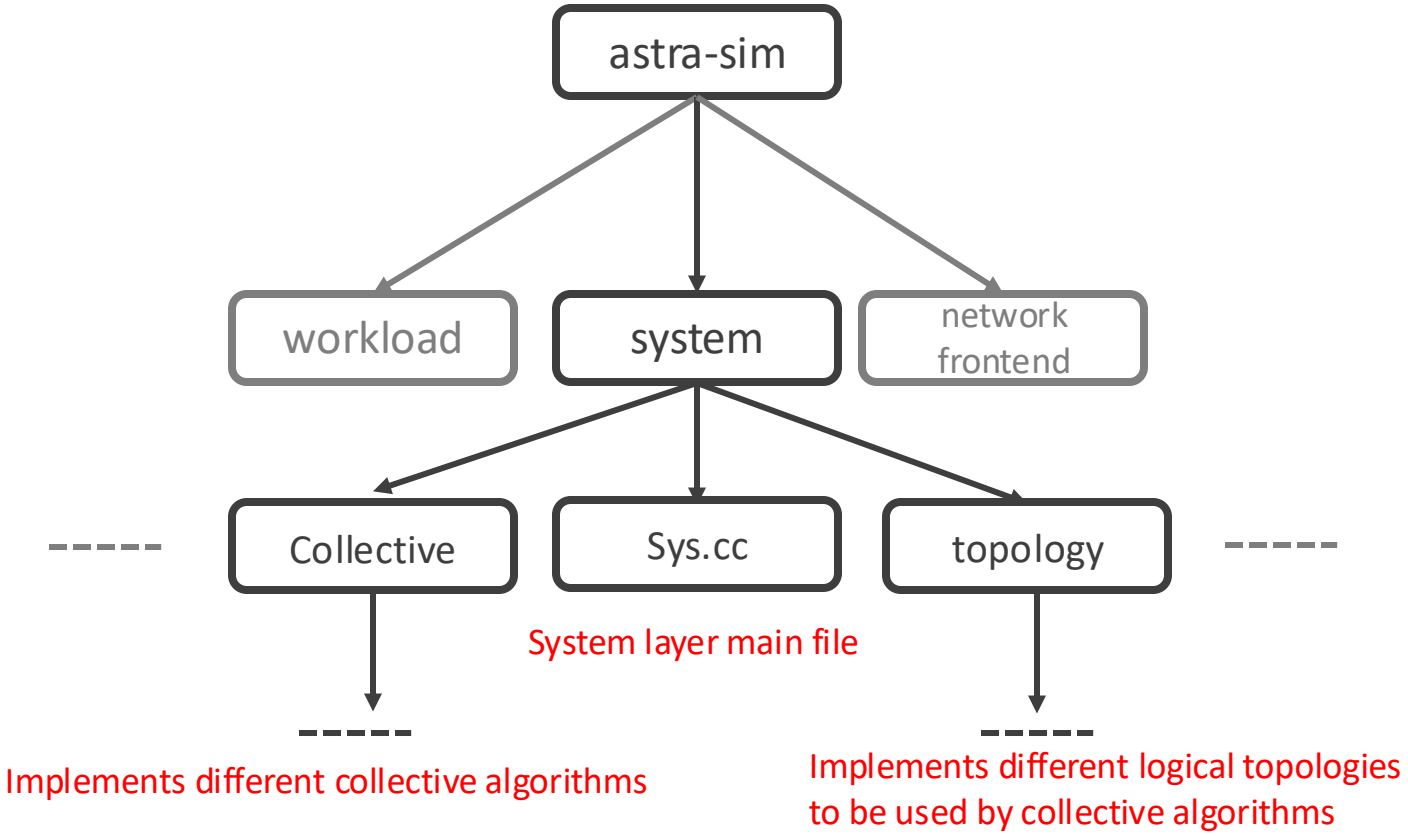
System Layer

- Collective algorithms are implemented using:
  - **Finite State Machine (FSM)**

- Following the textbook algorithm, the system layer:
  - **Dispatch send/recv operation** to the network layer
  - **Update** the FSM State
  - **Wait** for the network recv event handler
  - **Repeat** the process until Done

System Layer Collective Implementation

- Code Structure



Collective Algorithm

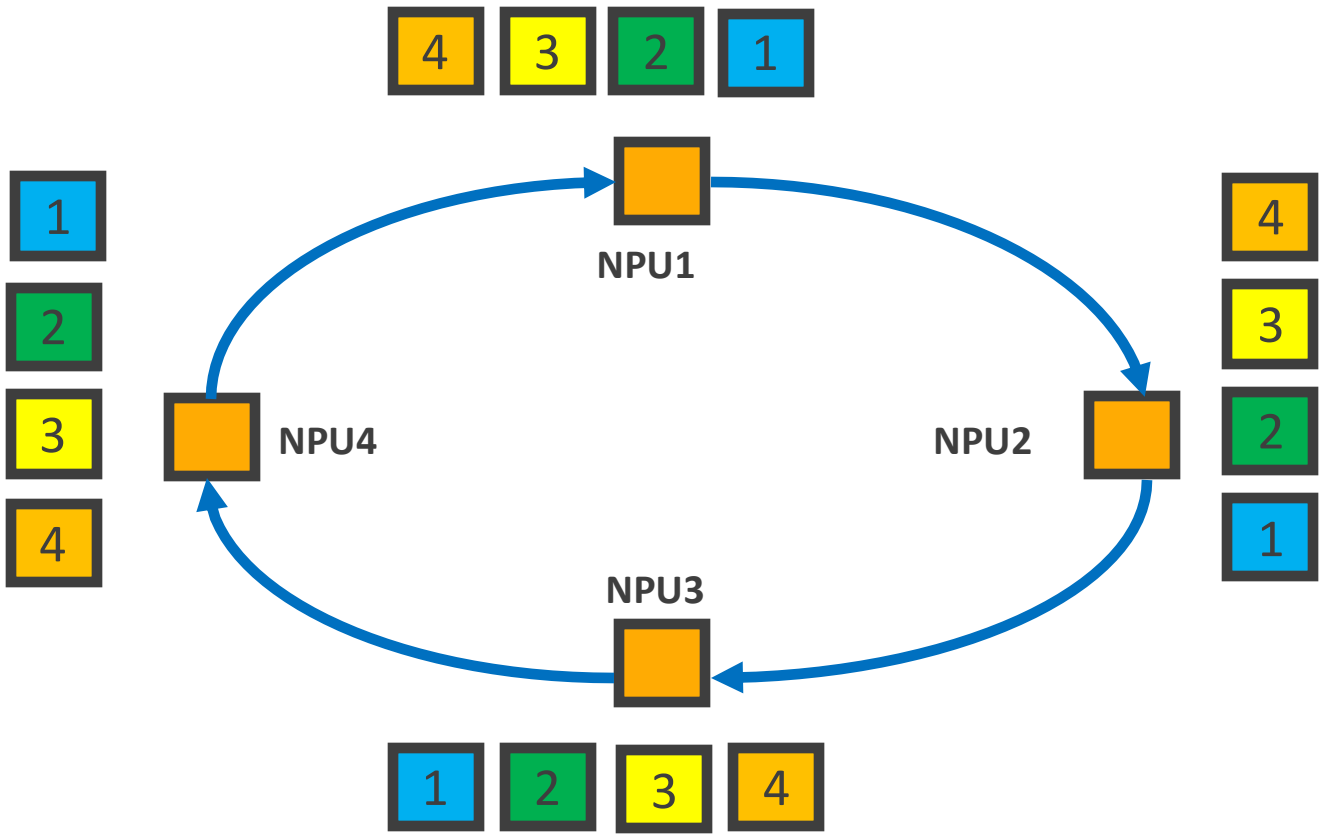
```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring"
    ],
    "all-gather-implementation": [
        "ring"
    ],
    "reduce-scatter-implementation": [
        "ring"
    ],
    "all-to-all-implementation": [
        "ring",
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

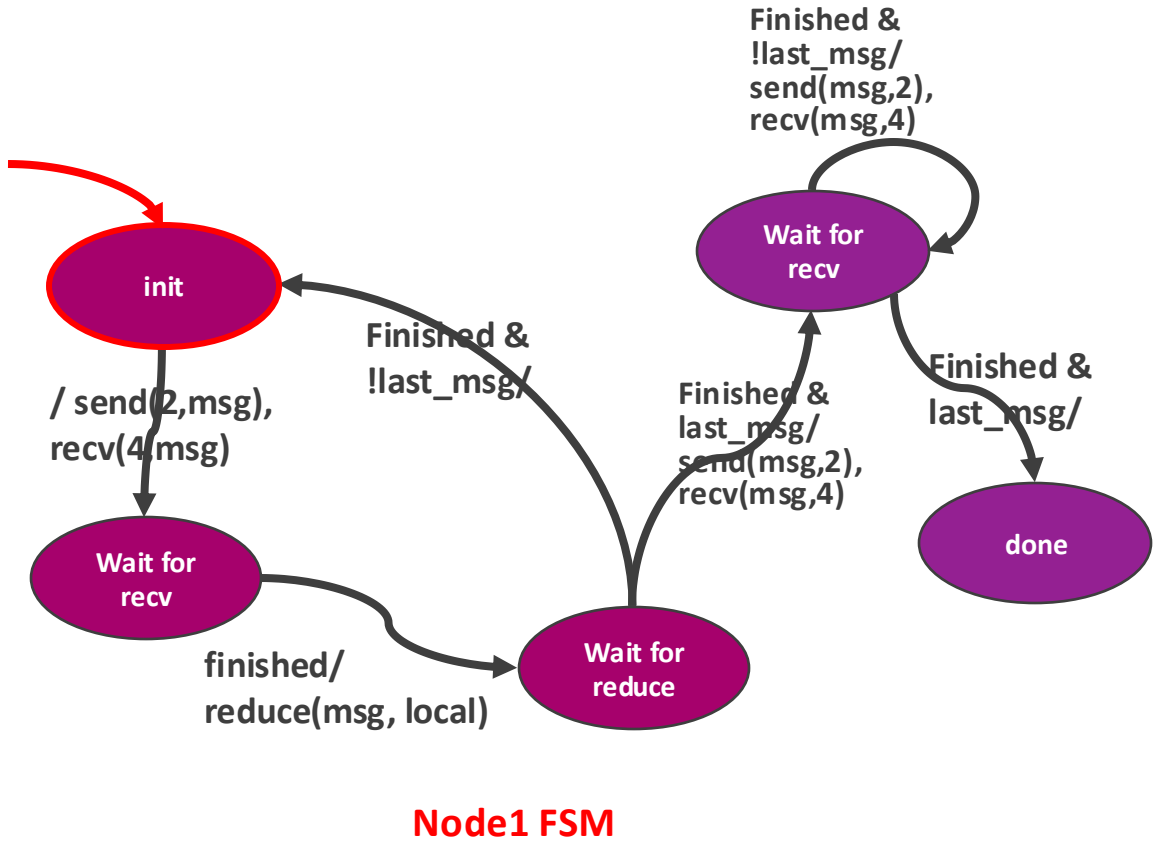
Declares (textbook) collective algorithms to use when running collectives

- Collective Options in ASTRA-sim:
  - ring
  - direct
  - halvingDoubling
  - doubleBinaryTree

System Layer Collective Implementation

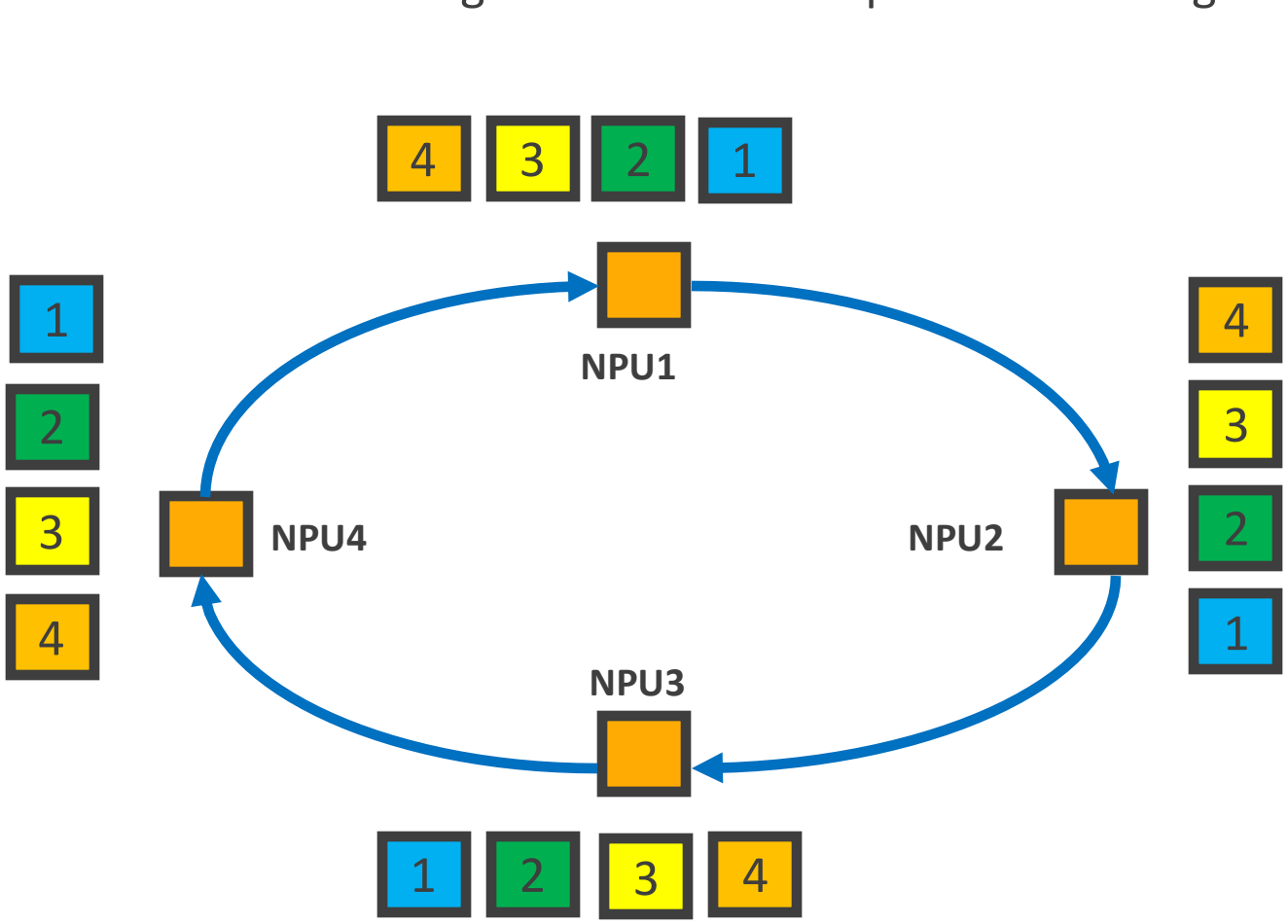
- Collective algorithms can be implemented using **state machines**.

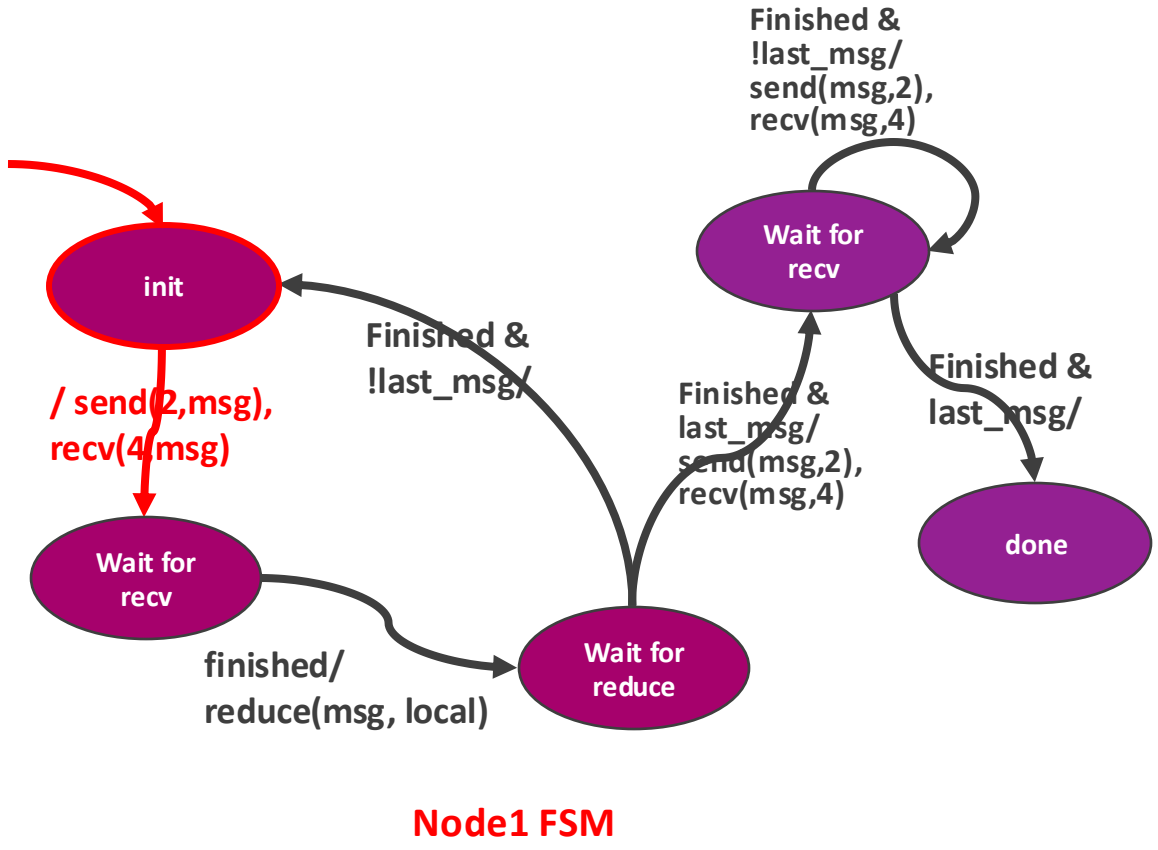




System Layer Collective Implementation

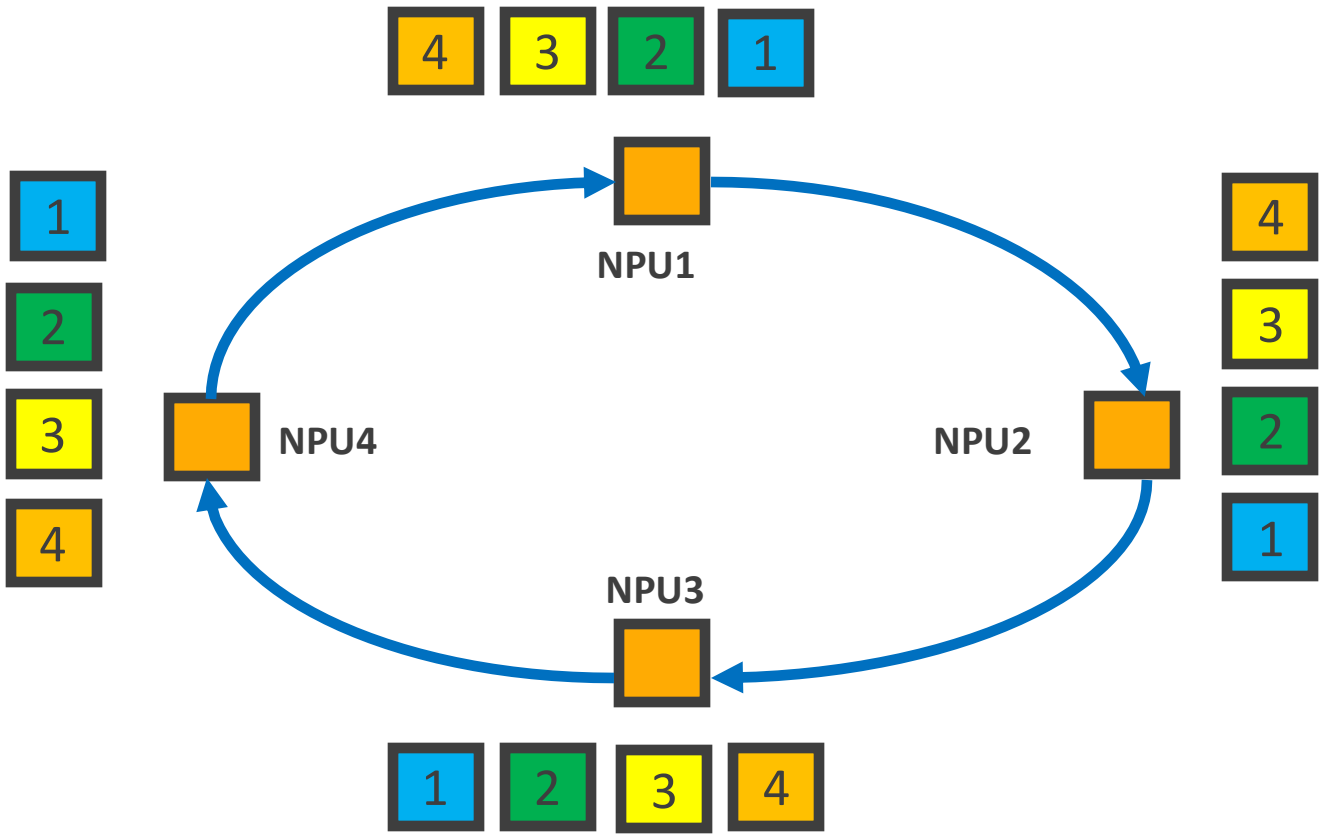
- Collective algorithms can be implemented using **state machines**.

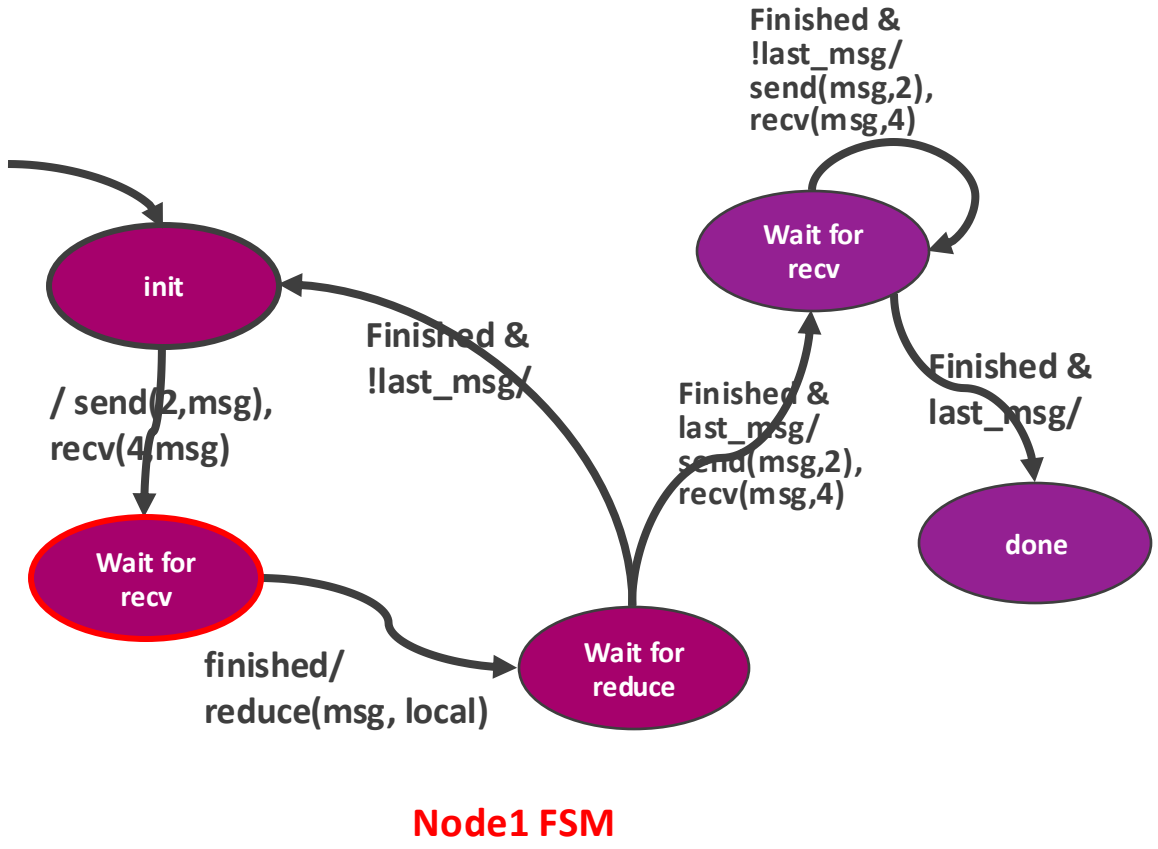




System Layer Collective Implementation

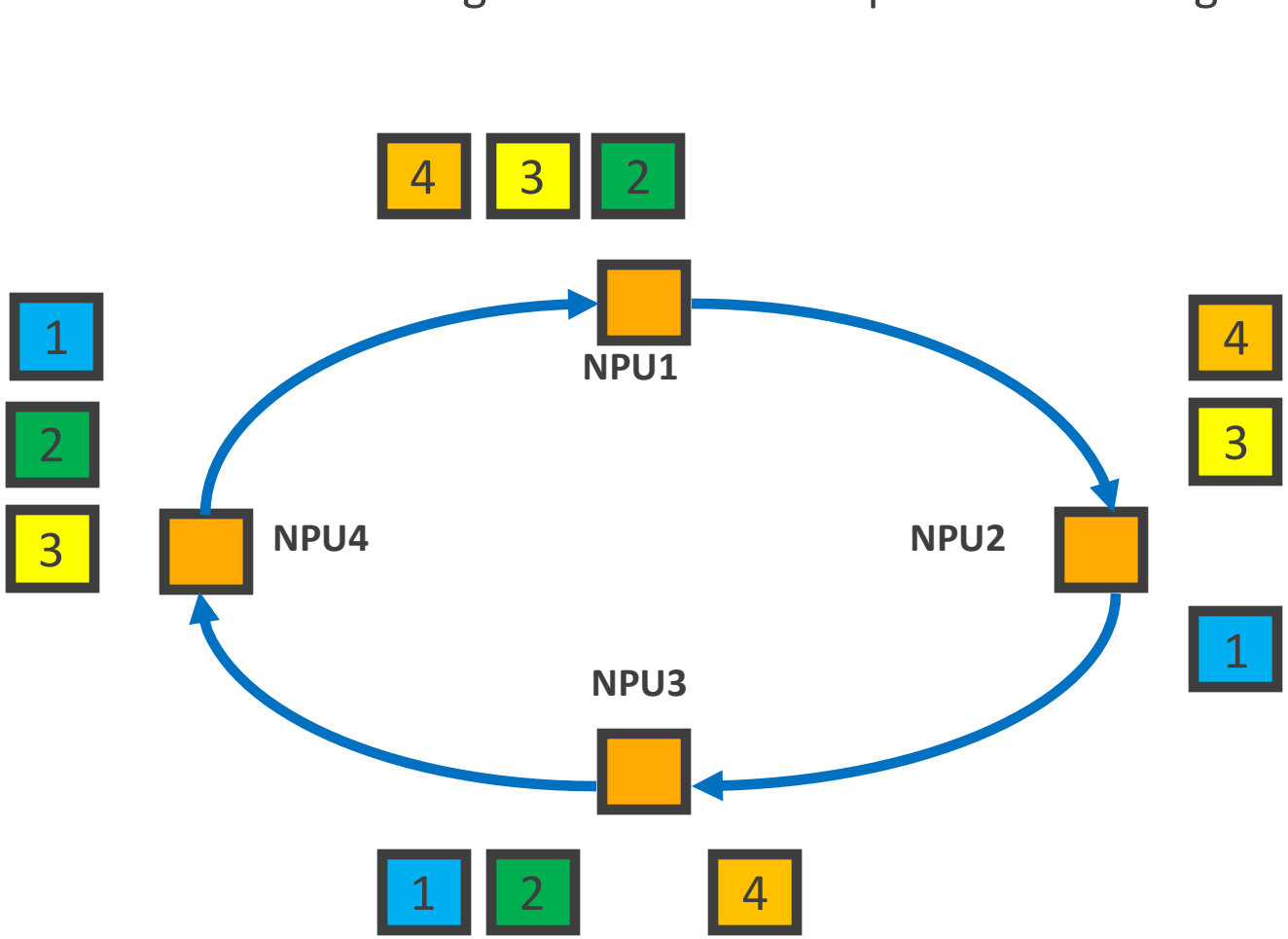
- Collective algorithms can be implemented using **state machines**.

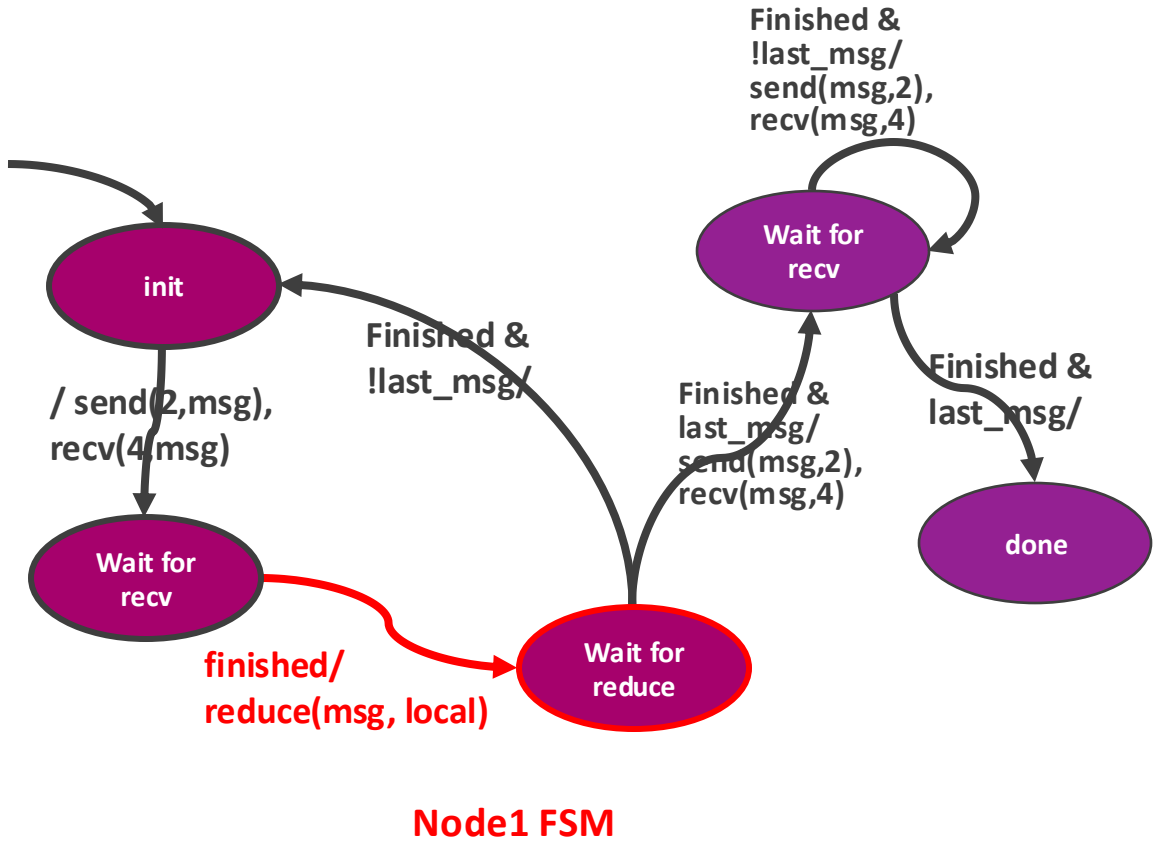




System Layer Collective Implementation

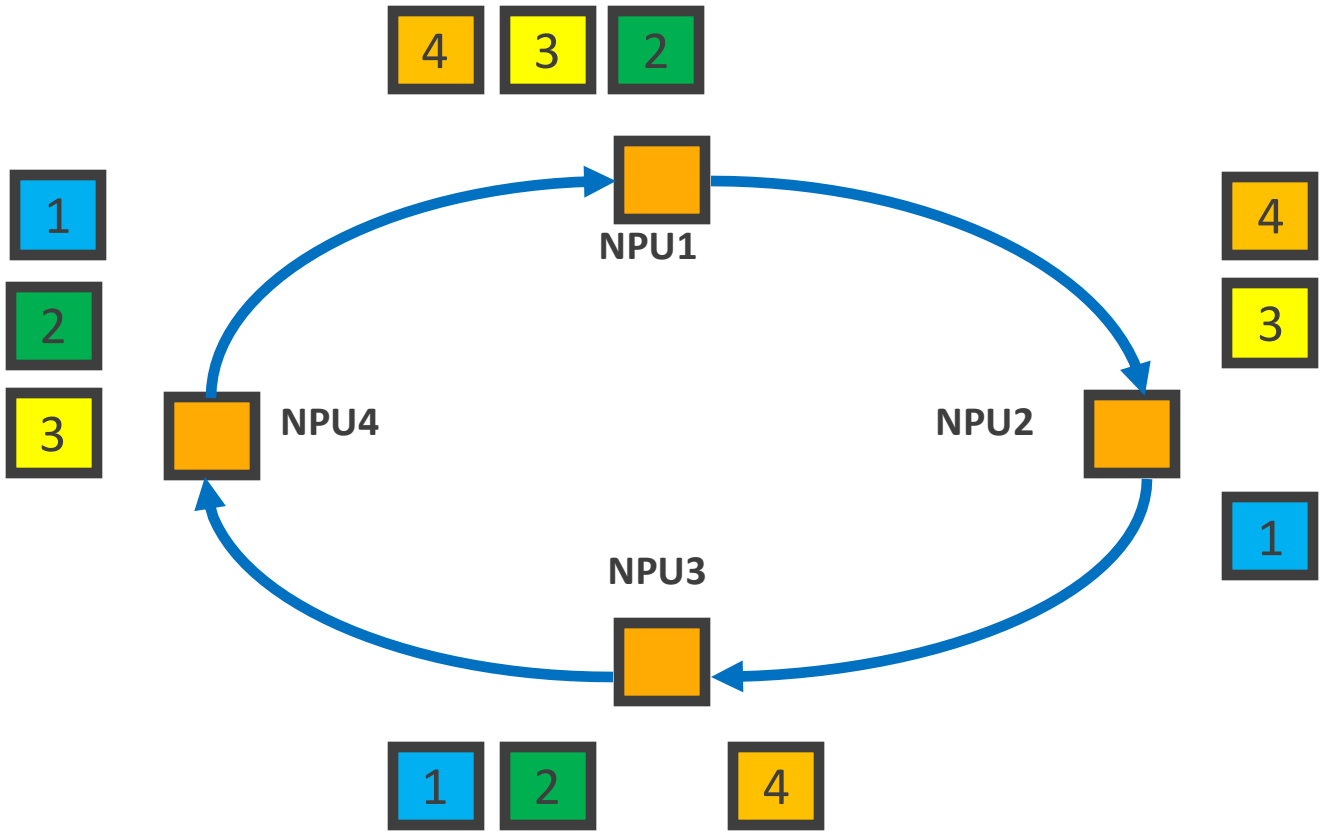
- Collective algorithms can be implemented using **state machines**.

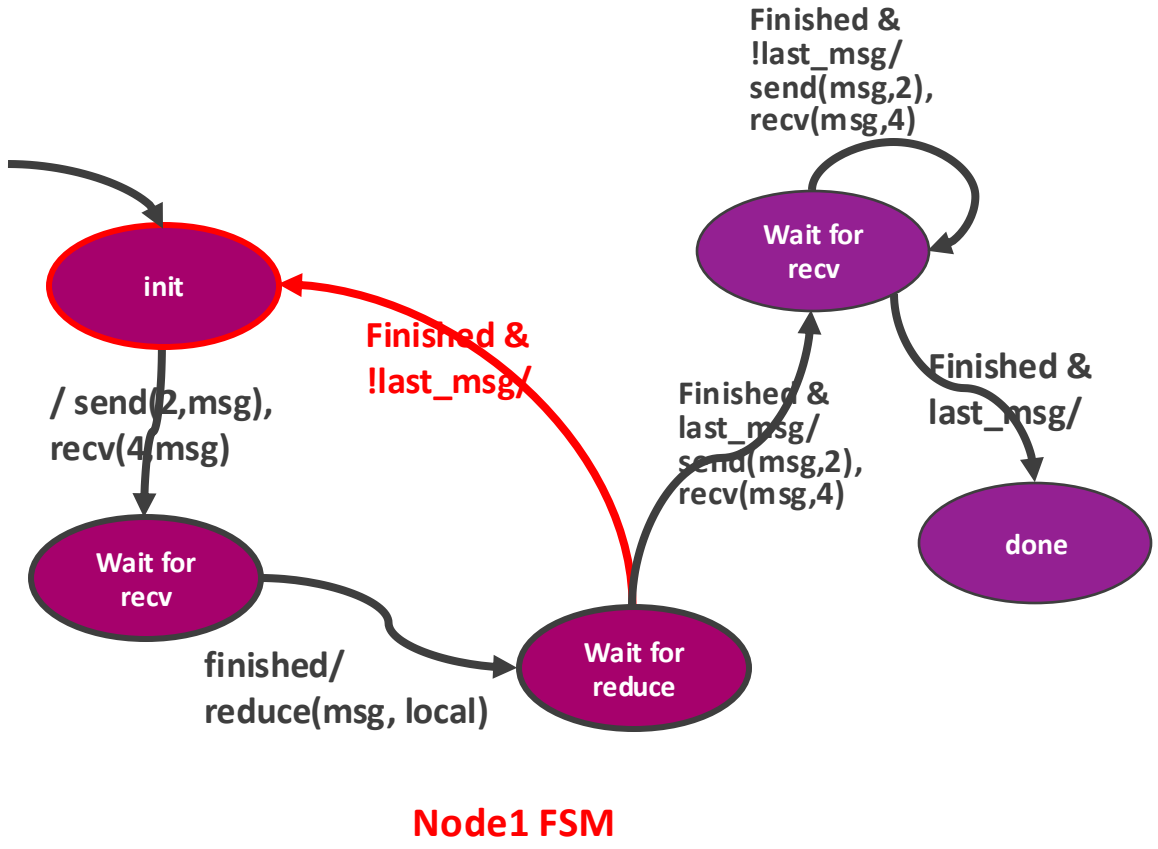




System Layer Collective Implementation

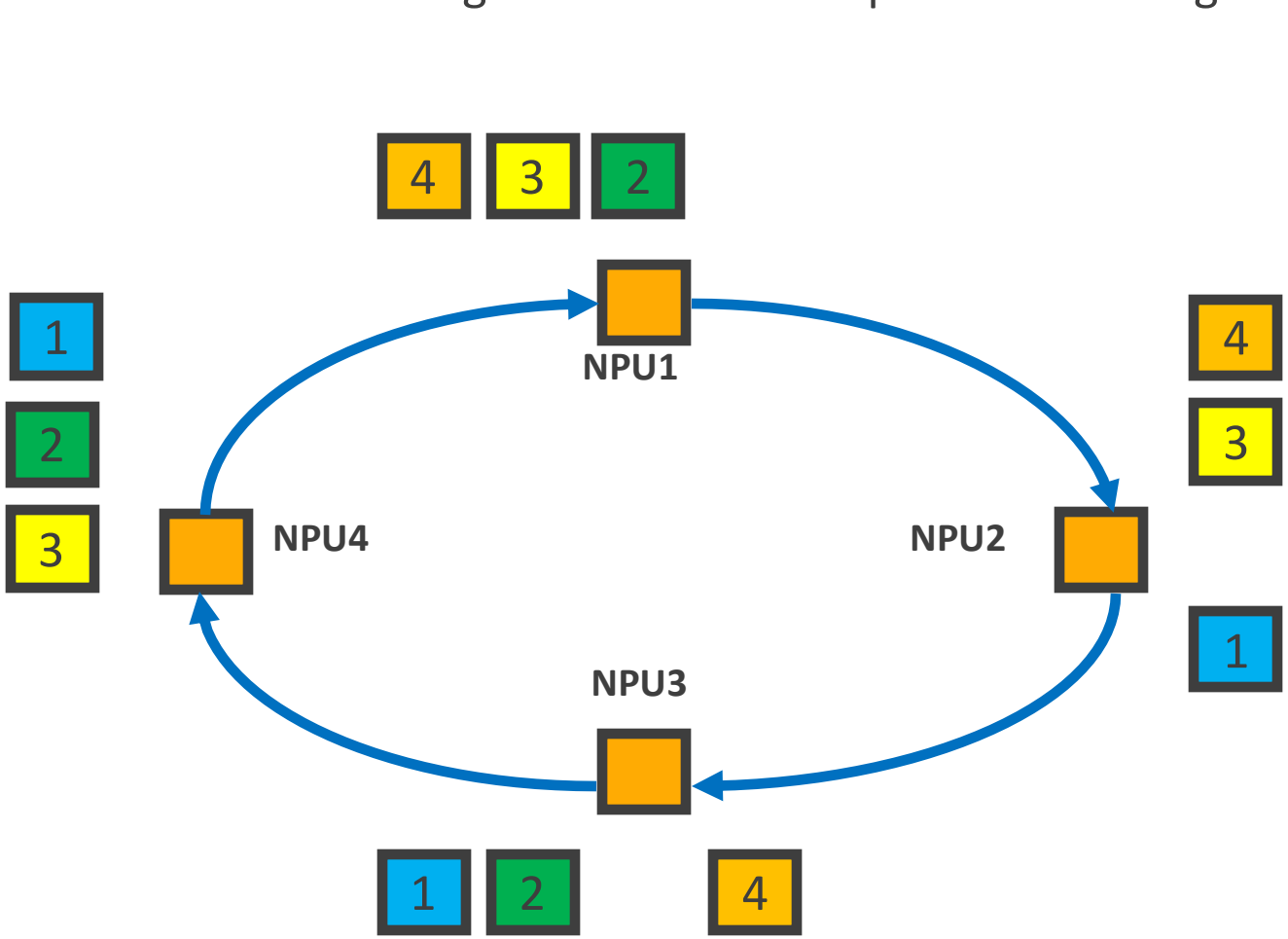
- Collective algorithms can be implemented using **state machines**.

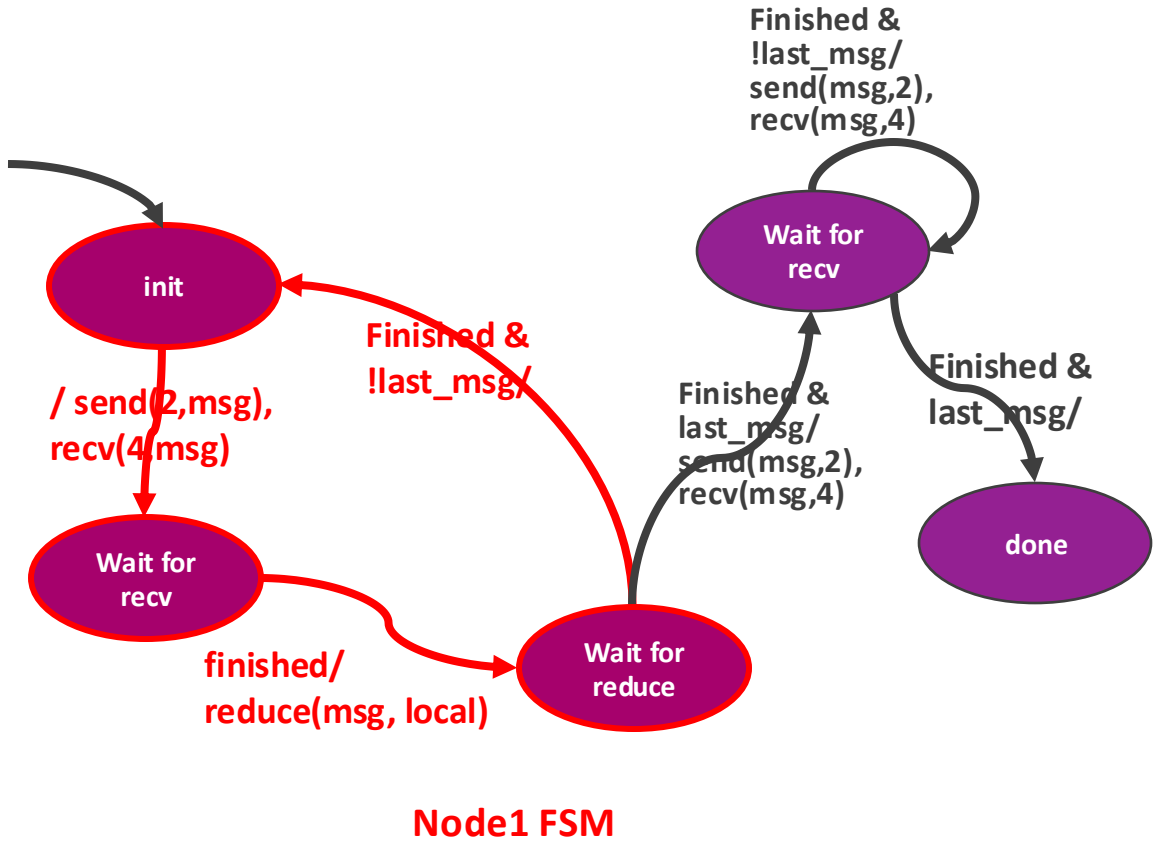




System Layer Collective Implementation

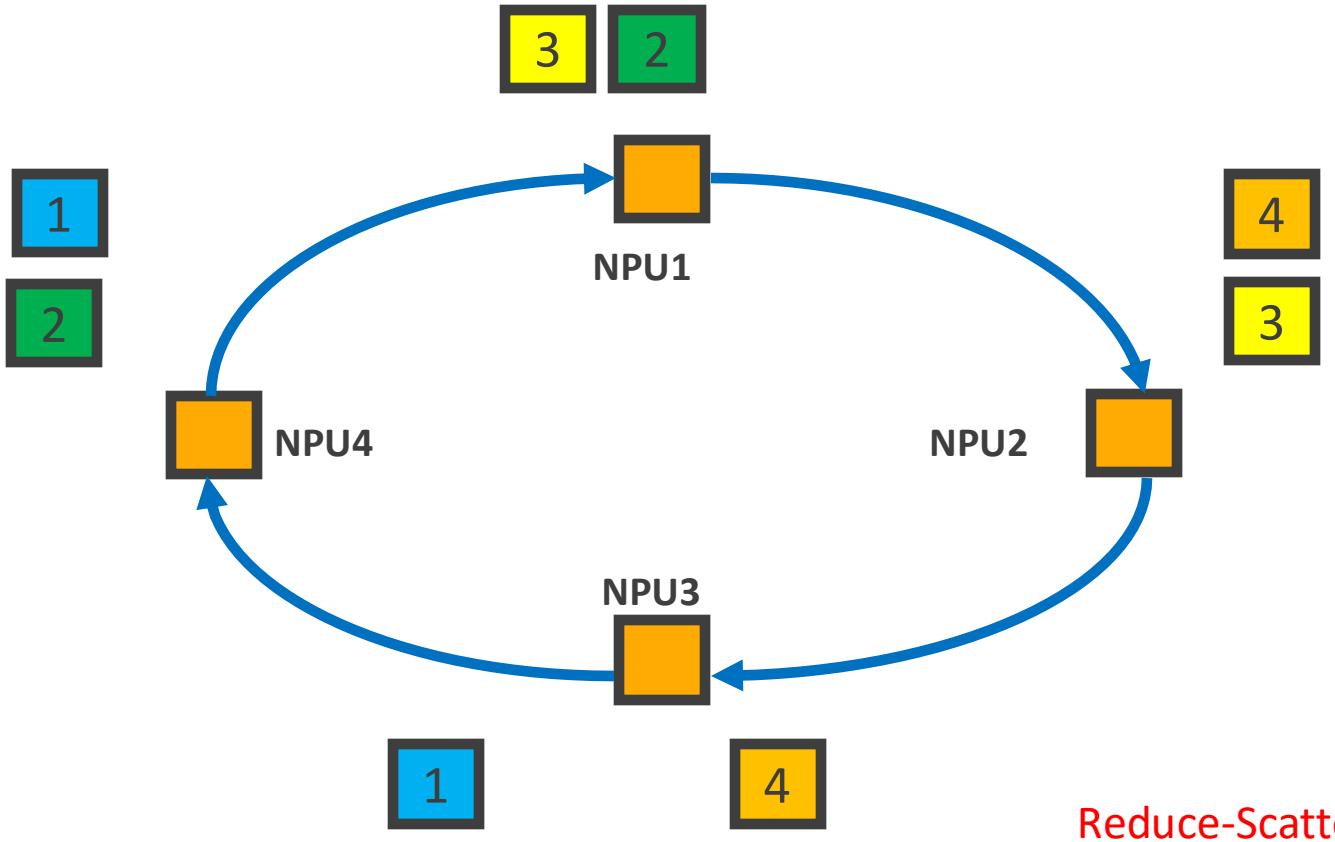
- Collective algorithms can be implemented using **state machines**.

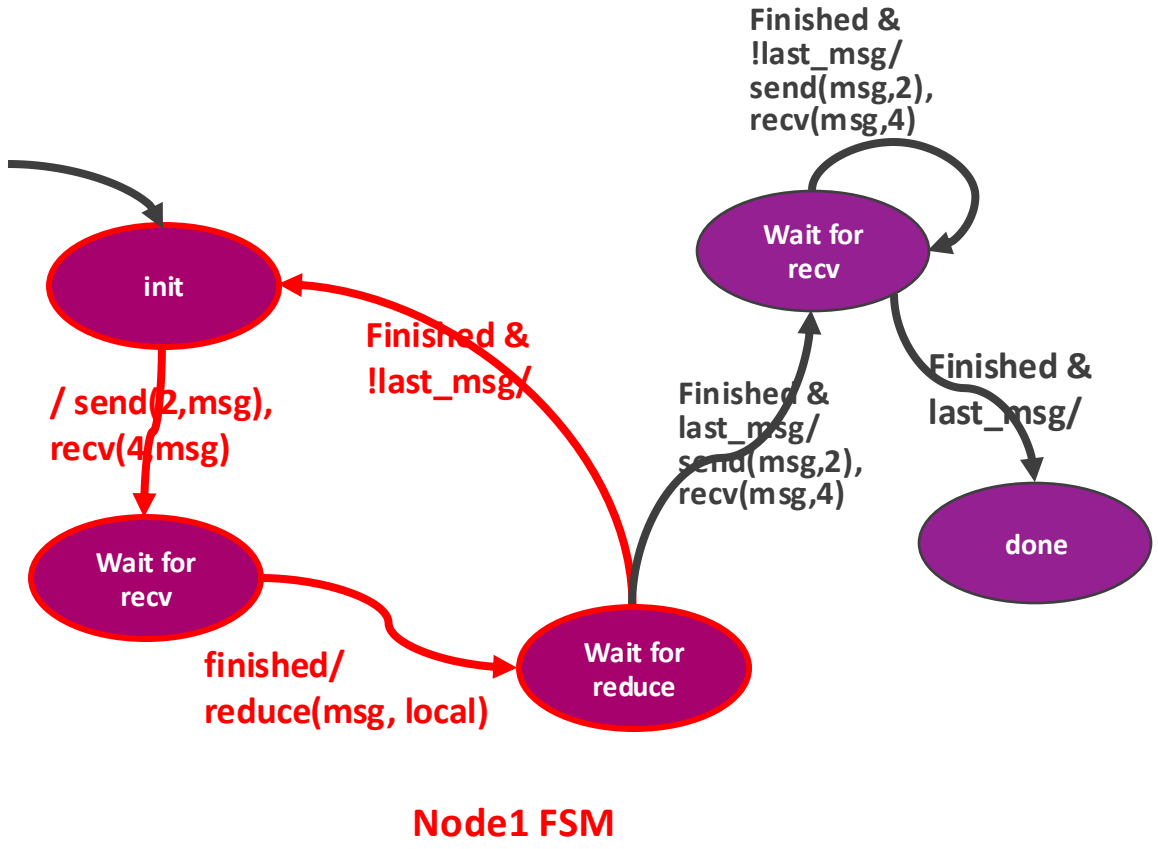




System Layer Collective Implementation

- Collective algorithms can be implemented using **state machines**.

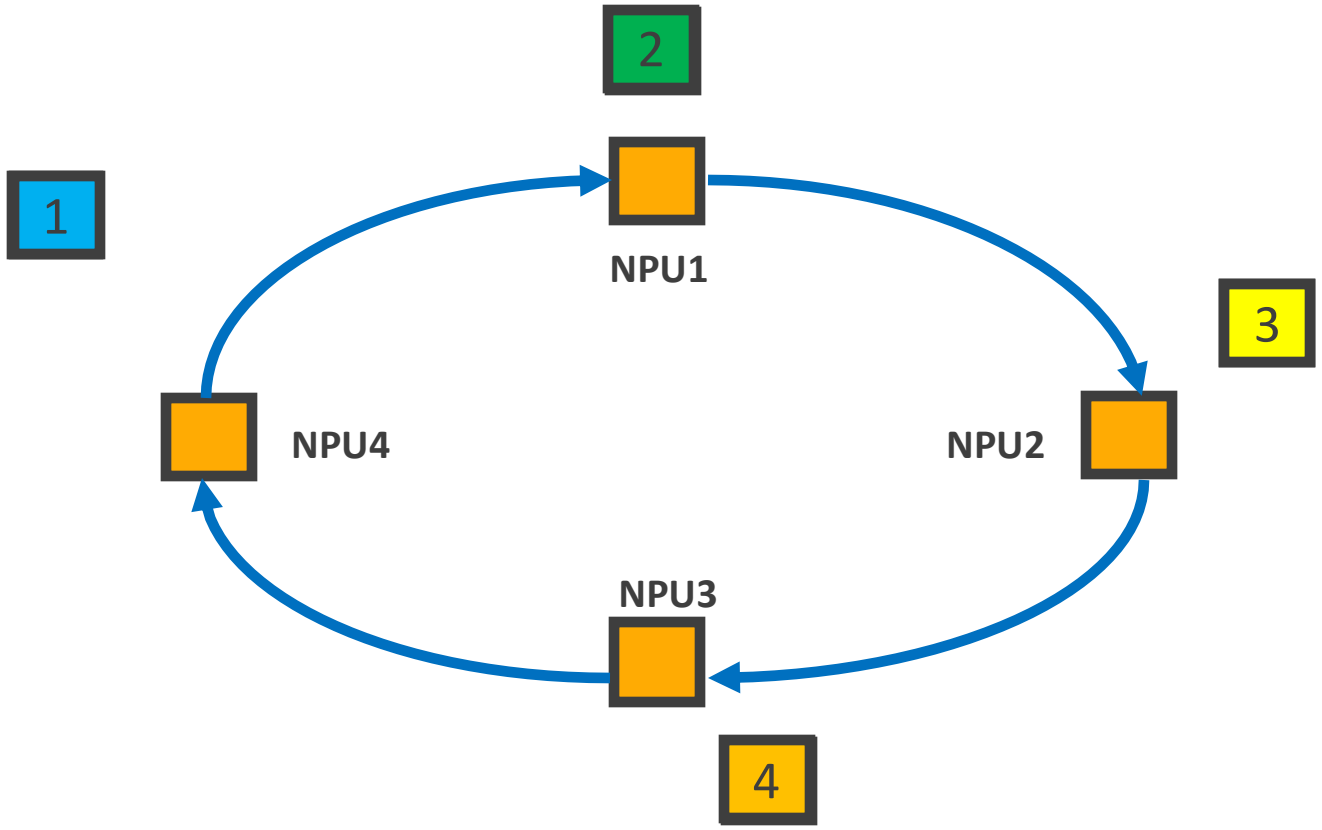


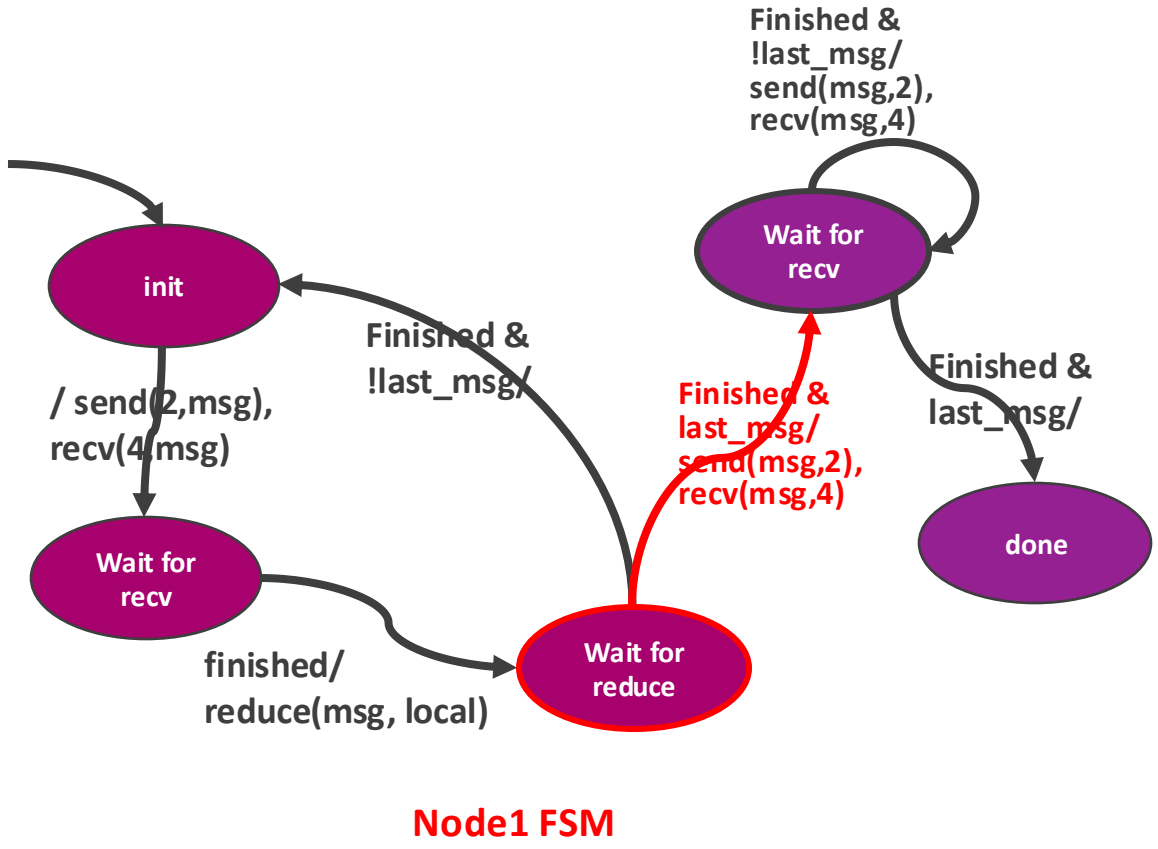


Reduce-Scatter done!

System Layer Collective Implementation

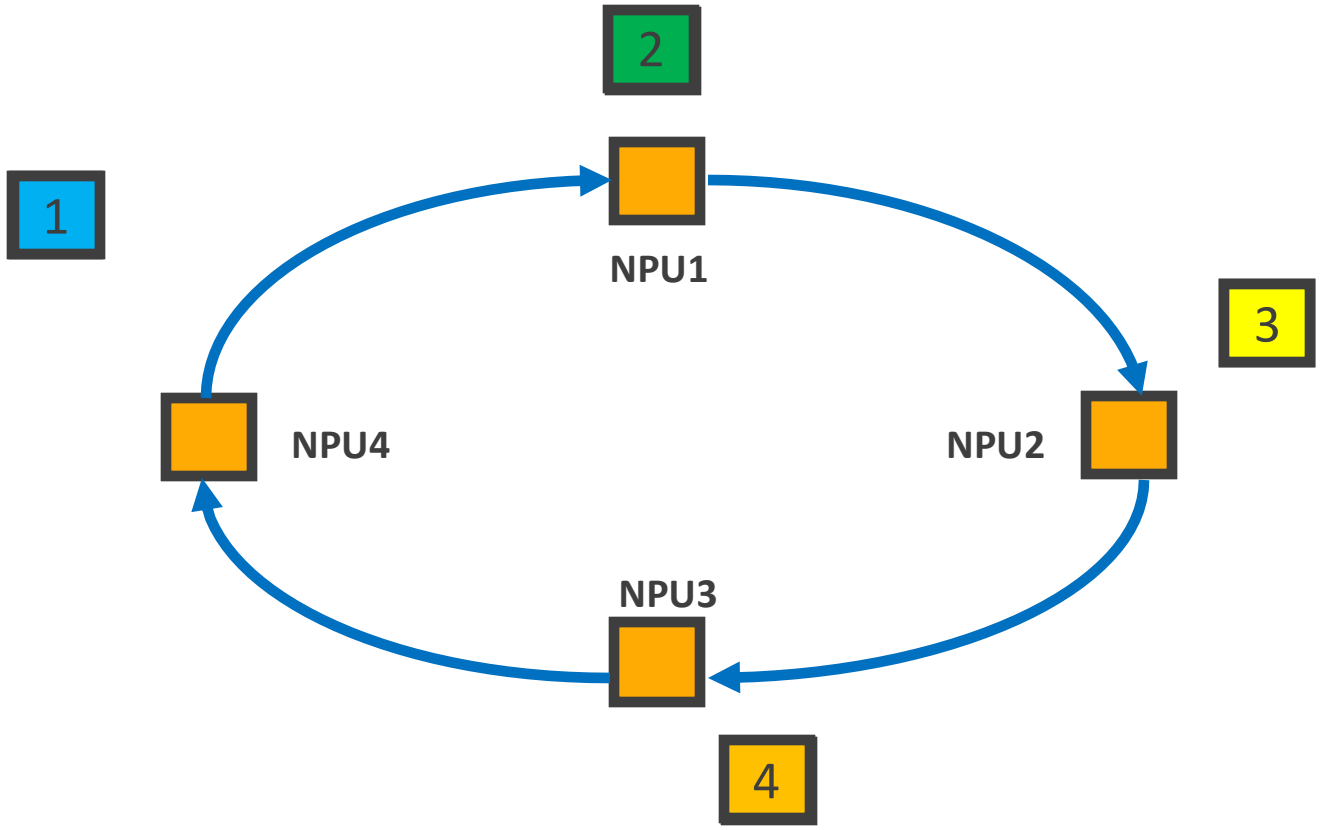
- Collective algorithms can be implemented using **state machines**.

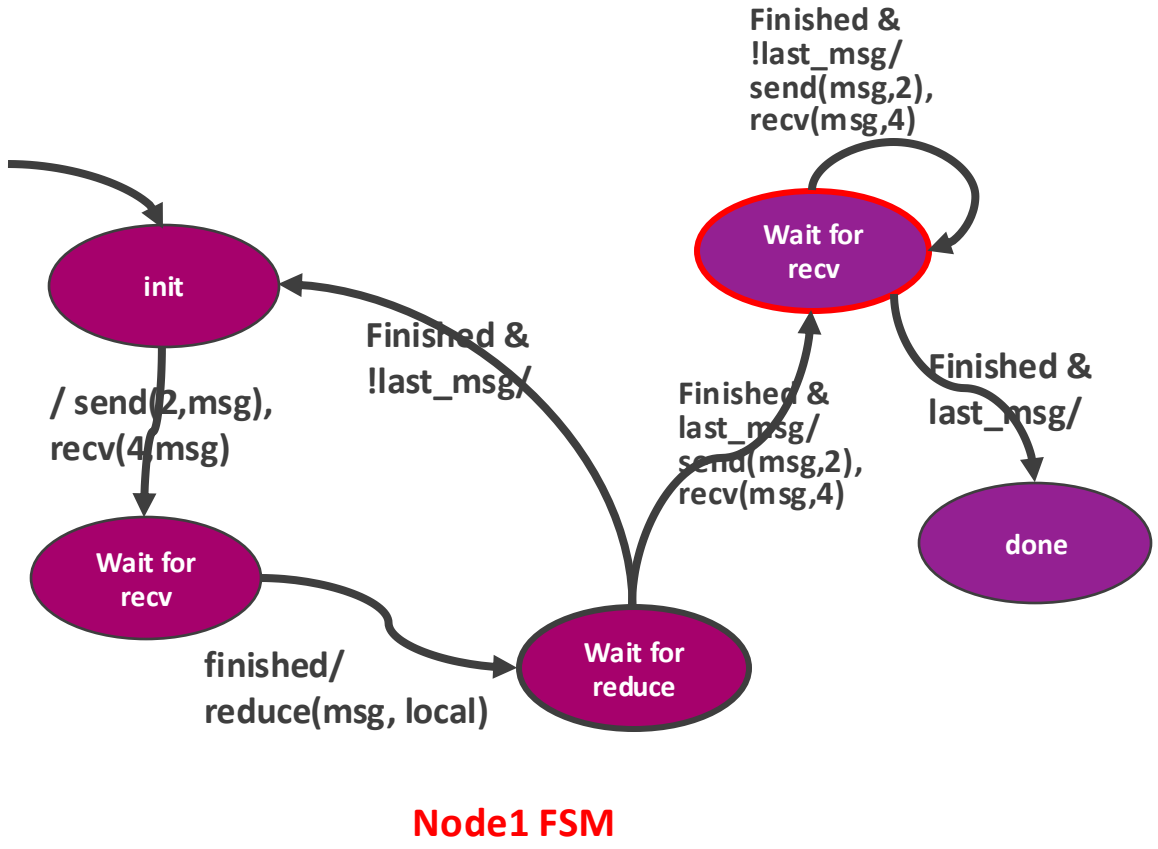




System Layer Collective Implementation

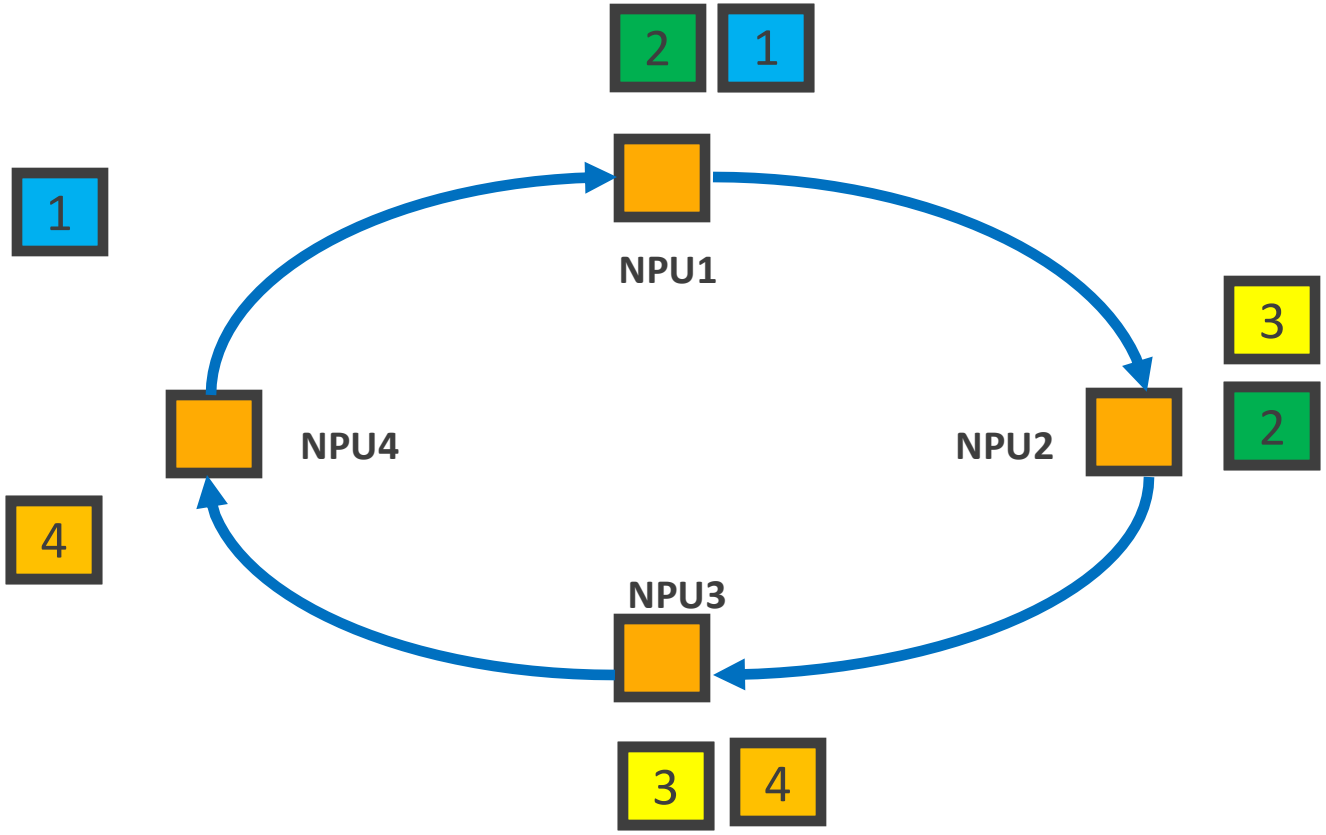
- Collective algorithms can be implemented using **state machines**.

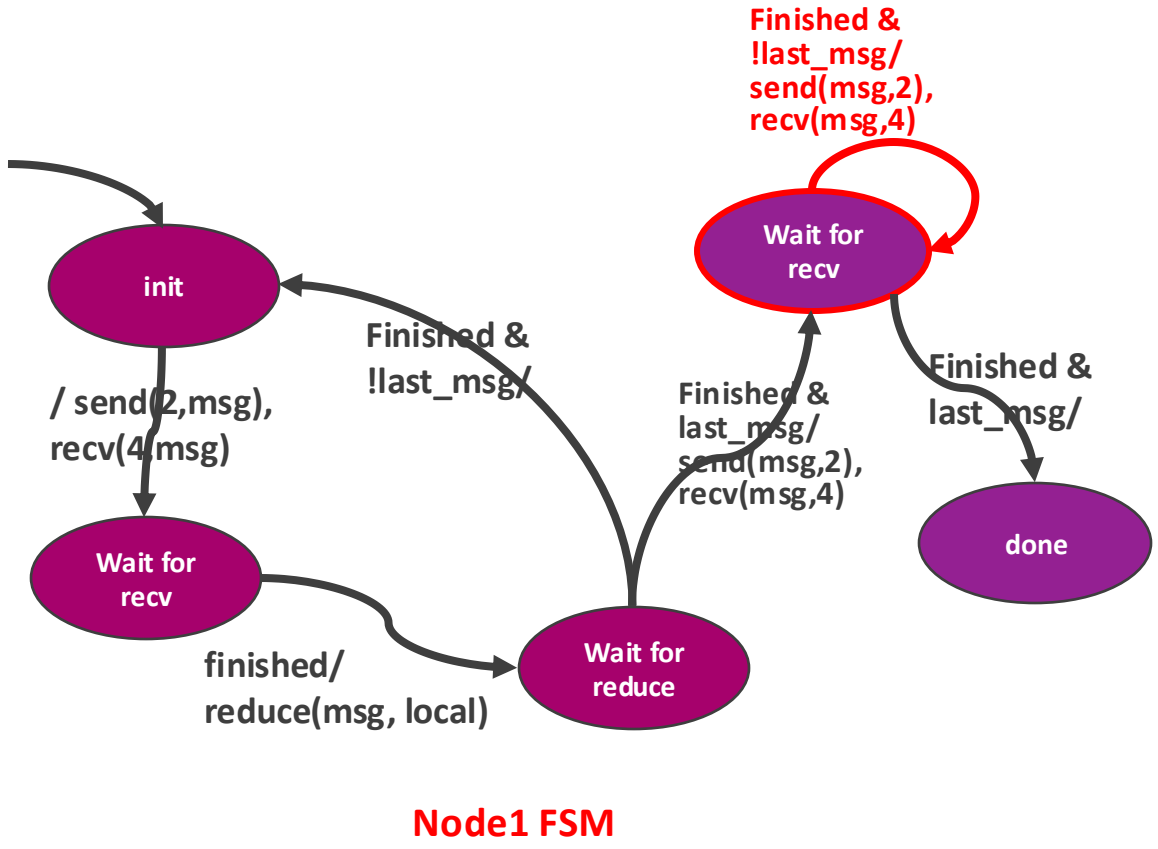




System Layer Collective Implementation

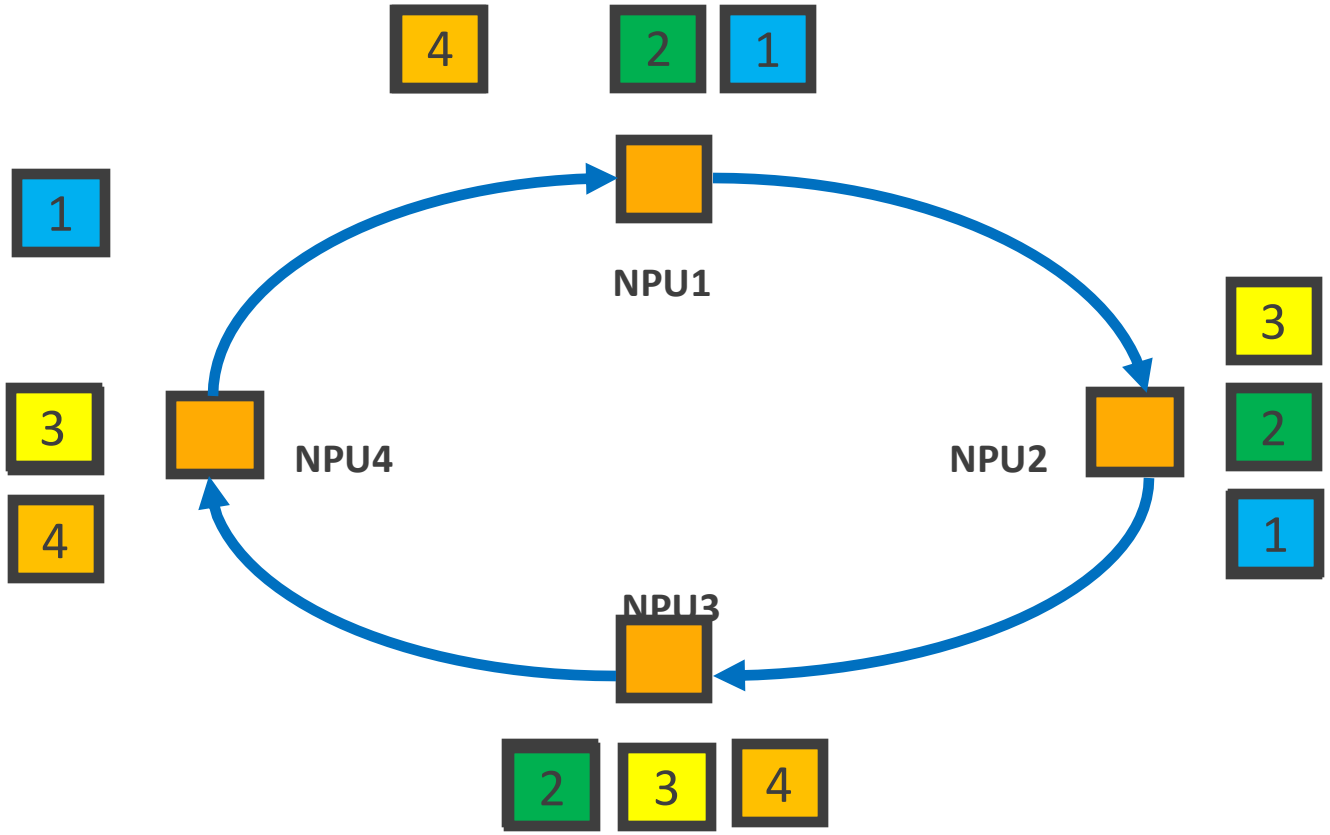
- Collective algorithms can be implemented using **state machines**.

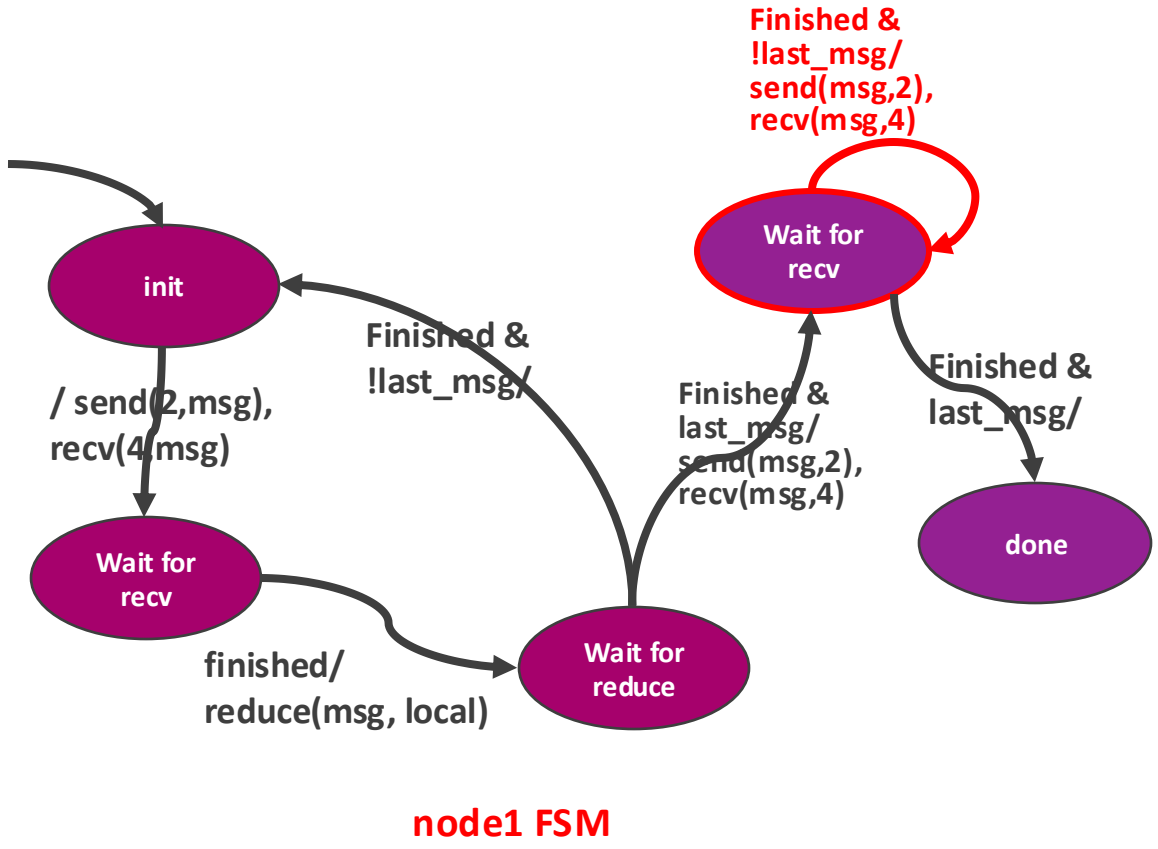




System Layer Collective Implementation

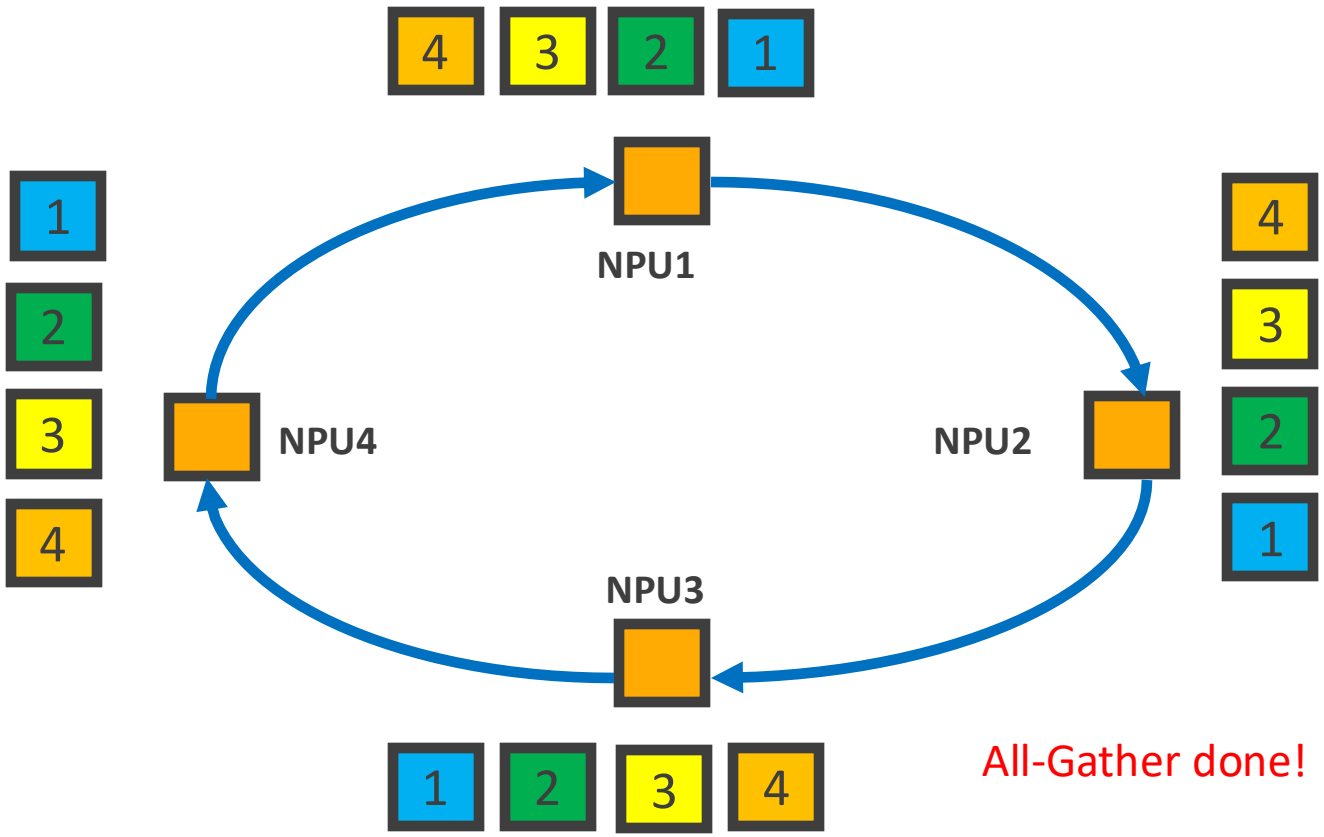
- Collective algorithms can be implemented using **state machines**.



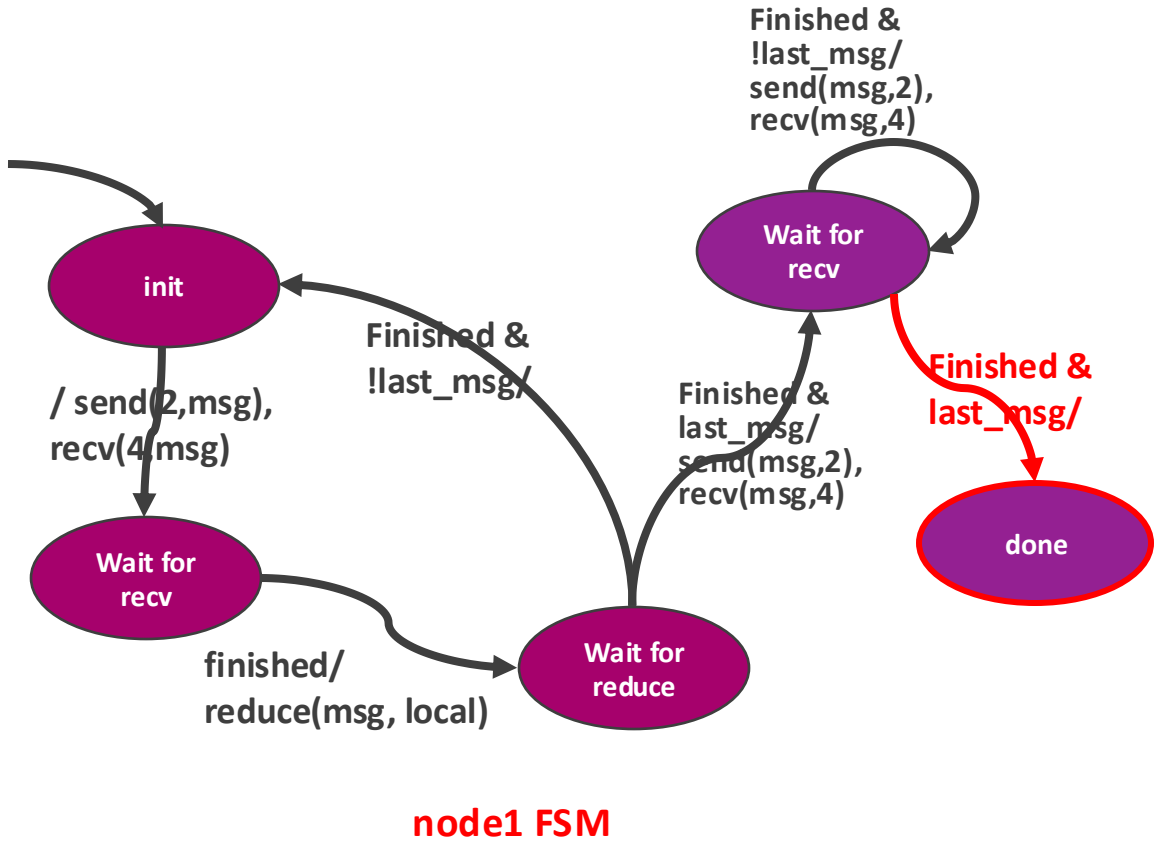


System Layer Collective Implementation

- Collective algorithms can be implemented using **state machines**.



All-Gather done!



node1 FSM

Collective Implementation

```
void Ring::run(EventType event, CallData* data) {
    if (event == EventType::General) {
        free_packets += 1;
        ready();
        iteratable();
    } else if (event == EventType::PacketReceived) {
        total_packets_received++;
        insert_packet(nullptr);
    } else if (event == EventType::StreamInit) {
        for (int i = 0; i < parallel_reduce; i++) {
            insert_packet(nullptr);
        }
    }
}
```

(i) issue send/recv operations
(ii) update FSM State

(i) check if Done
(ii) wait for the recv handler

Collective Implementation

```
void Ring::run(EventType event, CallData* data) {
    if (event == EventType::General) {
        free_packets += 1;
        ready();
        iteratable();
    } else if (event == EventType::PacketReceived) {
        total_packets_received++;
        insert_packet(nullptr);
    } else if (event == EventType::StreamInit) {
        for (int i = 0; i < parallel_reduce; i++) {
            insert_packet(nullptr);
        }
    }
}
```

when recv event is Done:
(i) insert next chunks to send
(ii) repeat the process

Multi-dimensional Collectives

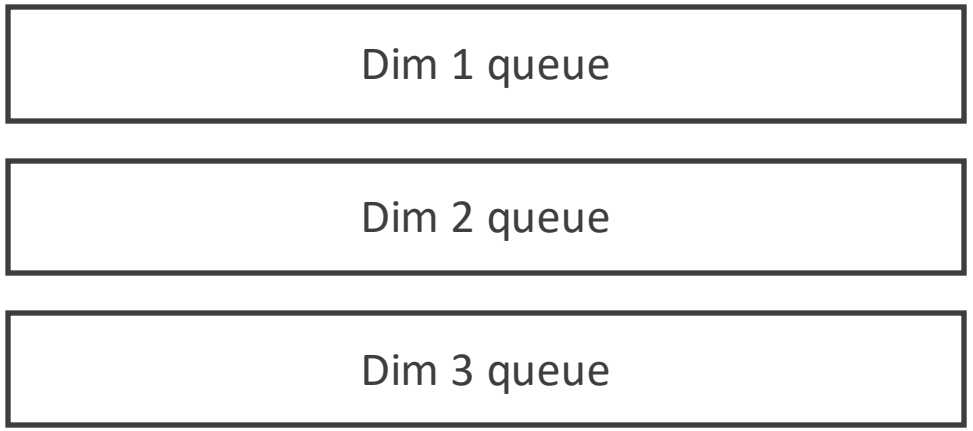
```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

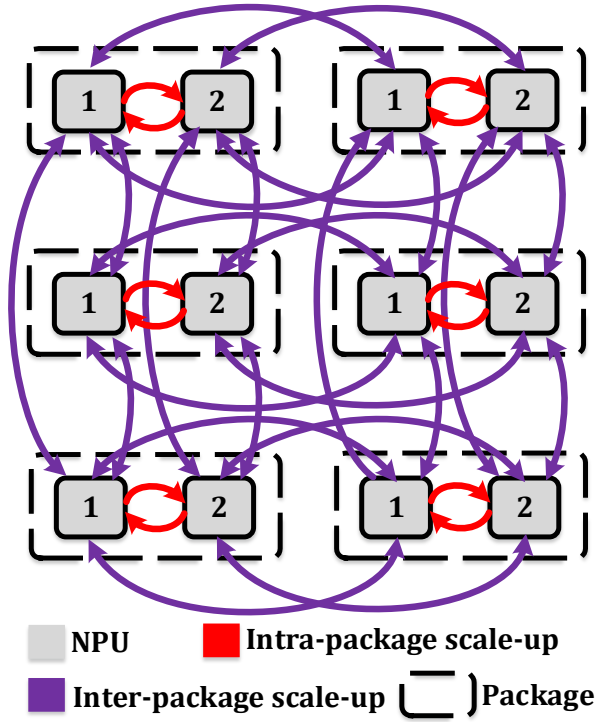
Support for multi-dimensional (hierarchical) collective algorithms

"baseline": process one chunk at a time
"localBWAware": pipelined execution

Collective Scheduler

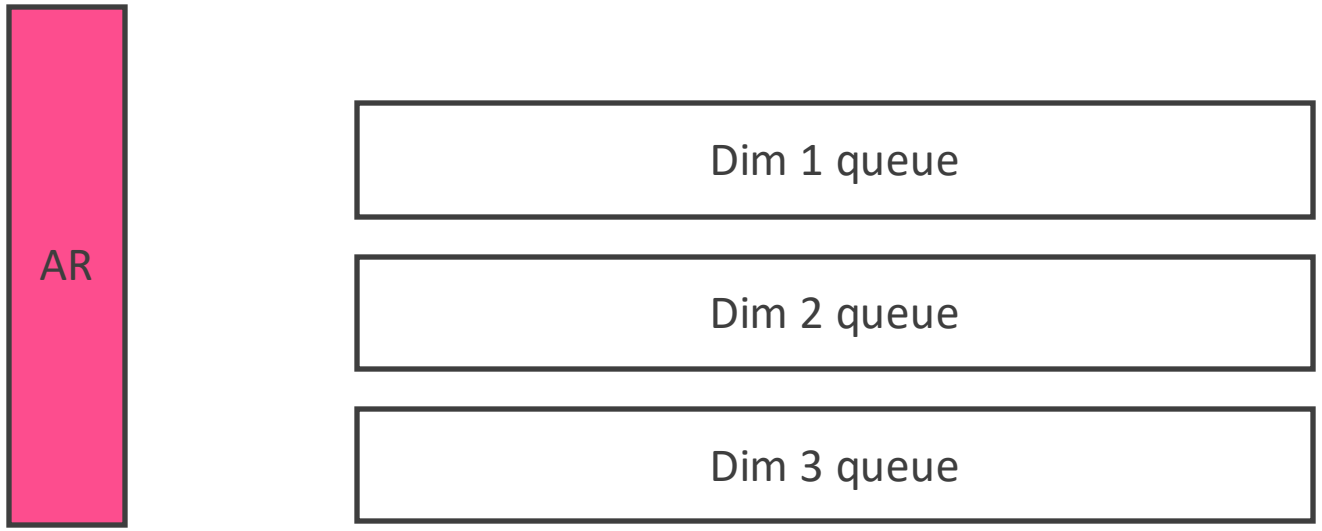
- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.



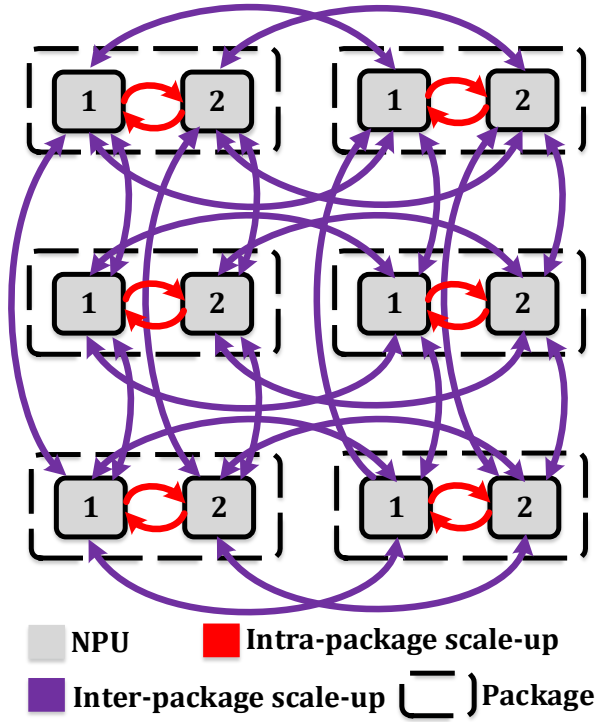


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

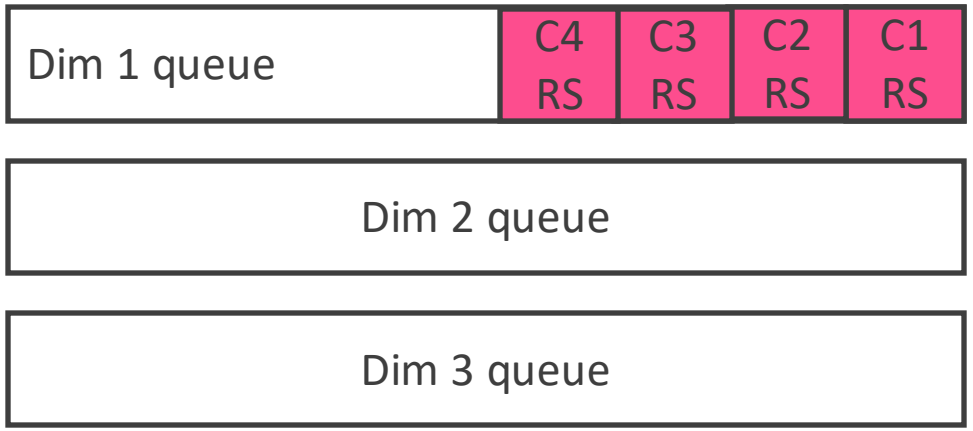


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

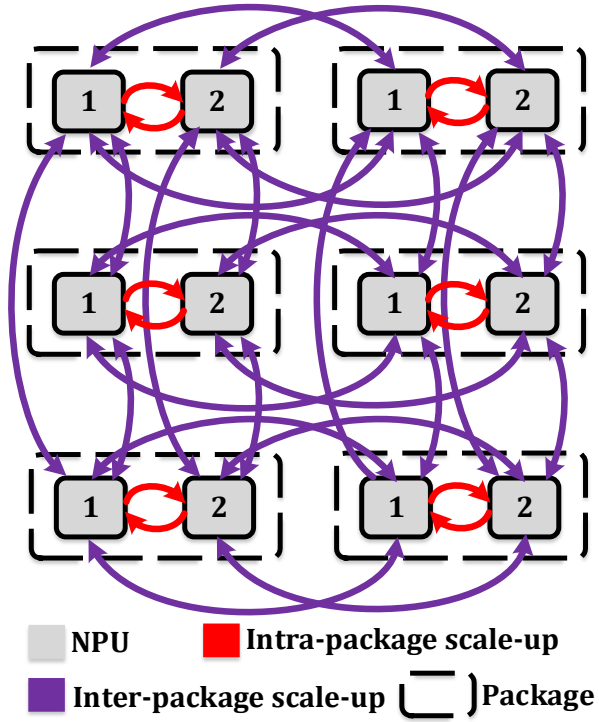


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

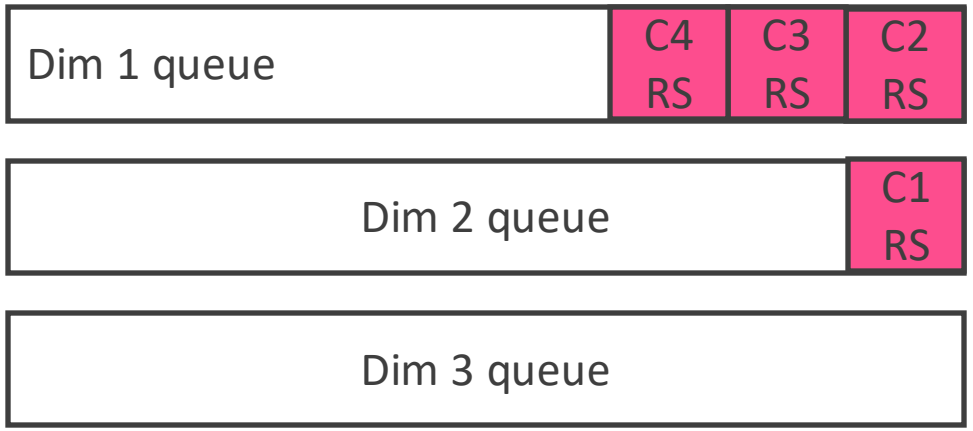


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

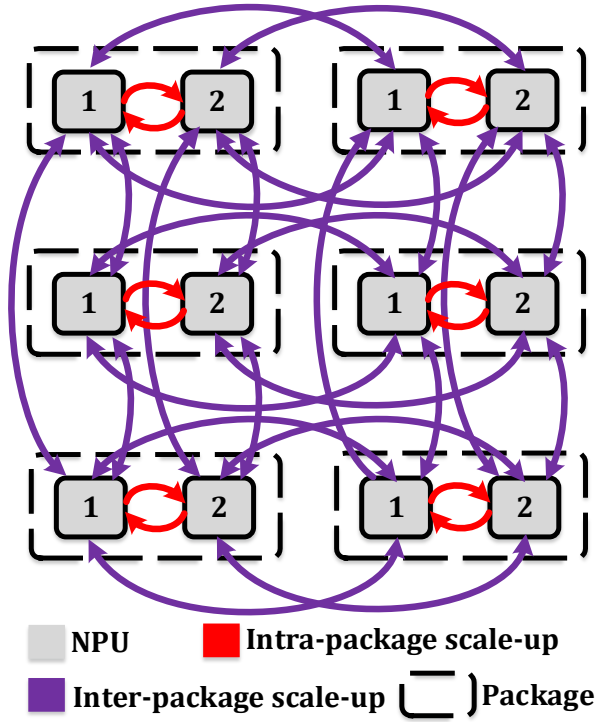


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

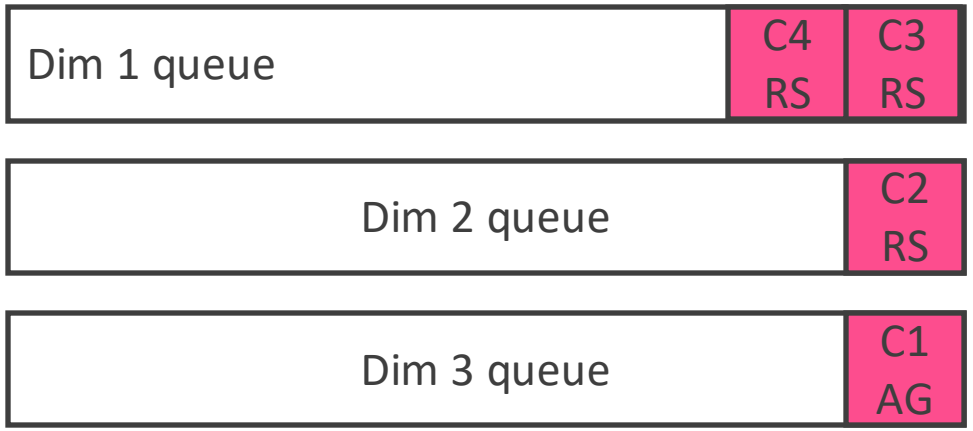


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

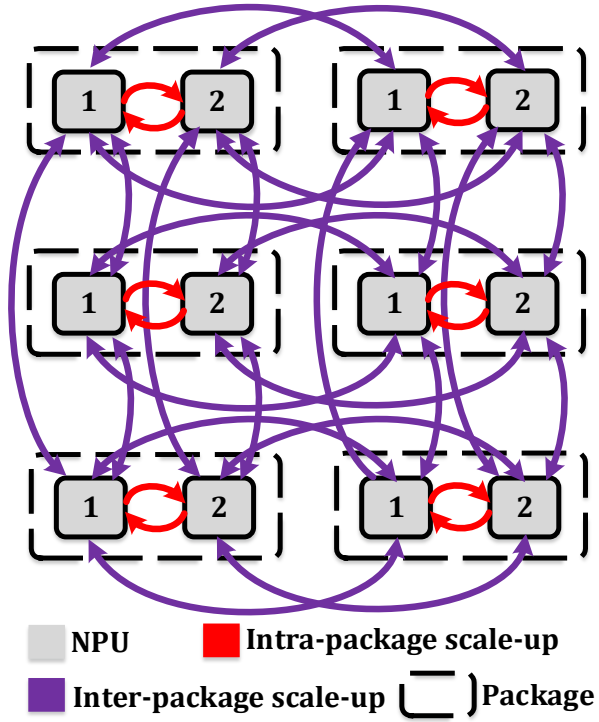


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

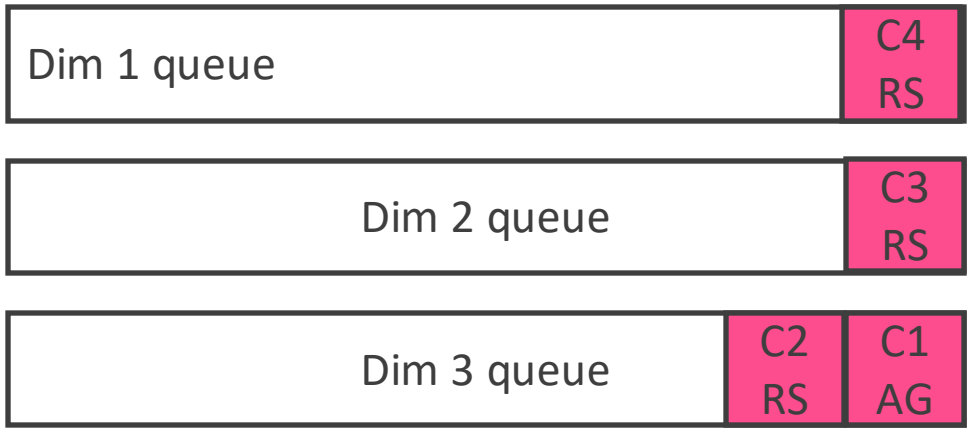


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

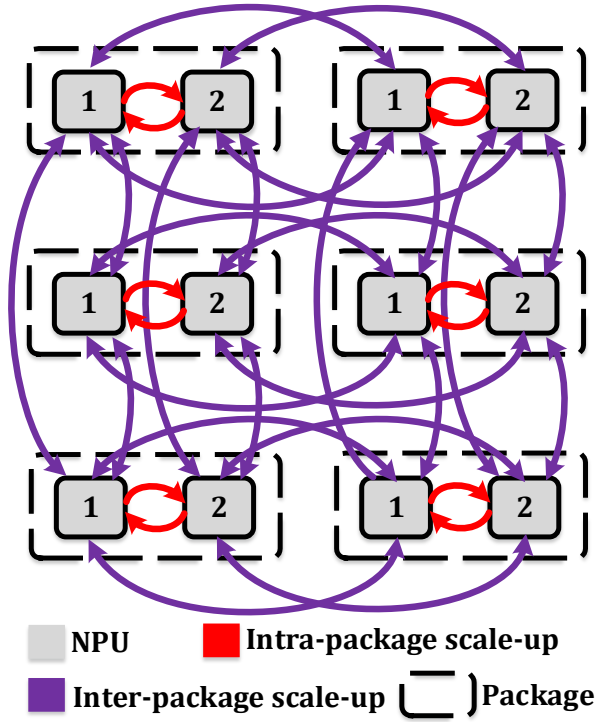


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

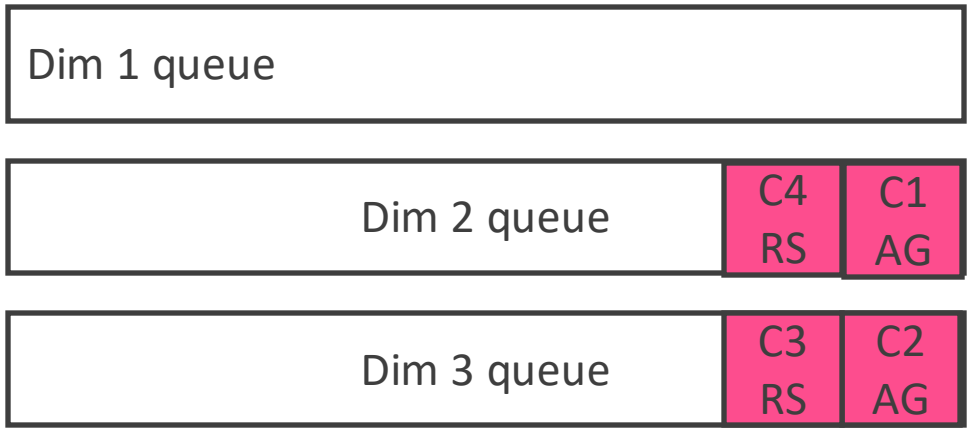


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

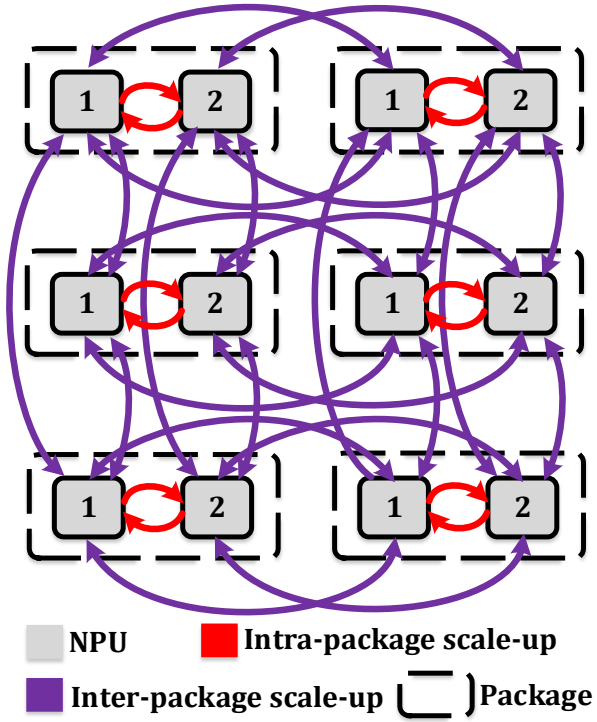


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

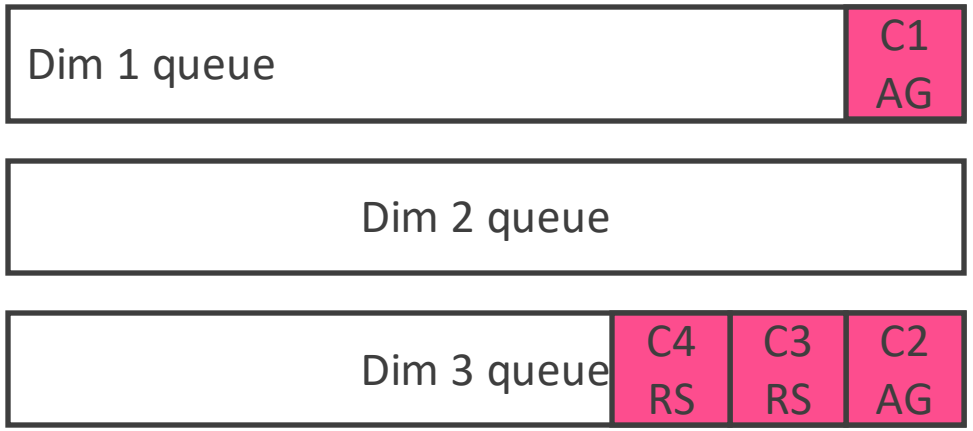


AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather

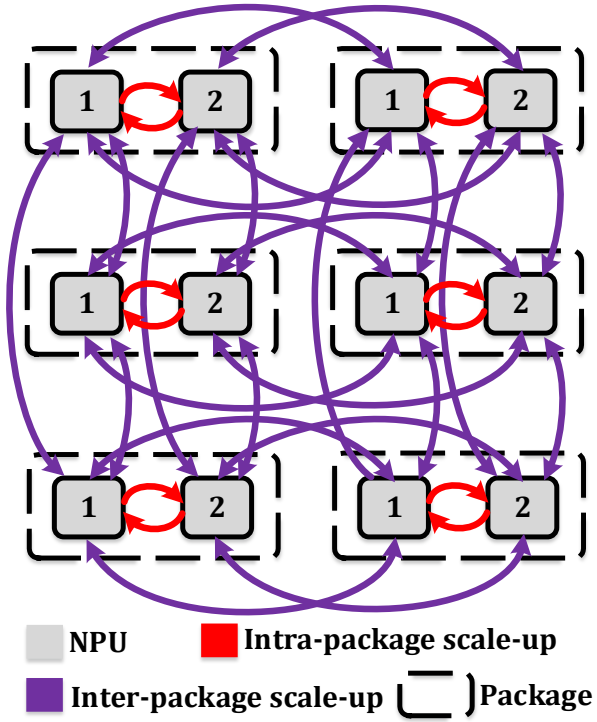


Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.

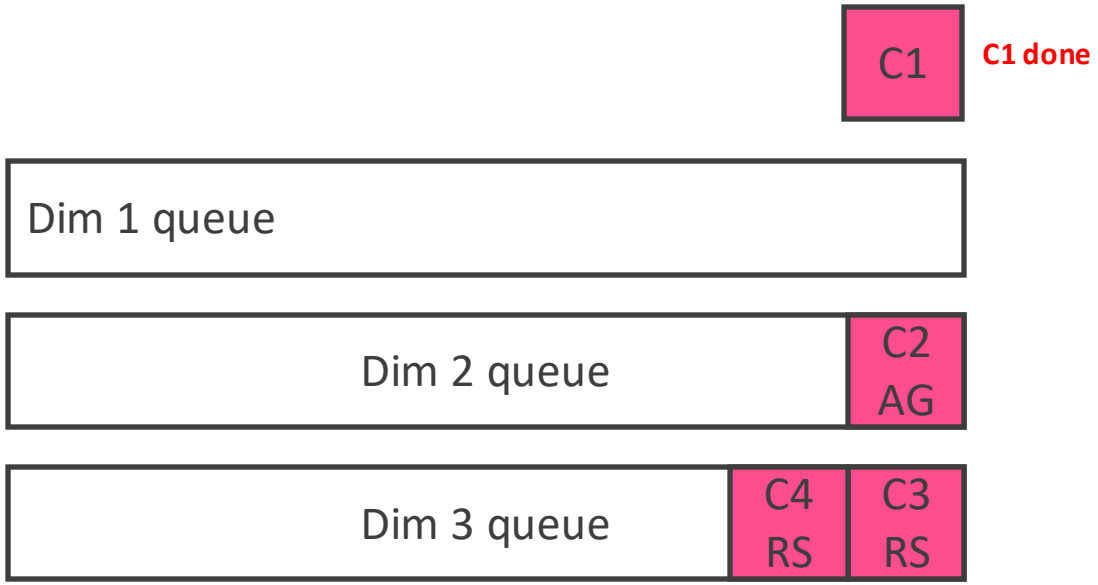


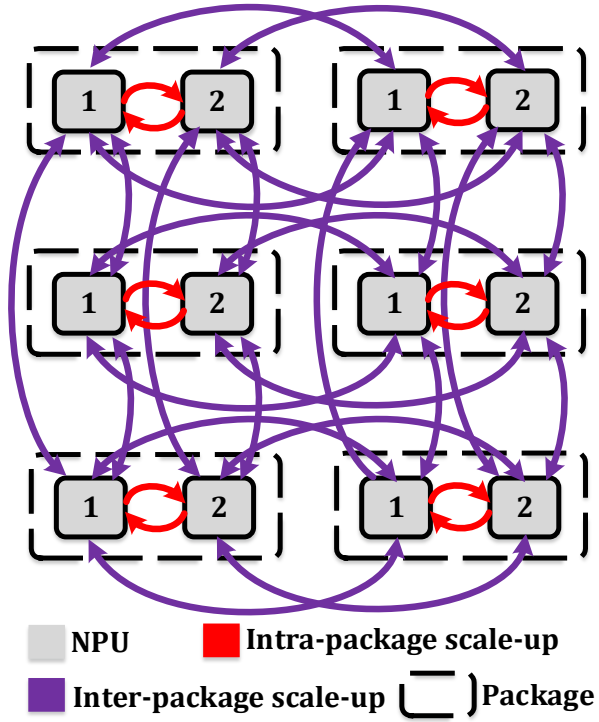
AR: All-Reduce
RS: Reduce-Scatter
AG: All-Gather



Collective Scheduler

- There are one/multiple queue(s) per each physical network dimension.
- A collective is broken into multiple chunks and inserted into the first queue.
- Queues process chunks in-order.





System Input File

```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

processed #chunks per dim queue

How many chunks per collective

System Input File

```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

inter-collective scheduling policy (FIFO or LIFO)

System Input File

```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

Each NPU's constant delay per send/recv/reduction operation

System Input File

```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

local memory BW to model data I/O

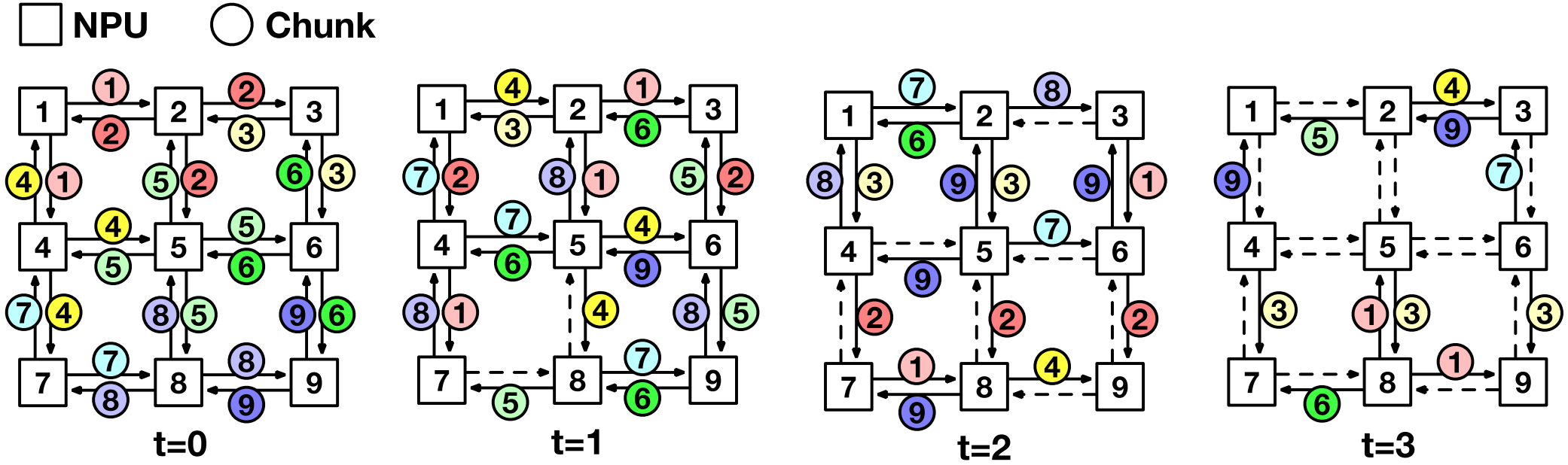
System Input File

```
{
    "scheduling-policy": "LIFO",
    "endpoint-delay": 10,
    "active-chunks-per-dimension": 1,
    "preferred-dataset-splits": 4,
    "all-reduce-implementation": [
        "ring", "ring", "ring"
    ],
    "all-gather-implementation": [
        "ring", "ring", "ring"
    ],
    "reduce-scatter-implementation": [
        "ring", "ring", "ring"
    ],
    "all-to-all-implementation": [
        "ring", "ring", "ring"
    ],
    "collective-optimization": "localBWAware",
    "local-mem-bw": 1600,
    "boost-mode": 0
}
```

0: simulate entire cluster
1: simulate only GPU 0 (to speed up simulation)

TACOS: Custom Collective

- A mechanism to generate **topology-aware custom collective plan**

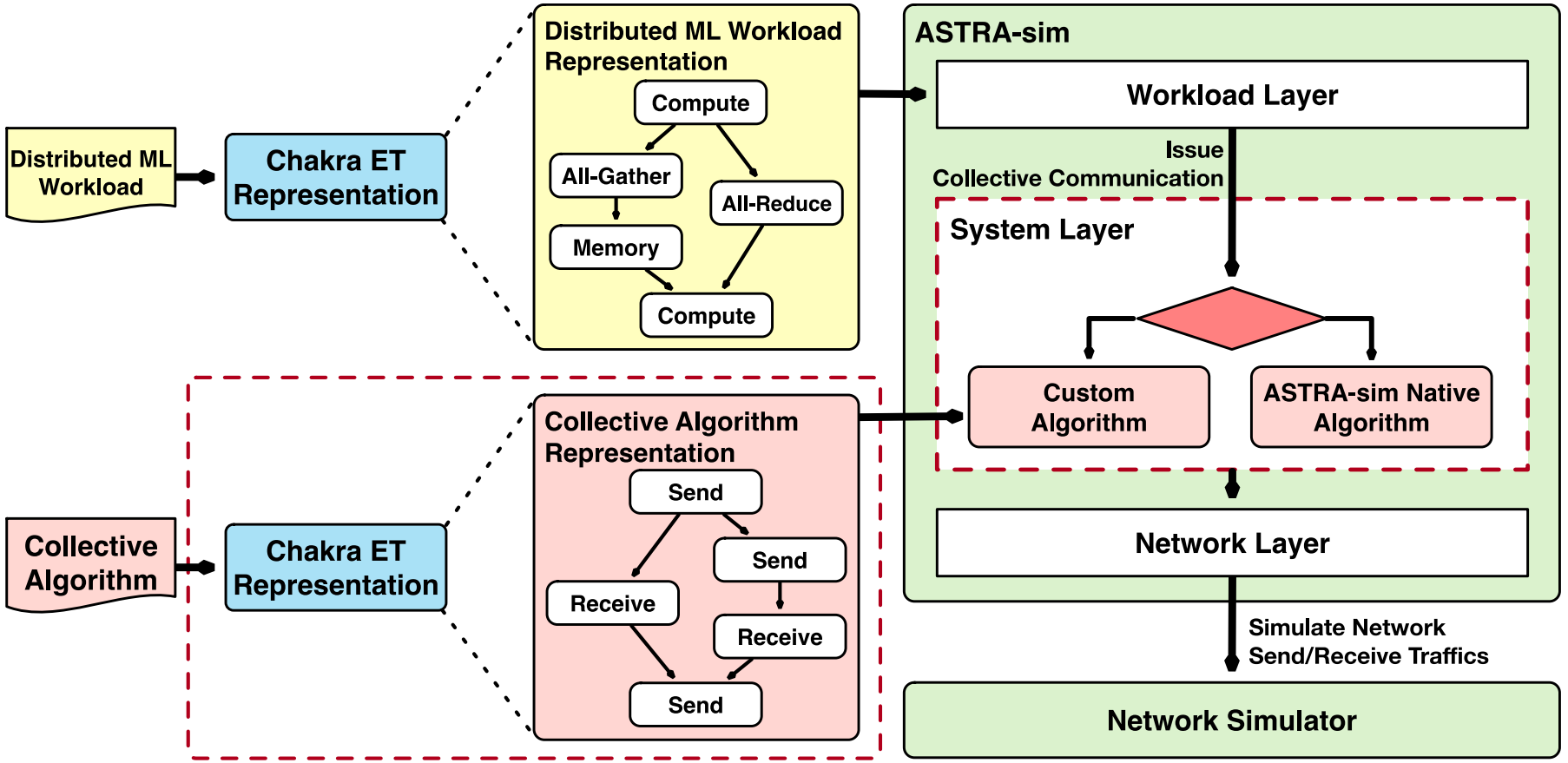


TACOS custom collective plan

This Tuesday, Session 6B (11 am, Room B)

- *https://arxiv.org/abs/2304.05301*

Custom Collective Support in ASTRA-sim



This tutorial, 4:10 pm

- *ASTRA-sim New Features*