Hardware-Software Coordination for High-Performance Concurrent Data Structures with Near-Data-Processing

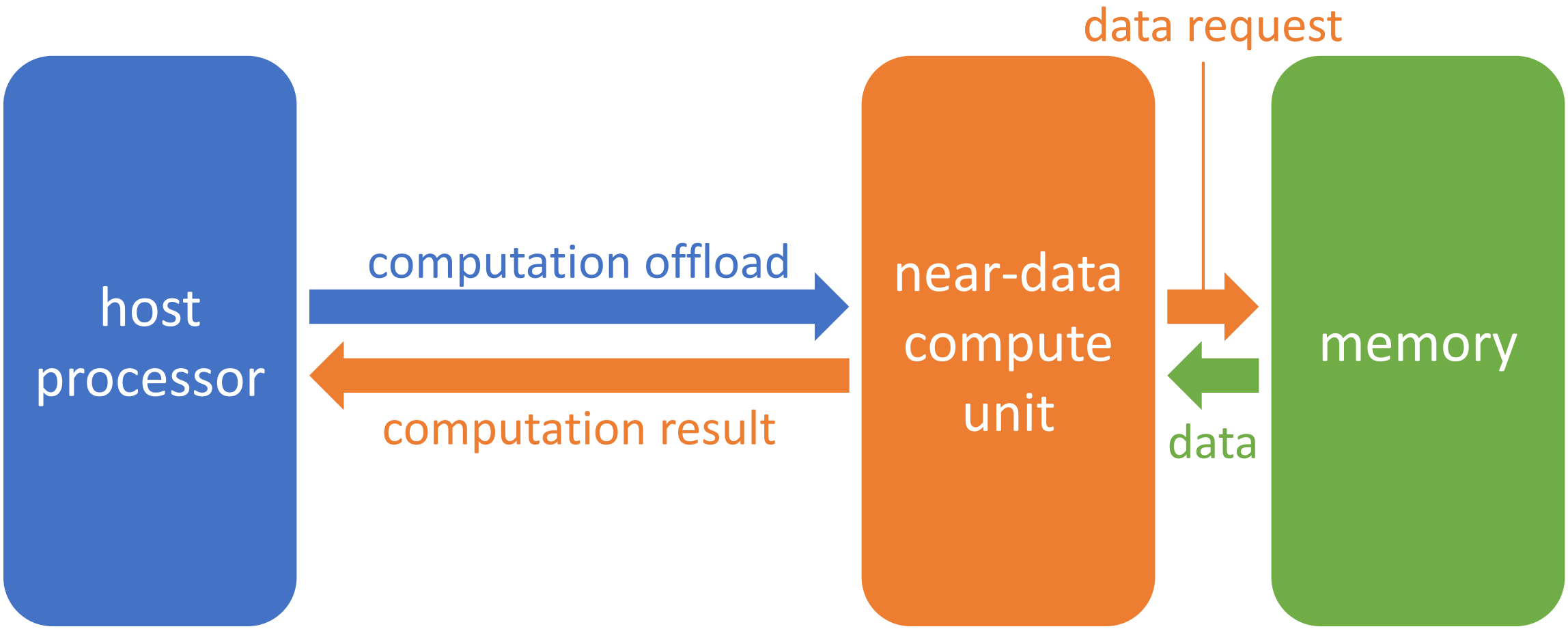
Jiwon Choe, Amy Huang, Tali Moreshet, Maurice Herlihy, Iris Bahar

1/25/2019

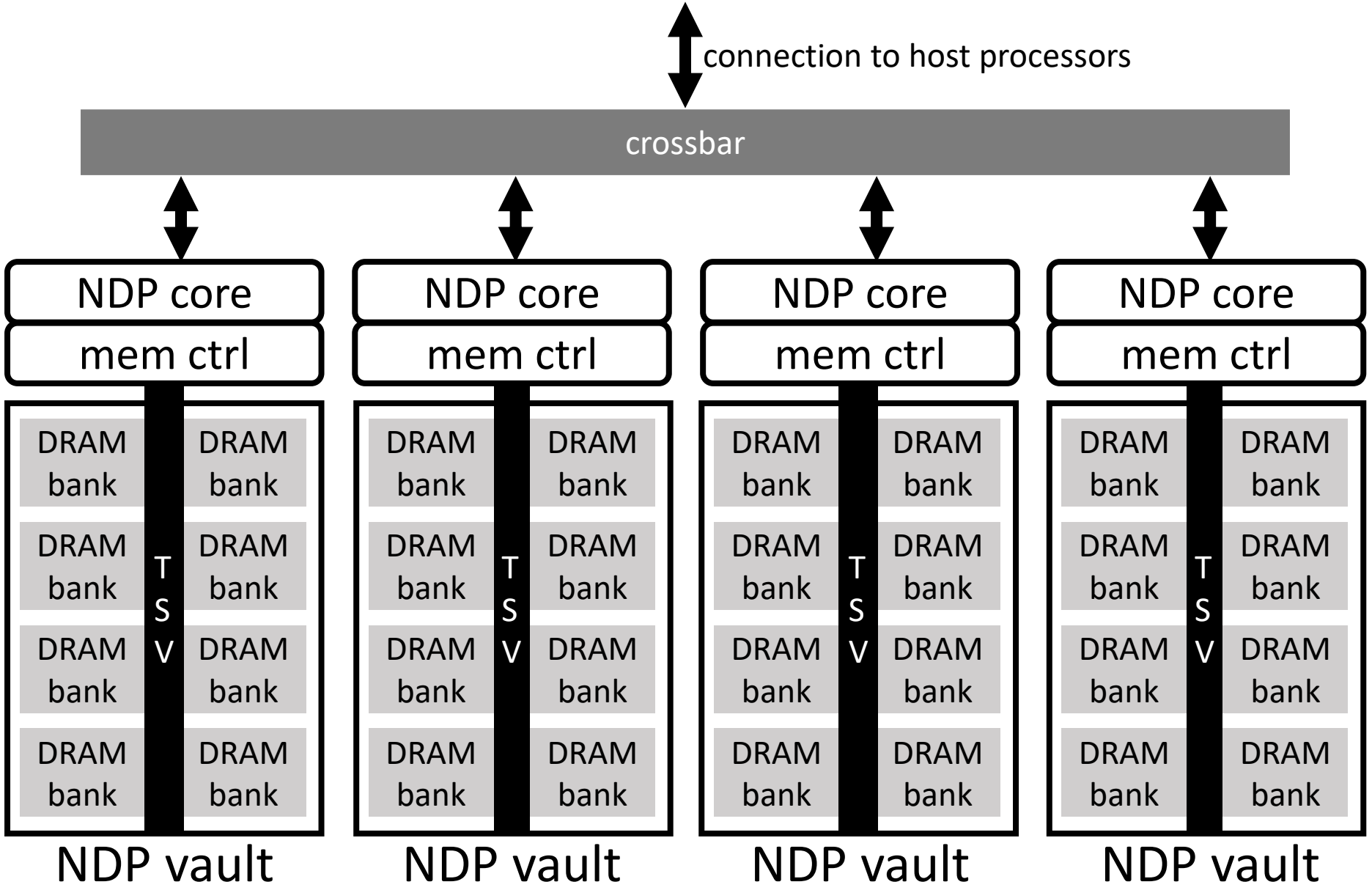
Boston Area Architecture Workshop

Near-Data-Processing (NDP)

Near-Data-Processing (NDP)



Near-Data-Processing (NDP)



Concurrent Data Structures

- Bottlenecks on conventional architectures:
  - Poor cache locality (pointer-chasing data structures)
  - High-contention spots (contended data structures)

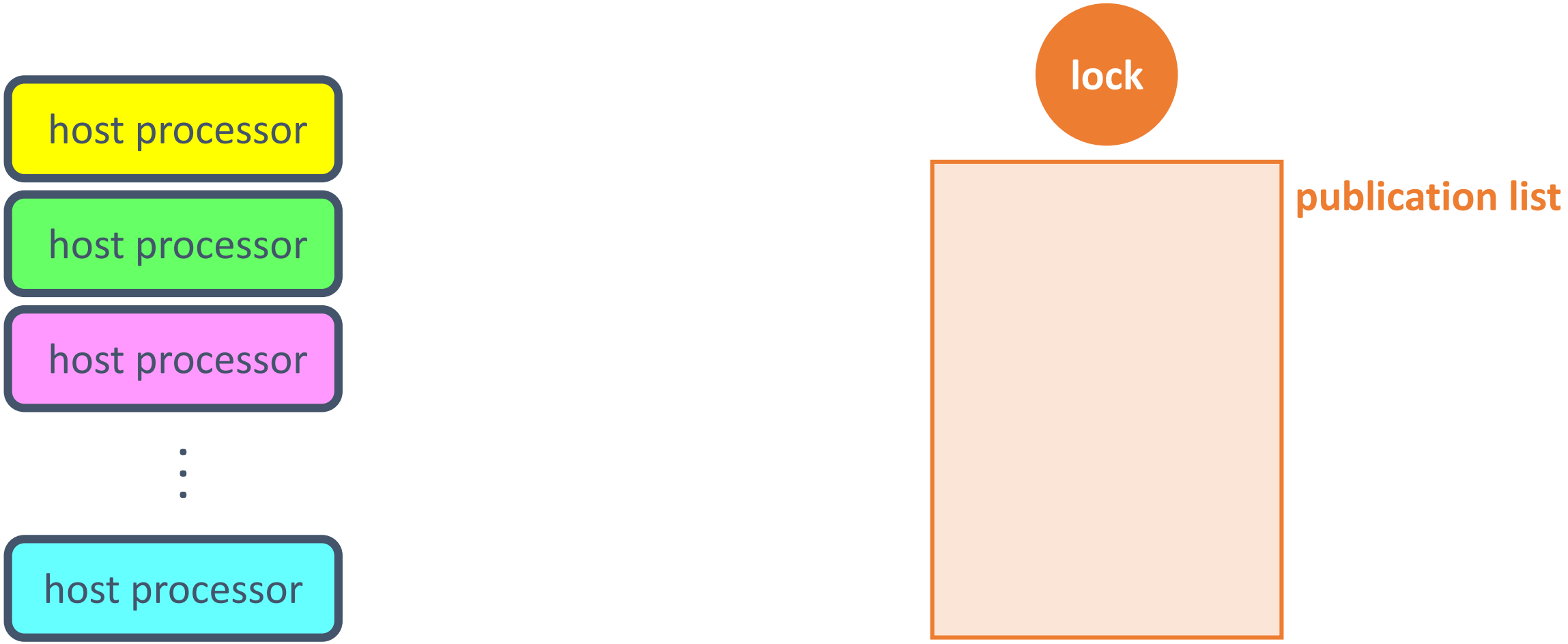
Concurrent Data Structures

- Bottlenecks on conventional architectures:
  - Poor cache locality (pointer-chasing data structures)
  - High-contention spots (contended data structures)

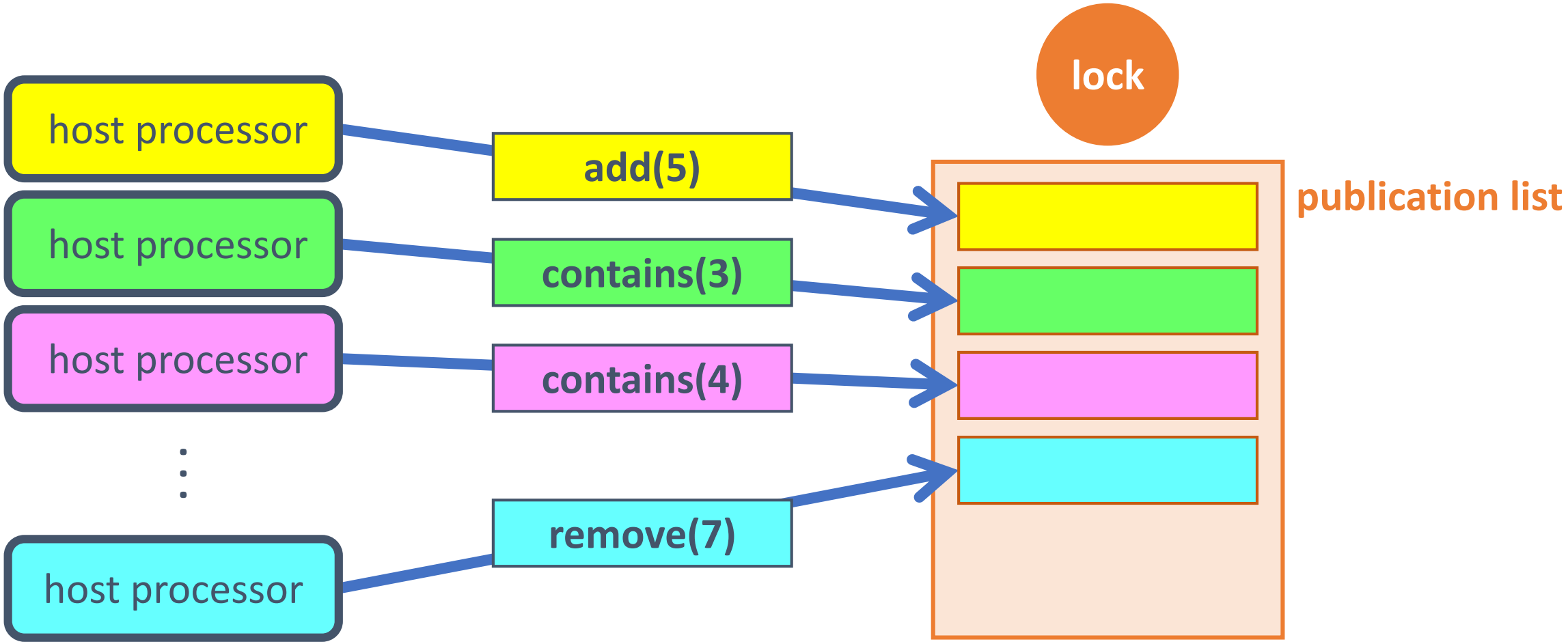
- Concurrency must be retained with NDP-based implementations

  Liu *et al*. SPAA 2017:
  - Naïve implementations on NDP will serialize data structure operations
  - Flat-combining techniques suggested

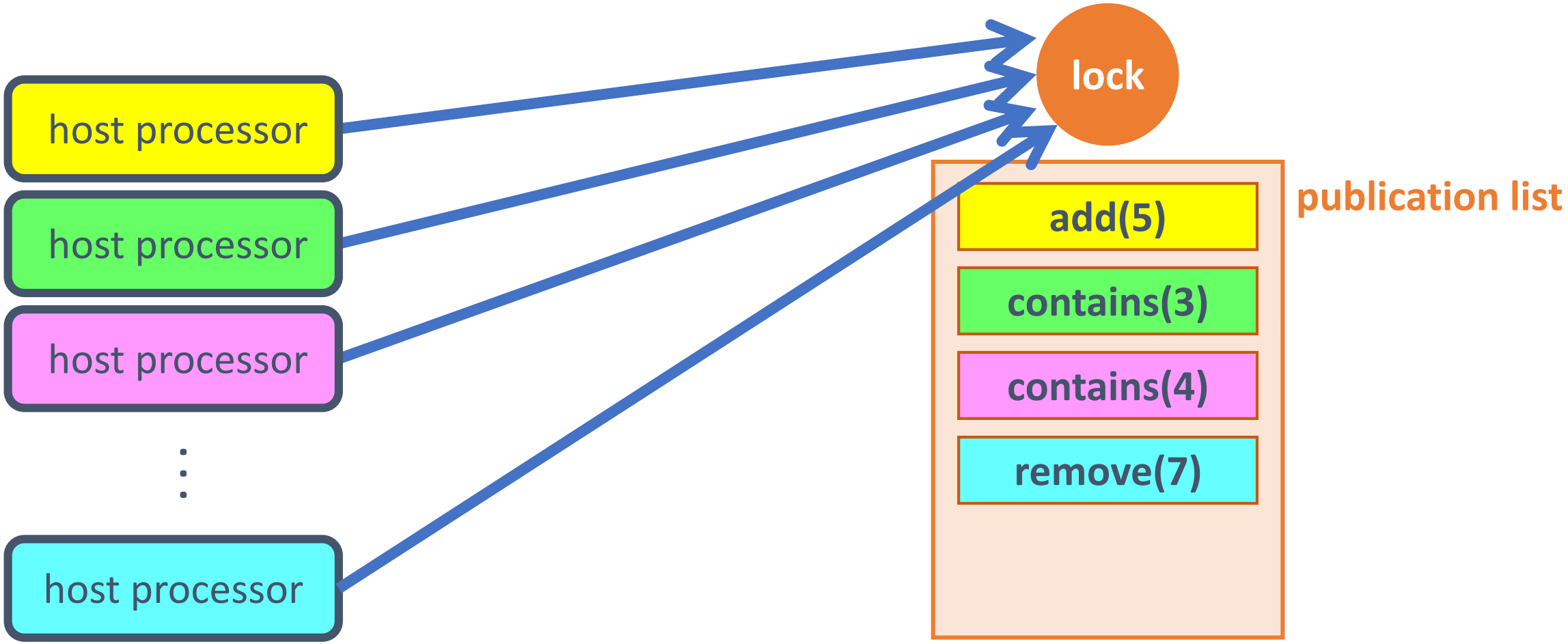
Flat-Combining (FC) Hendler *et al*. 2010



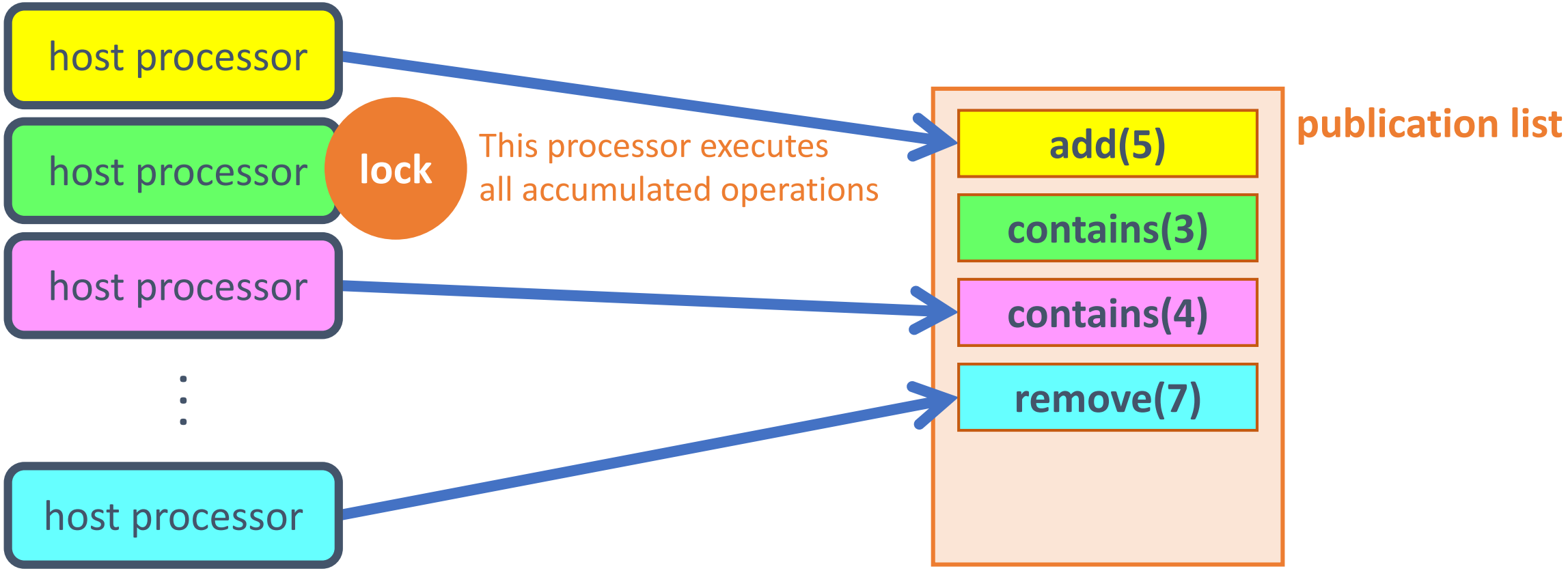
Flat-Combining (FC) Hendler *et al*. 2010



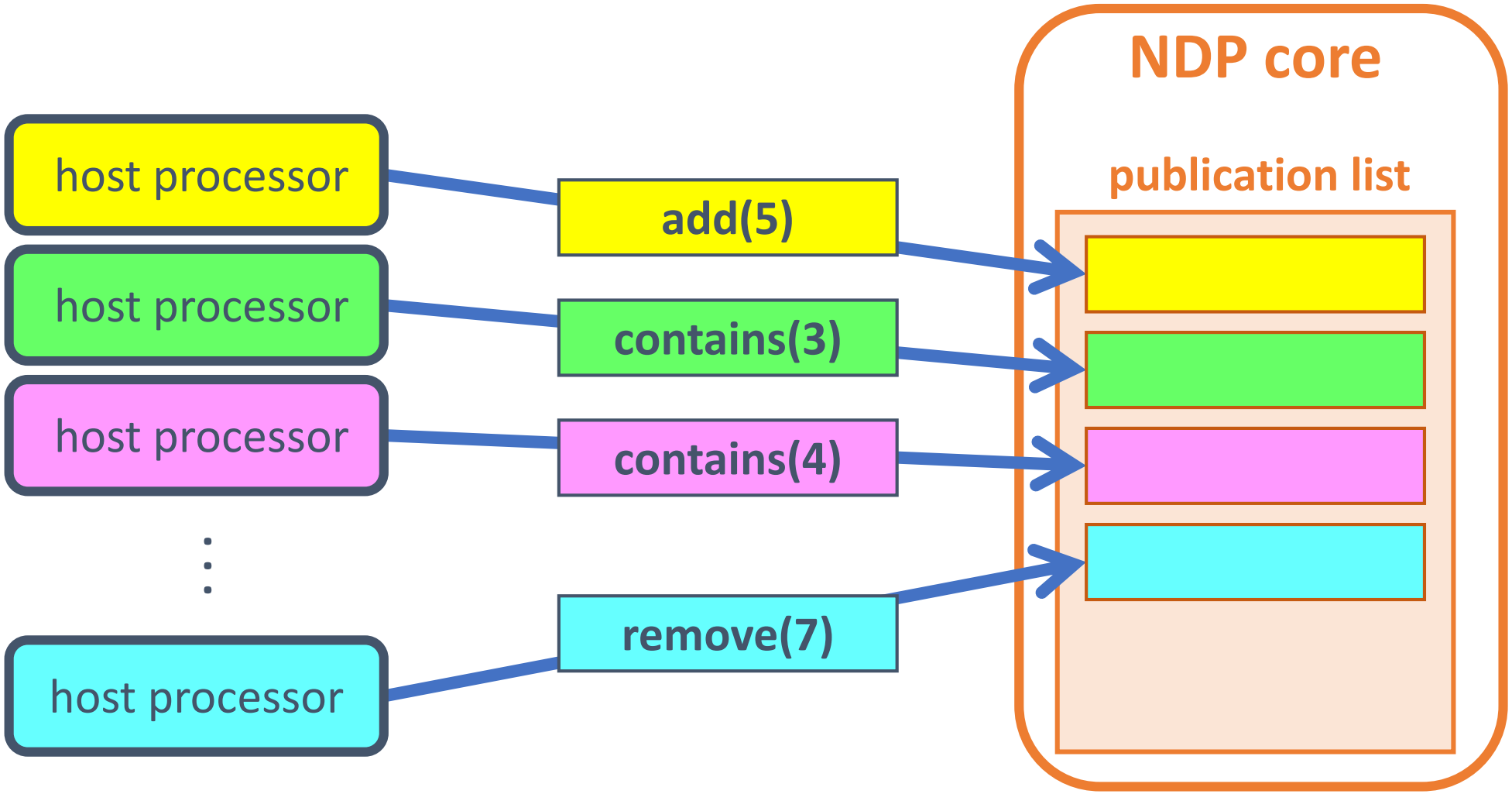
Flat-Combining (FC) Hendler *et al*. 2010



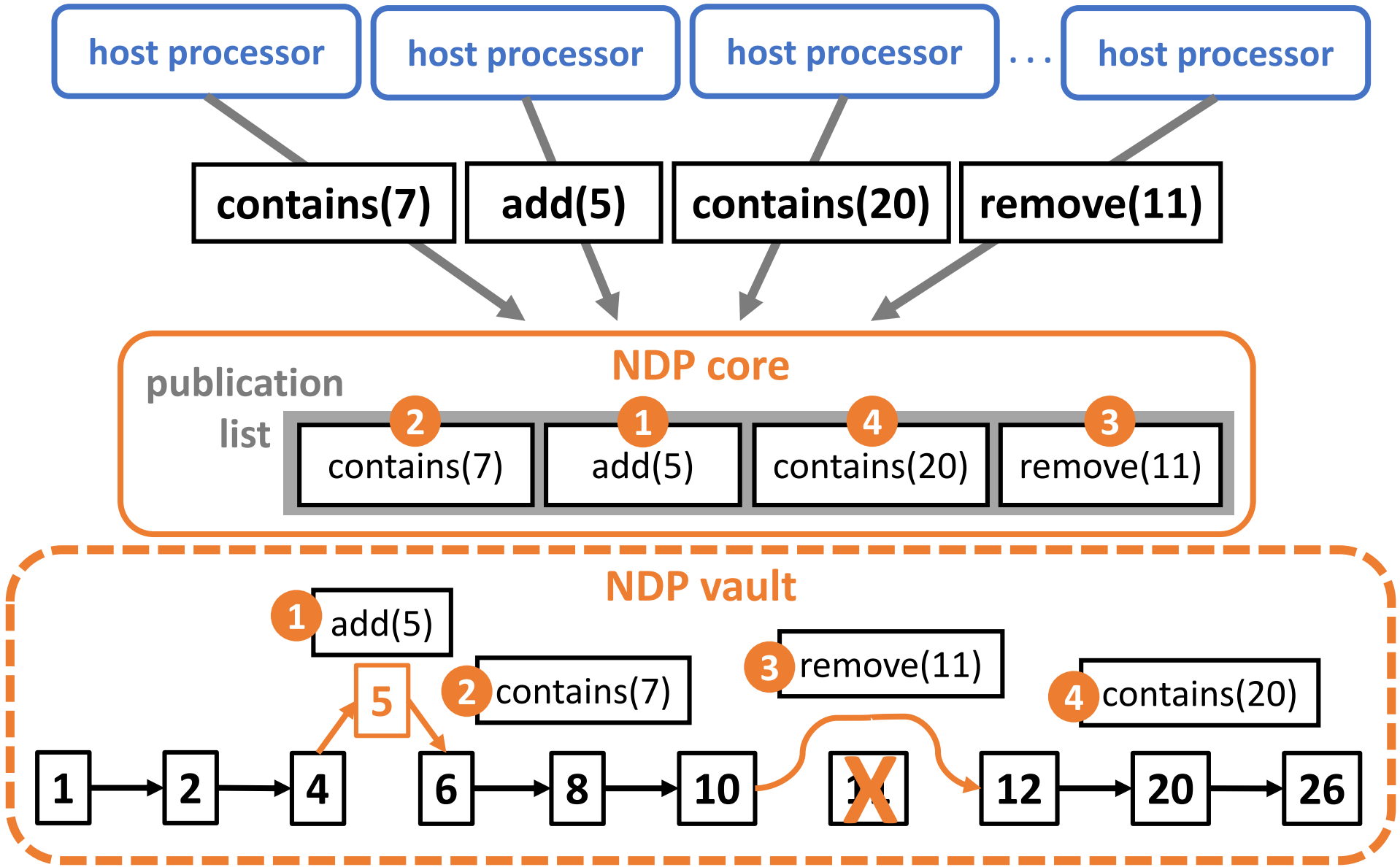
Flat-Combining (FC) Hendler *et al*. 2010



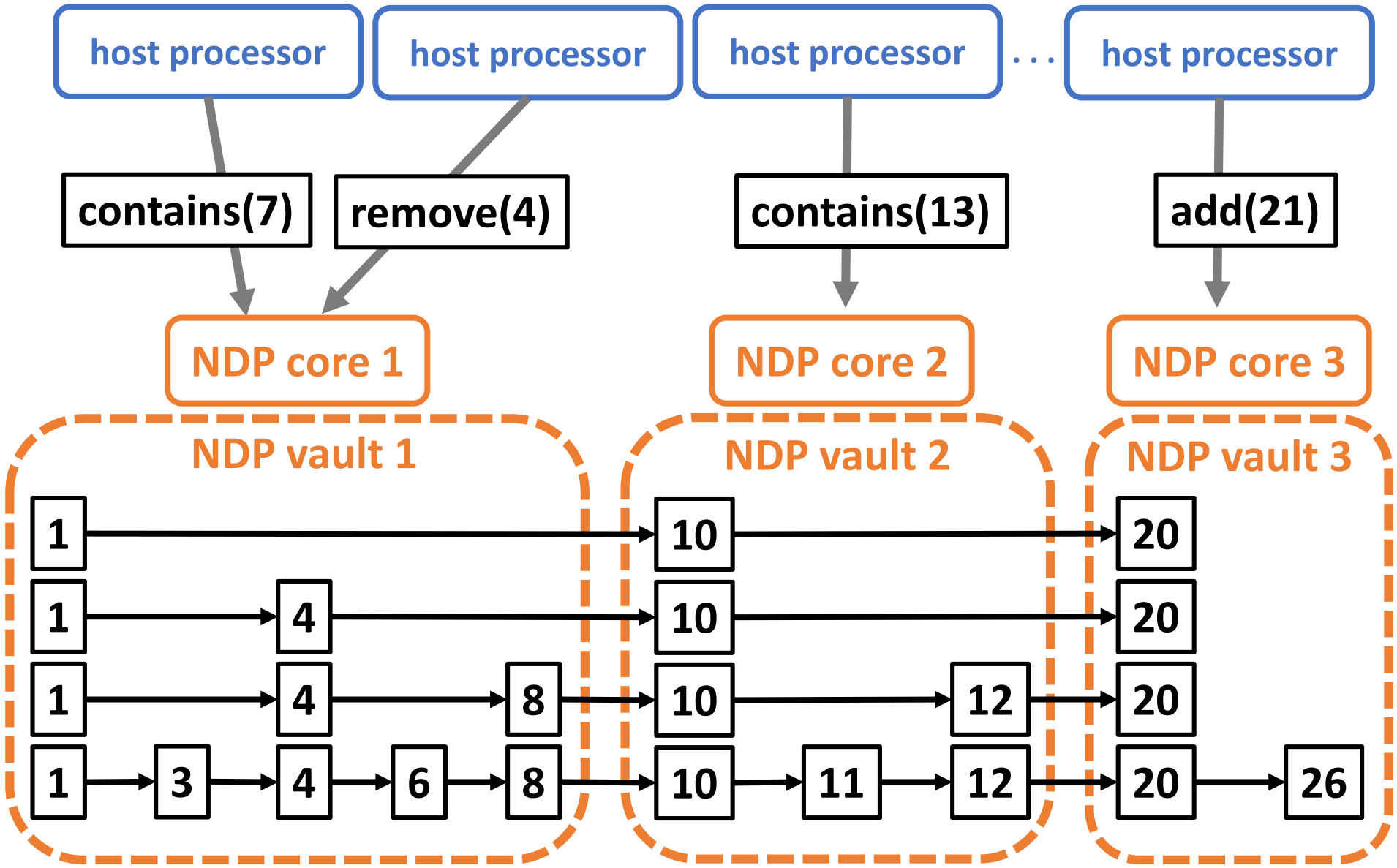
Flat-Combining with NDP Liu *et al*. 2017



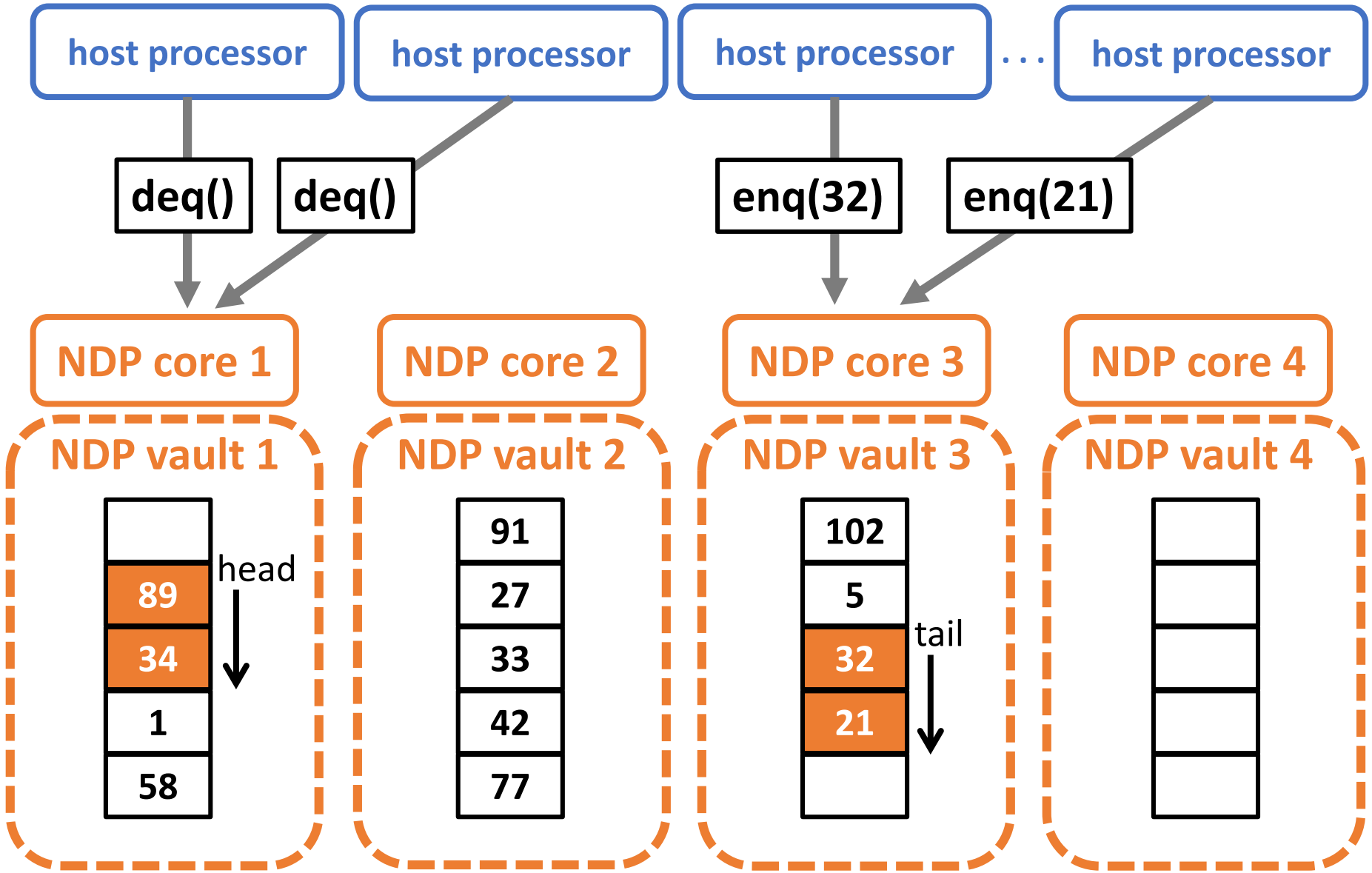
NDP-based Linked-List Liu et al. 2017



NDP-based Skiplist Liu et al. 2017



NDP-based FIFO Queue Liu et al. 2017



Limitations of Prior Work

Relied only on theoretical analysis with simple assumptions:

1. Overlooked cache impacts in host-based data structures
2. Overly optimistic assumptions on NDP core's data access

Limitations of Prior Work

Relied only on theoretical analysis with simple assumptions:

1. Overlooked cache impacts in host-based data structures
2. Overly optimistic assumptions on NDP core's data access

Limitations of Prior Work

Relied only on theoretical analysis with simple assumptions:

1. Overlooked cache impacts in host-based data structures
2. Overly optimistic assumptions on NDP core's data access

Limitations of Prior Work

Relied only on theoretical analysis with simple assumptions:

1. Overlooked cache impacts in host-based data structures
2. Overly optimistic assumptions on NDP core's data access

NDP does not magically remove all DRAM access latencies!!

Limitations of Prior Work

Relied only on theoretical analysis with simple assumptions:

1. Overlooked cache impacts in host-based data structures
2. Overly optimistic assumptions on NDP core's data access

Lightweight hardware changes significantly improve NDP-based data structure performance.

DRAM Access Latency

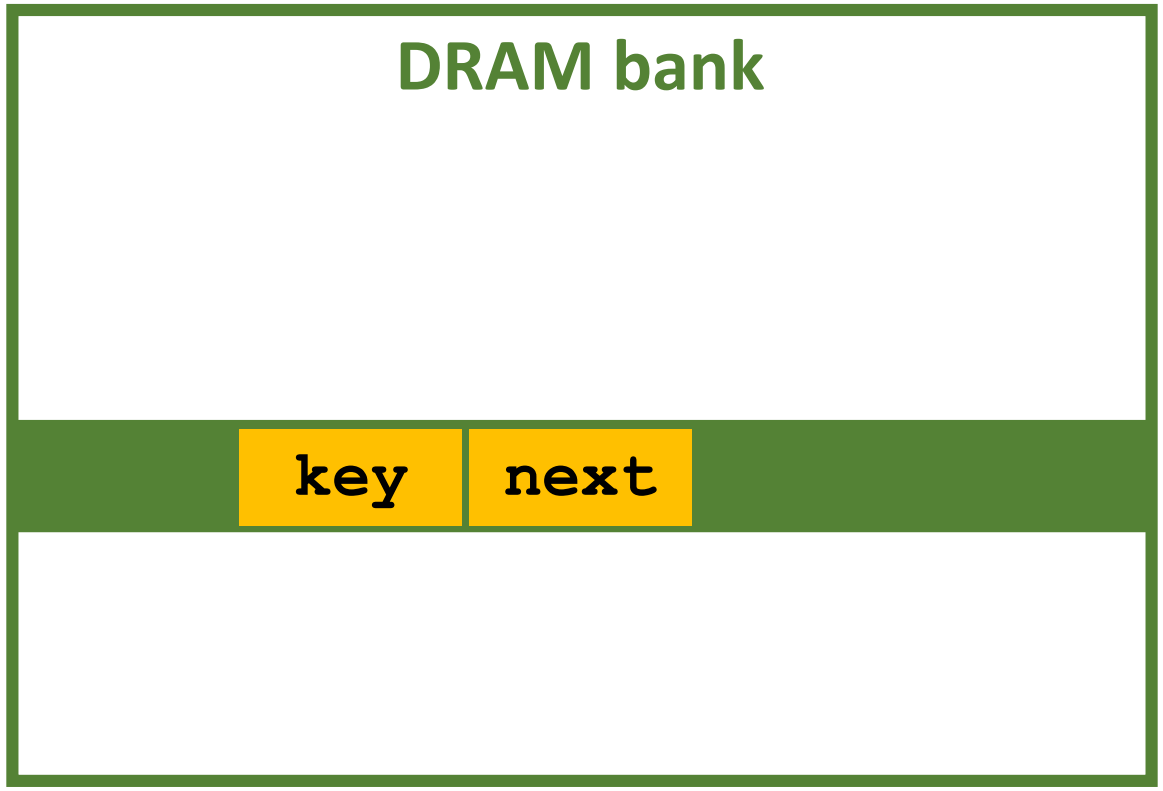
linked-list node definition:

```
struct node {
   uint32_t key;       // 4 bytes
   struct node *next; // 4 bytes
};
```

DRAM Access Latency

linked-list node definition:

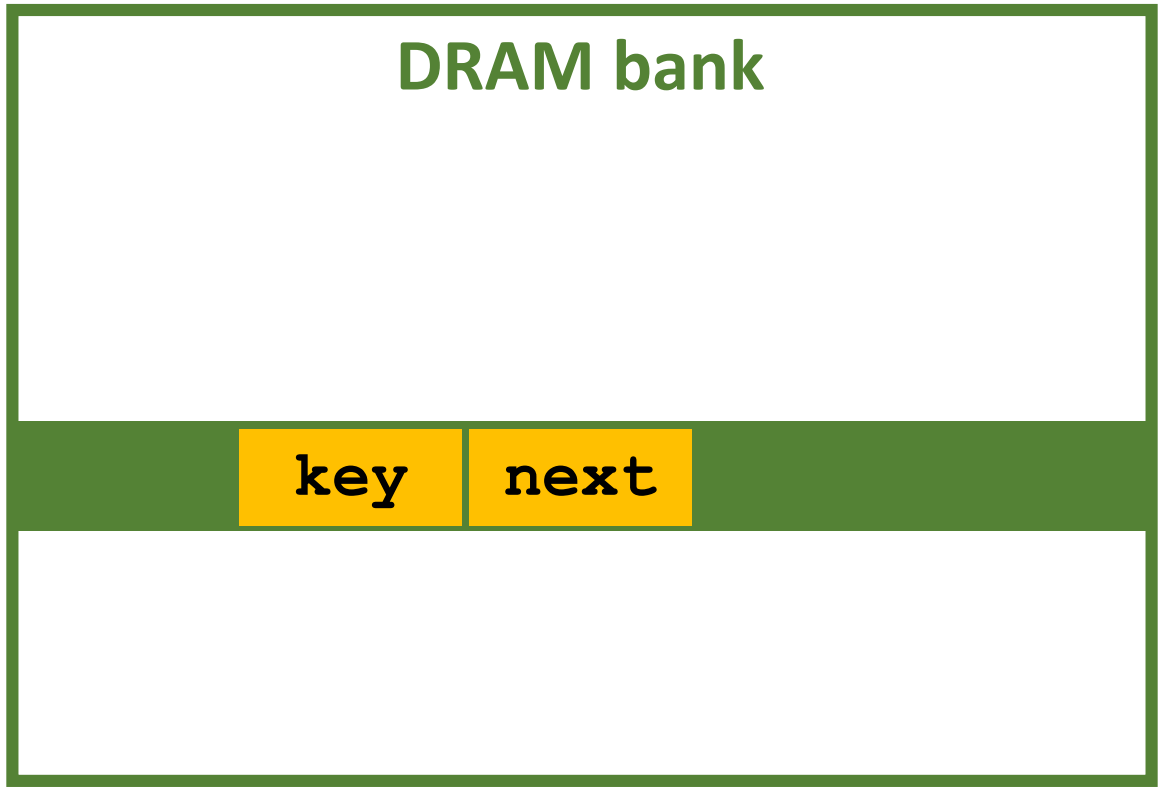
```
struct node {
   uint32_t key;        // 4 bytes
   struct node *next; // 4 bytes
};
```



DRAM Access Latency

linked-list traversal:

```
while (curr_node->key < param){
   curr_node = curr_node->next;
}
```

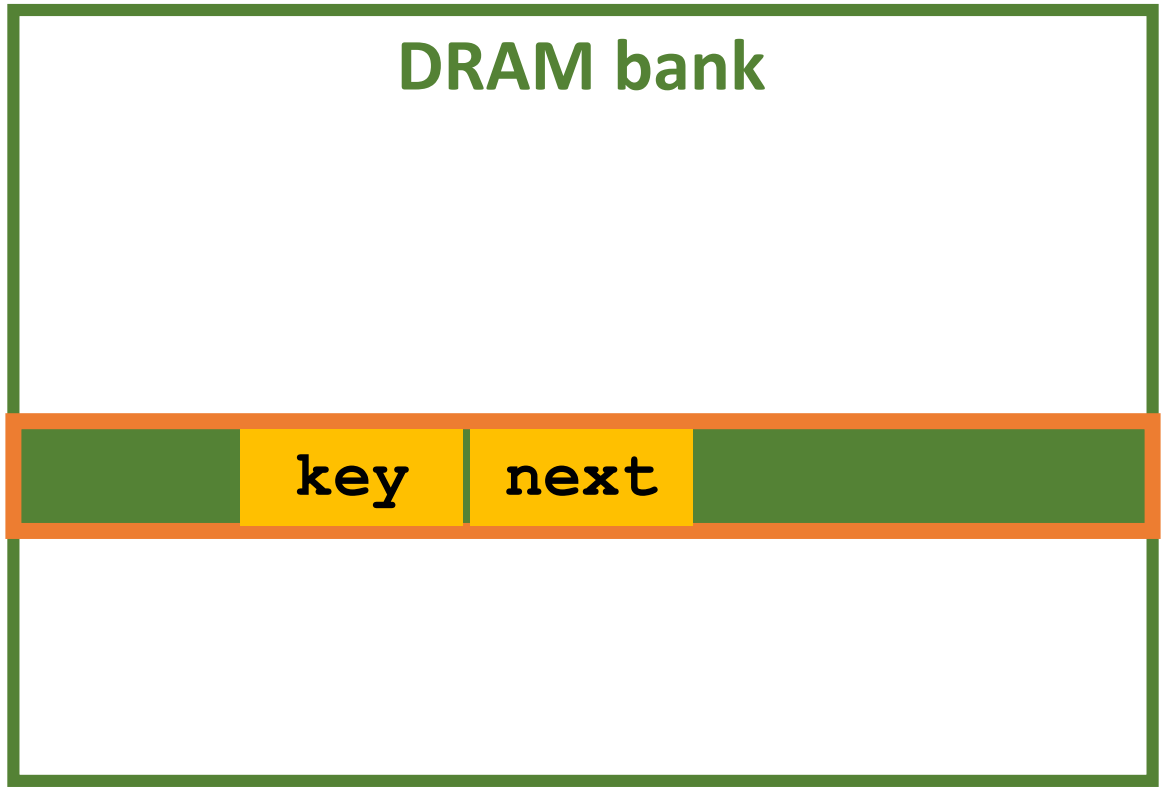


DRAM Access Latency

linked-list traversal:

```
while (curr_node->key < param){
   curr_node = curr_node->next;
}
```

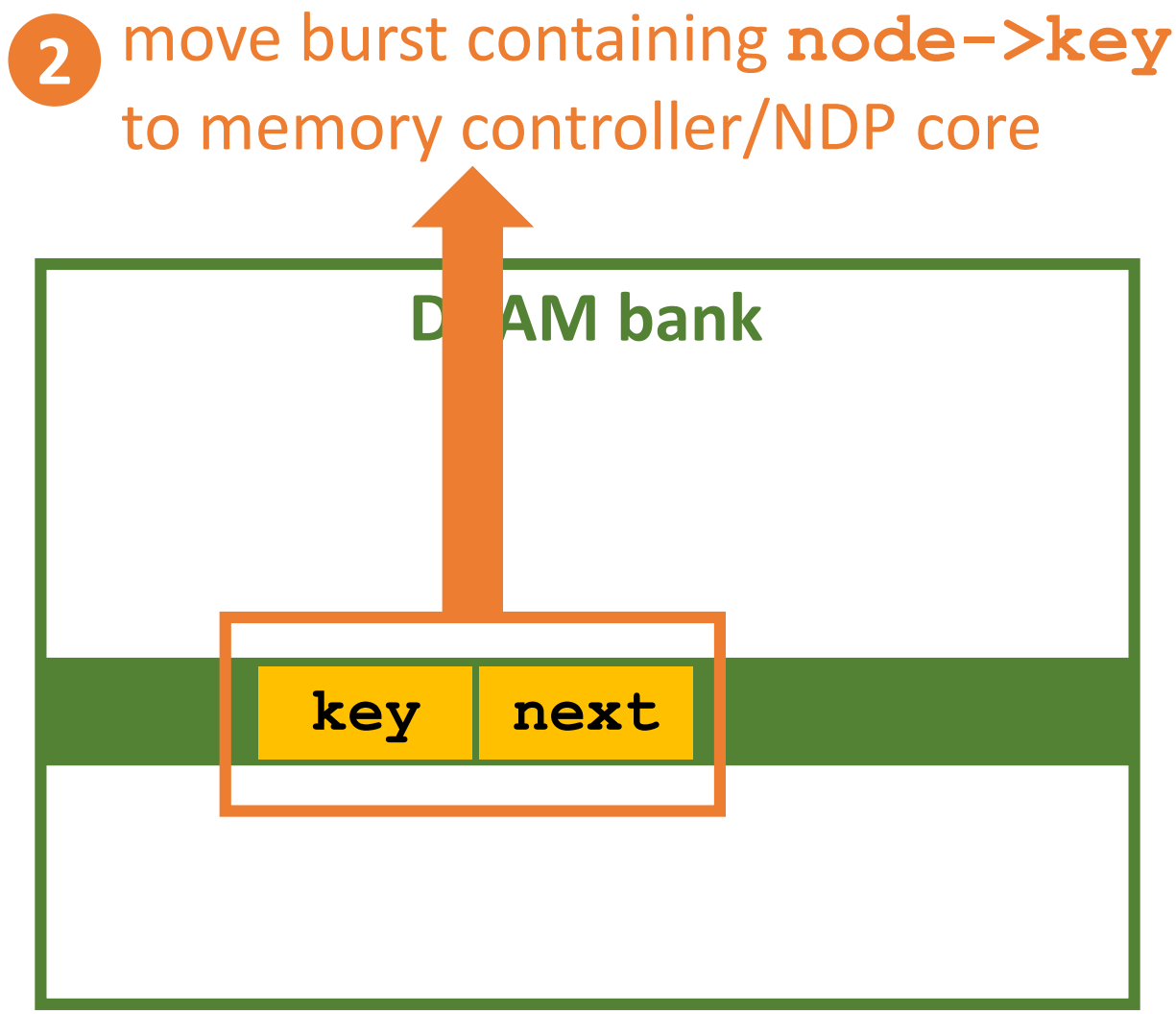
1. activate row containing `node->key`



DRAM Access Latency

linked-list traversal:

```
while (curr_node->key < param){
    curr_node = curr_node->next;
}
```

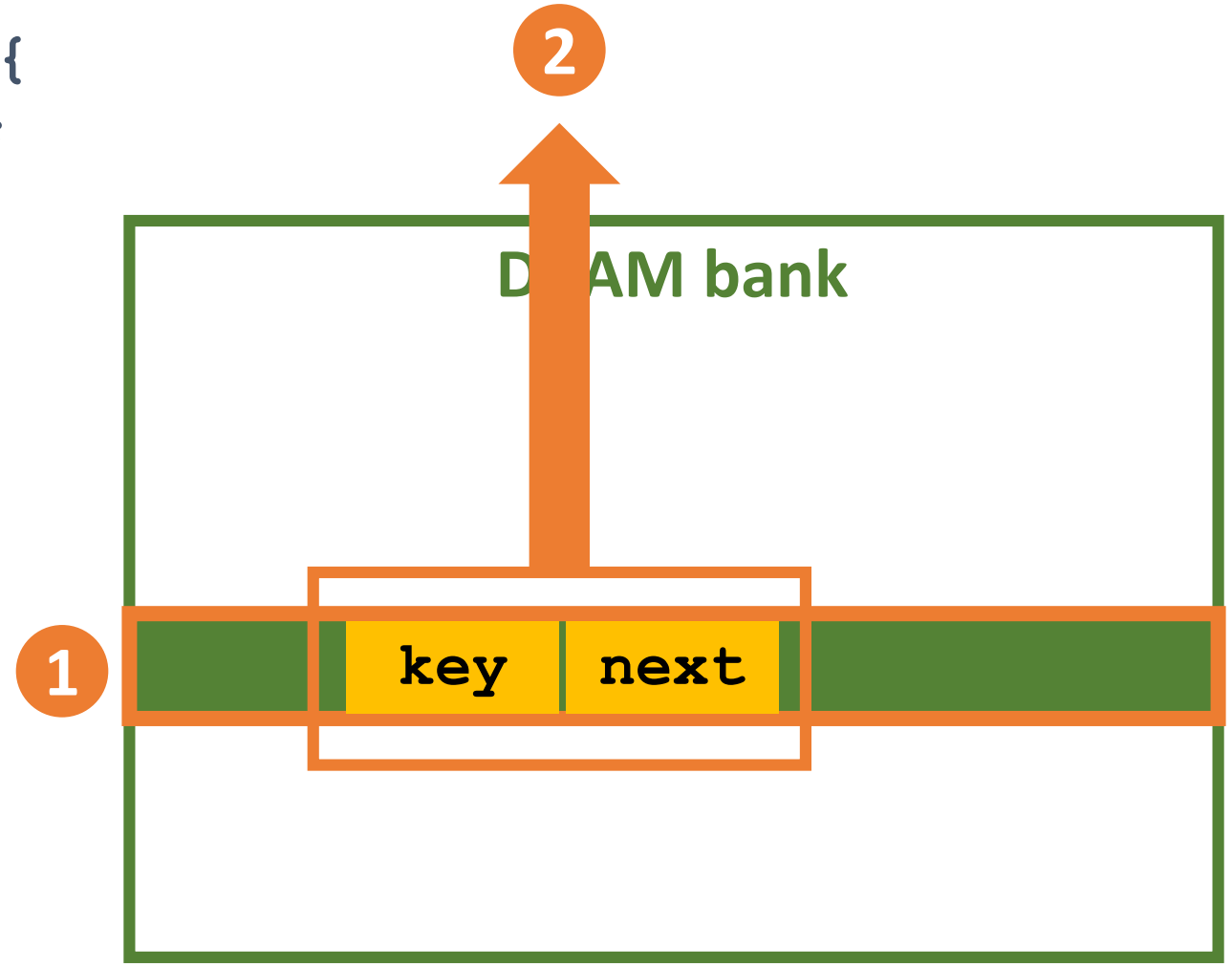


DRAM Access Latency

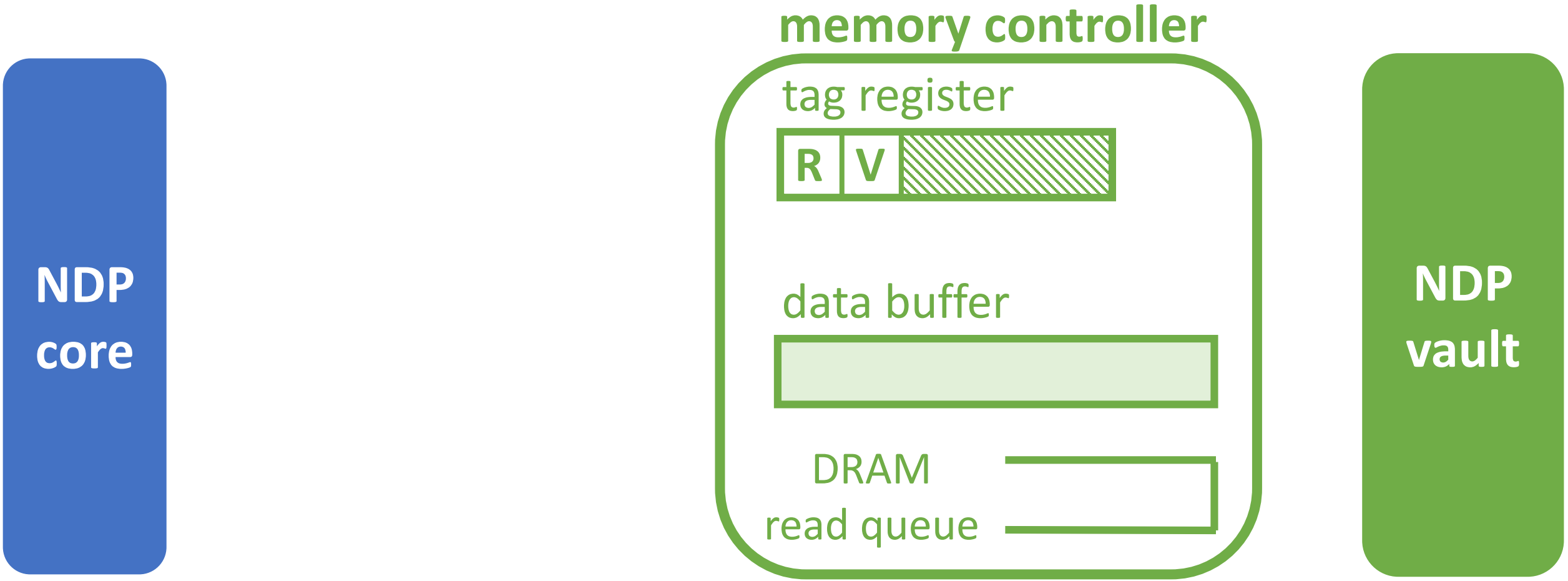
linked-list traversal:

```
while (curr_node->key < param){
   curr_node = curr_node->next;
}
```

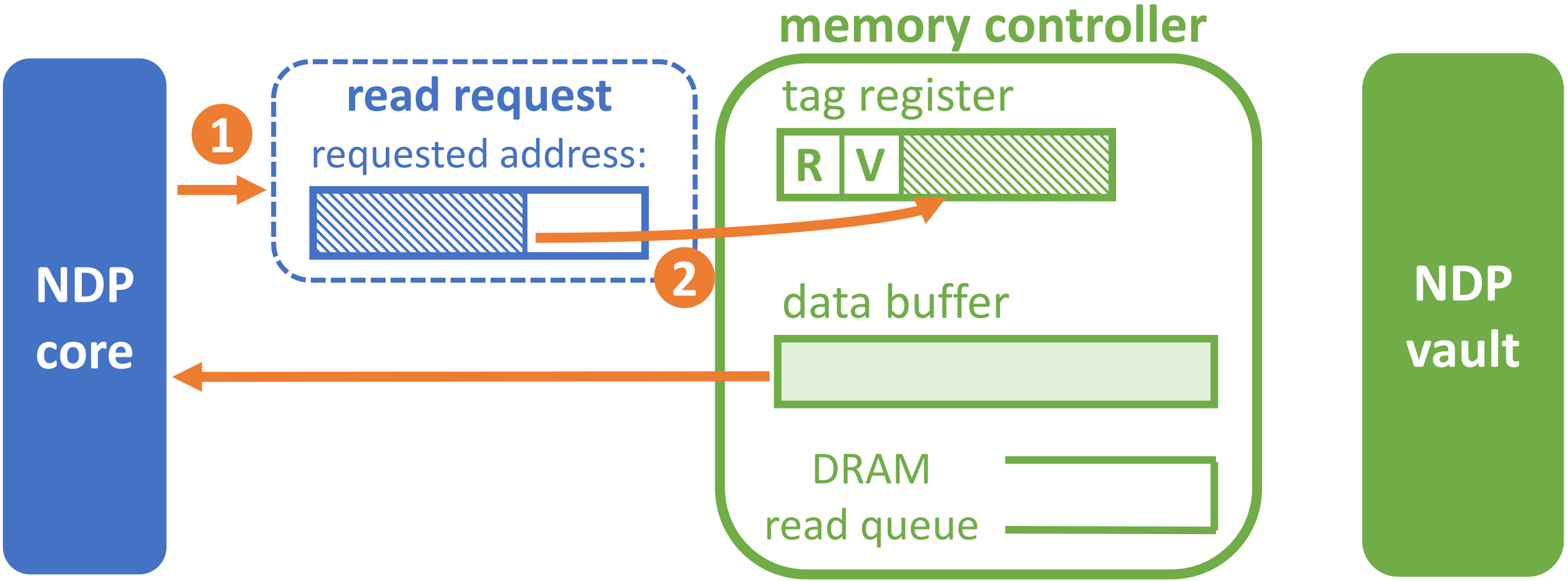
1, 2 repeated for `node->next`



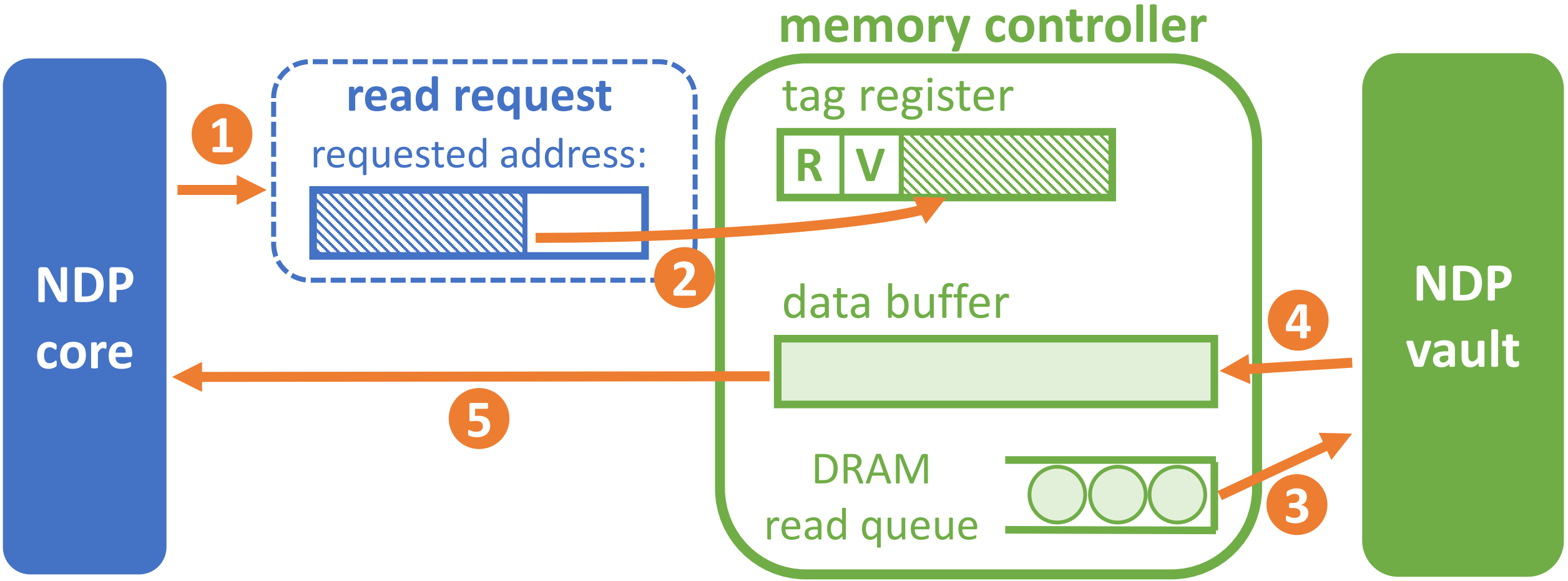
Memory Controller Design



Memory Controller Design



Memory Controller Design



Results

Operation throughput = # data structure operations / second

NDP data buffer compared against:

- **NDP original**: NDP-based implementation w/o hardware change
- **Host FC**: host-based equivalent of NDP-based algorithm
- Host-based state-of-the-art concurrent data structure

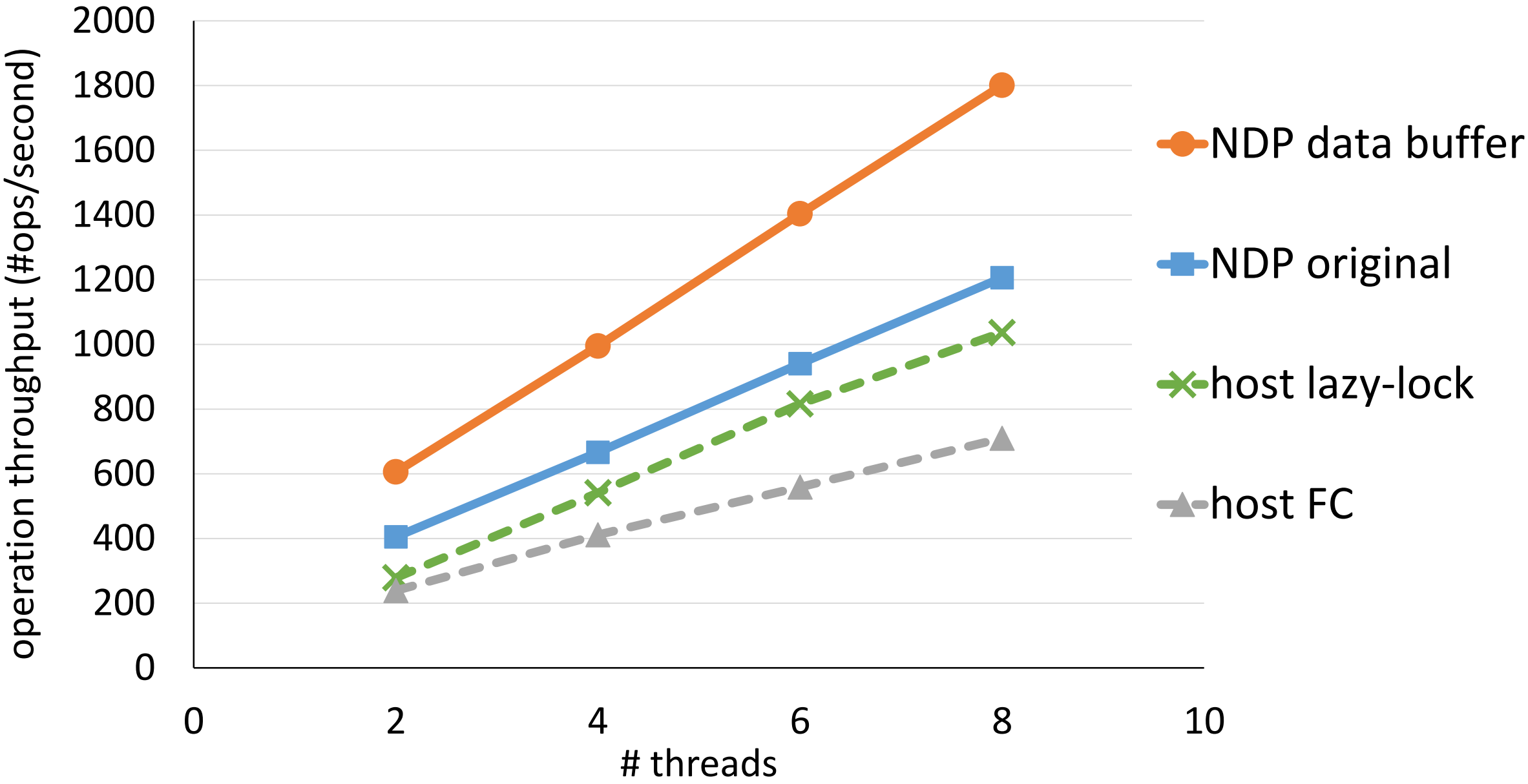
Results

Operation throughput = # data structure operations / second

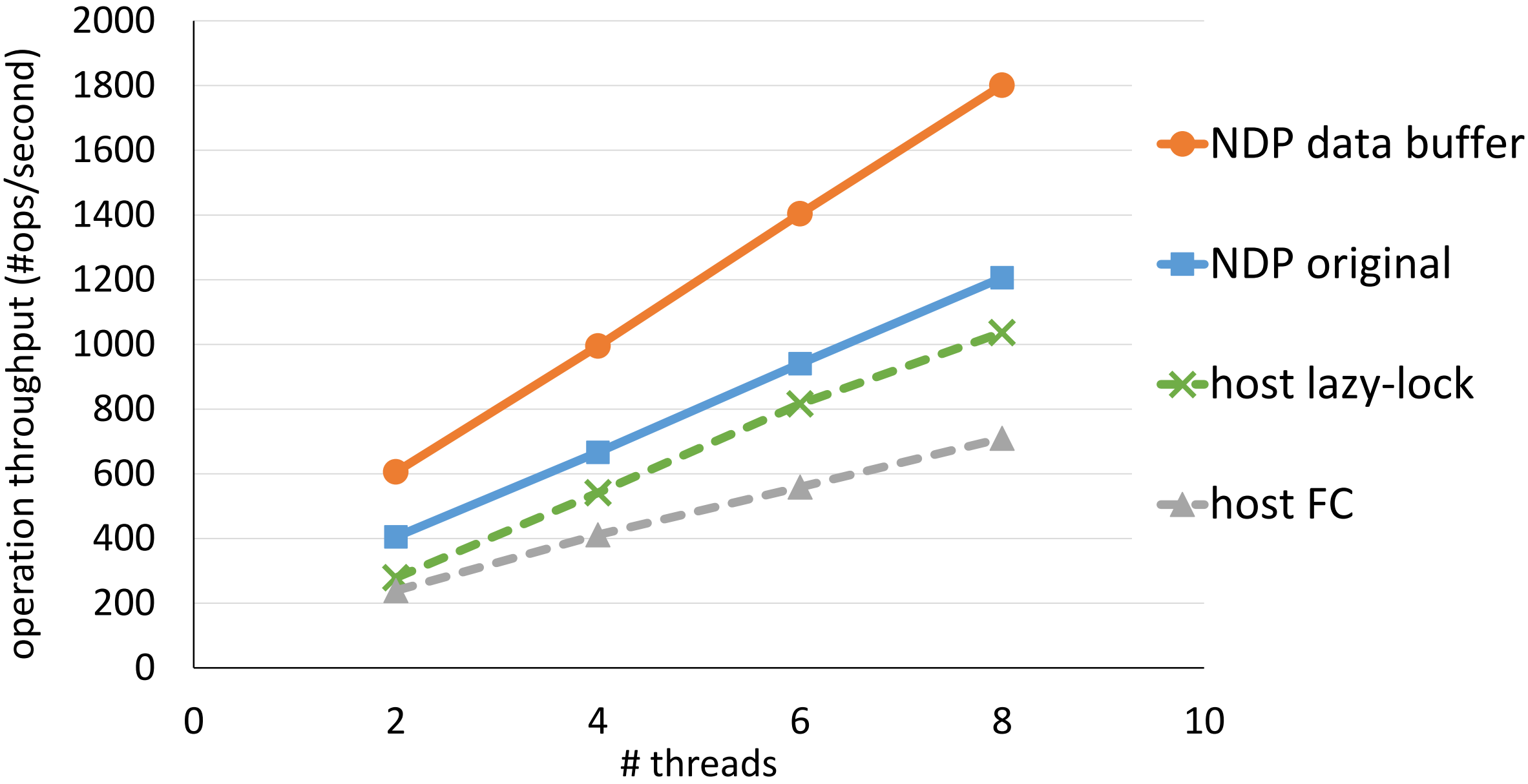
NDP data buffer compared against:

- **NDP original**: NDP-based implementation w/o hardware change
- **Host FC**: host-based equivalent of NDP-based algorithm
- Host-based state-of-the-art concurrent data structure

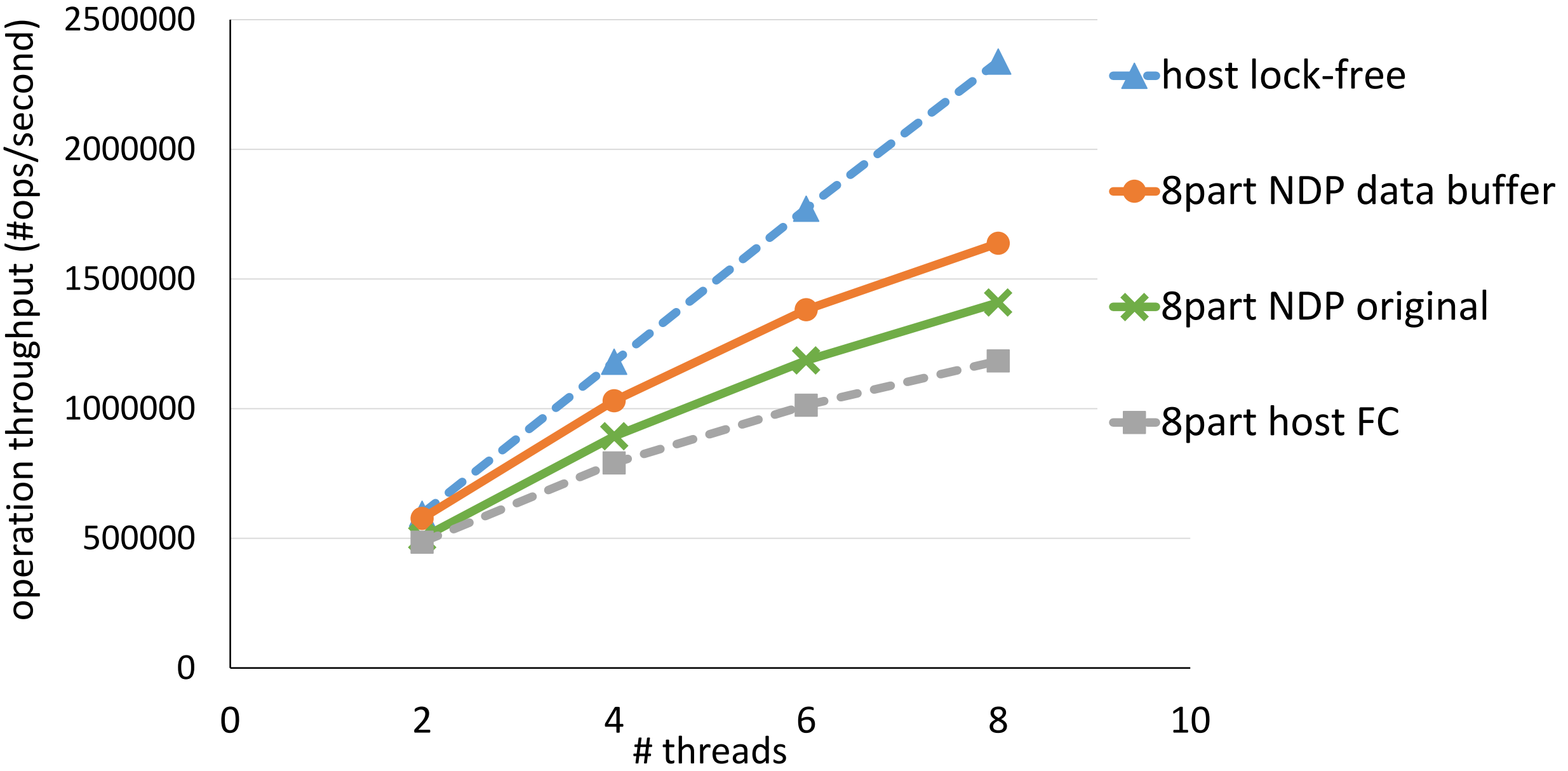
Linked-List Results



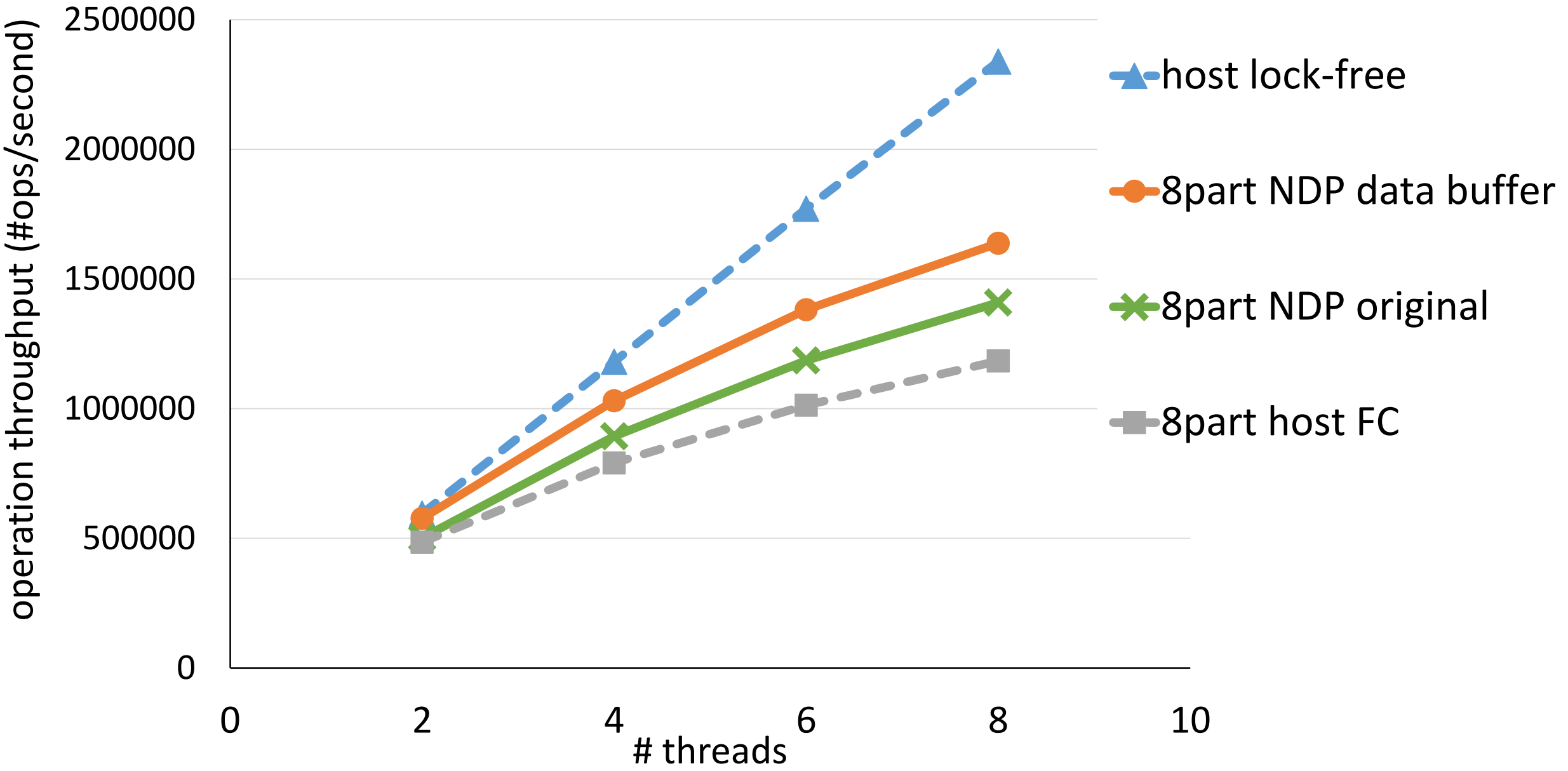
Linked-List Results



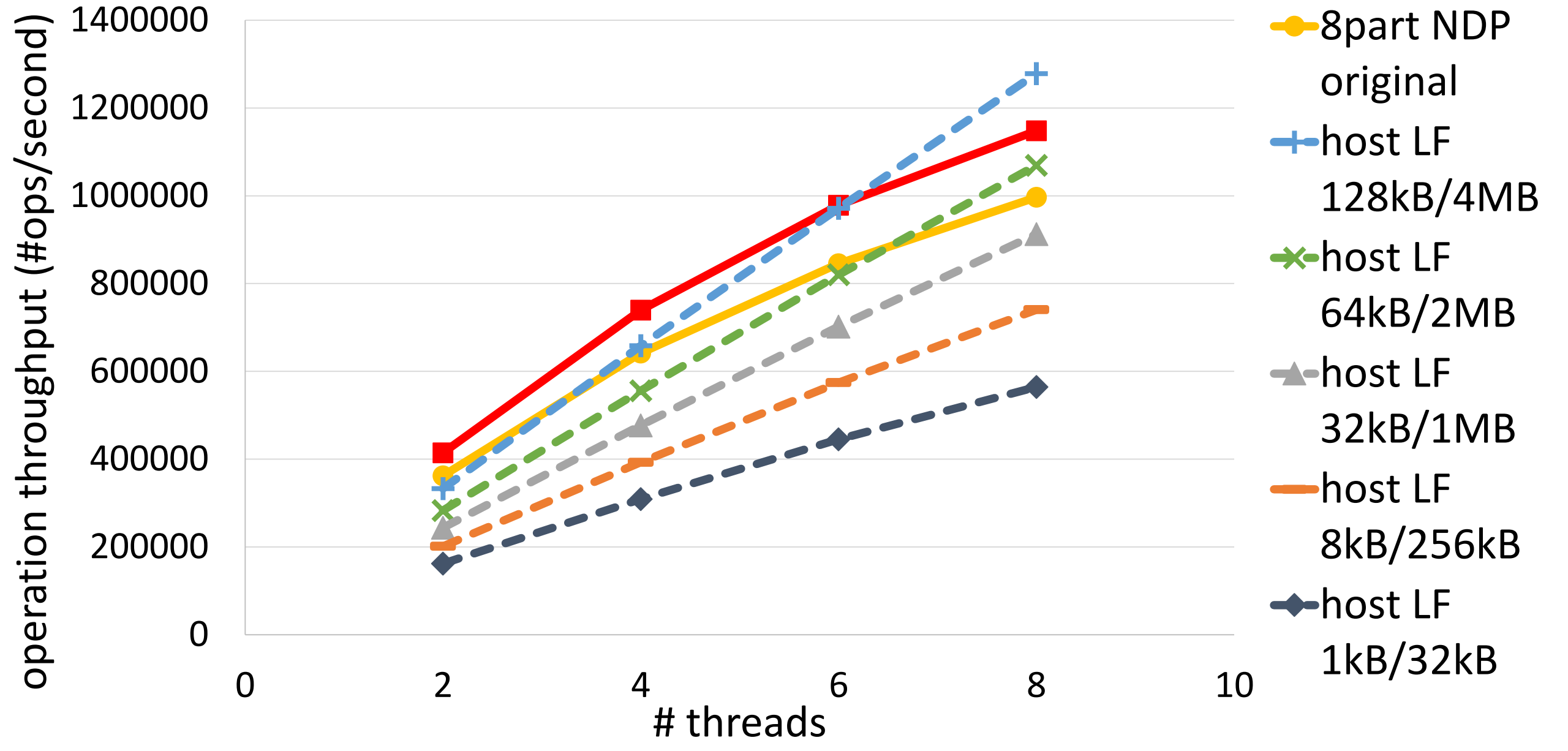
Skiplist Results



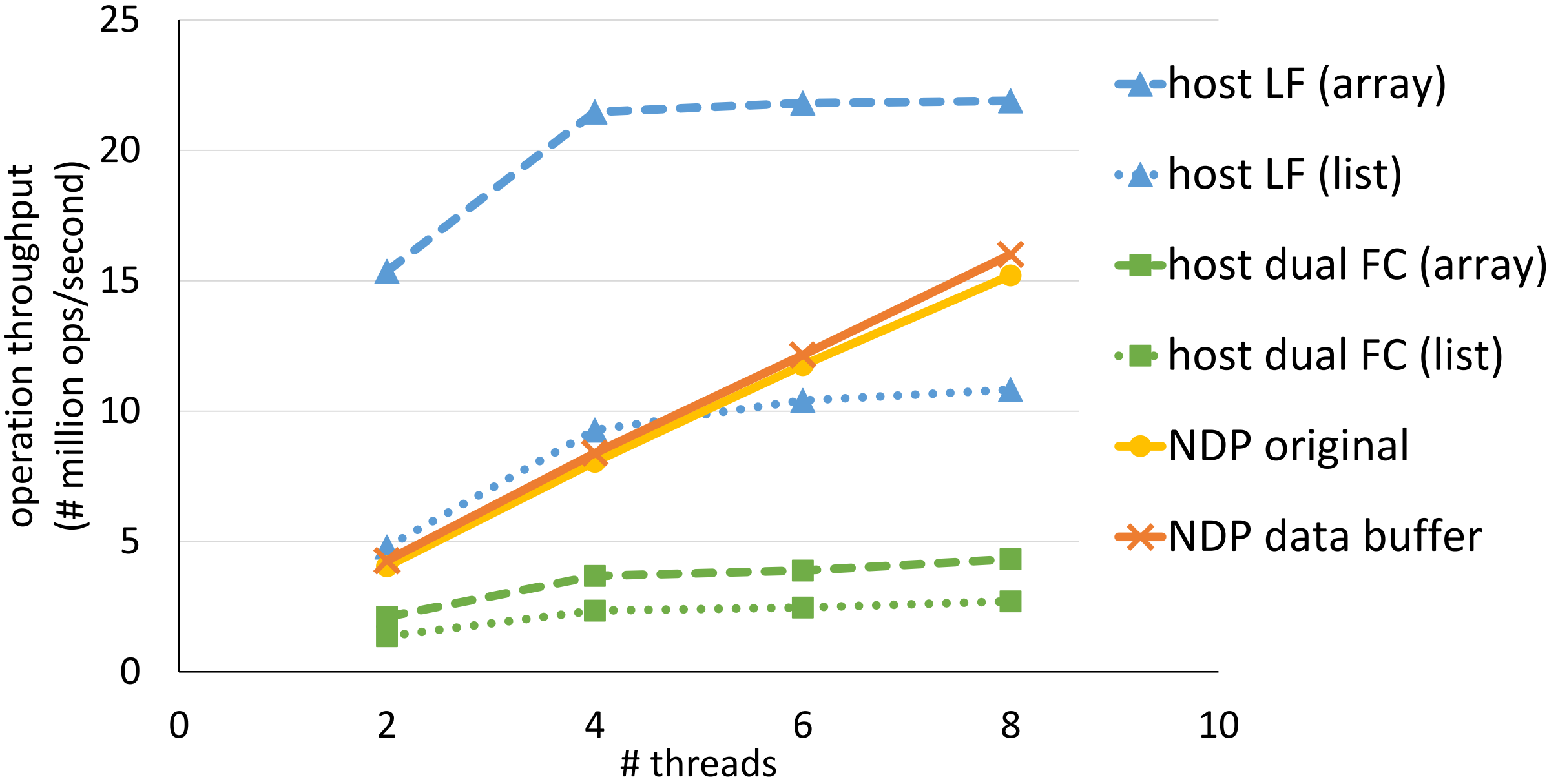
Skiplist Results



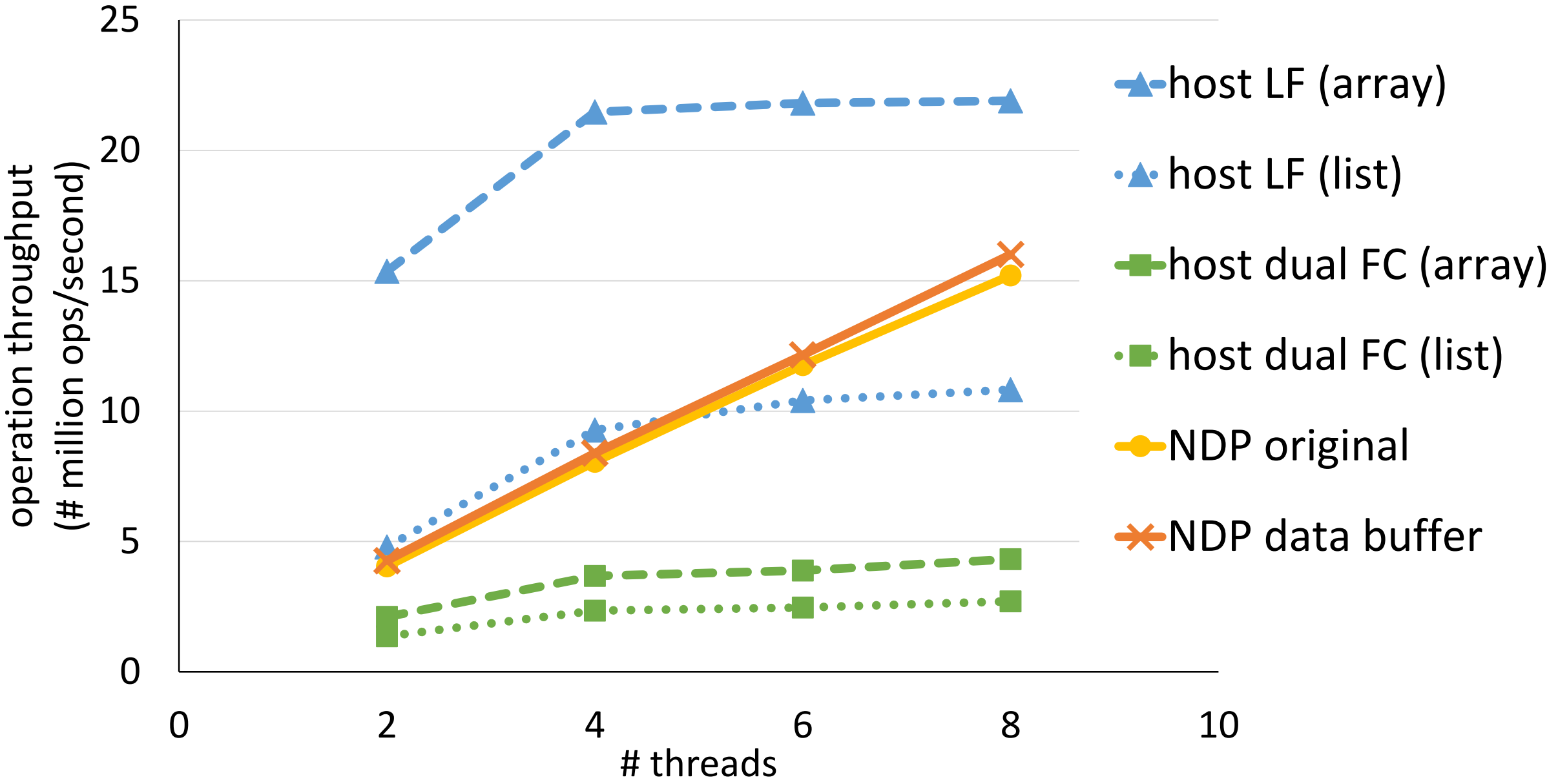
Skiplist Results



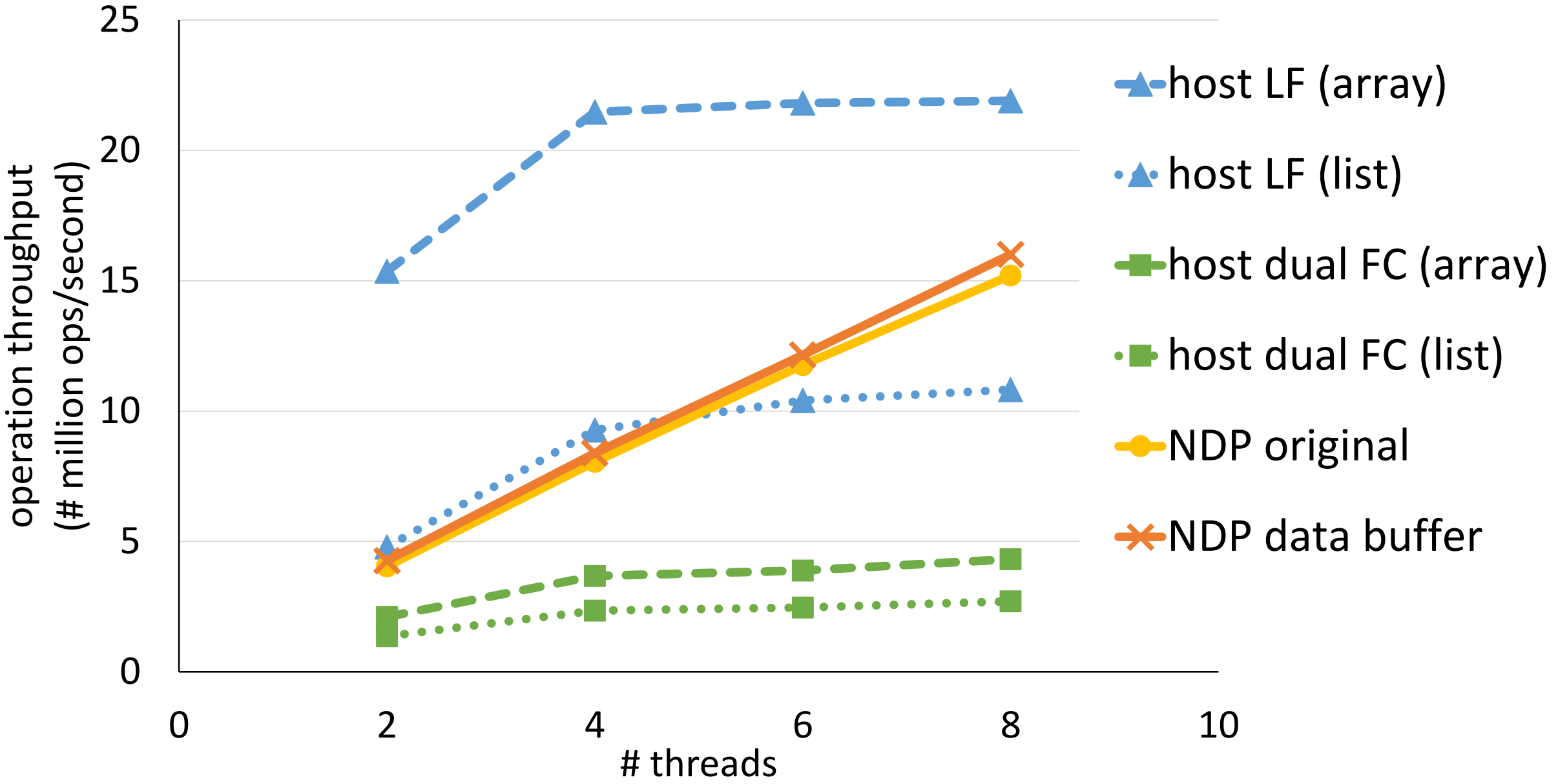
FIFO Queue Results



FIFO Queue Results



FIFO Queue Results



Summary

High performance concurrent data structures w/ NDP

- NDP does not remove DRAM access latencies completely
- Lightweight HW change significantly improves performance
  - Data buffer in memory controller acts as single block cache
  - **Performance improvement compared to w/o data buffer:**
    50% (linked-list), 17% (skiplist), 5% (FIFO queue)