A Fast Window Join over Data Streams

Hideyuki Kawashima

Presented at
ICNC'11, FPL'11, MCSoC'12, SSDBM'13.

Data Stream Management System (DSMS)

How many packets are arrived for port 80 in a minute ?

```
SELECT  COUNT(*)
FROM    eth0[TIME 1 MIN]
WHERE   port = 80
```

20

Q1

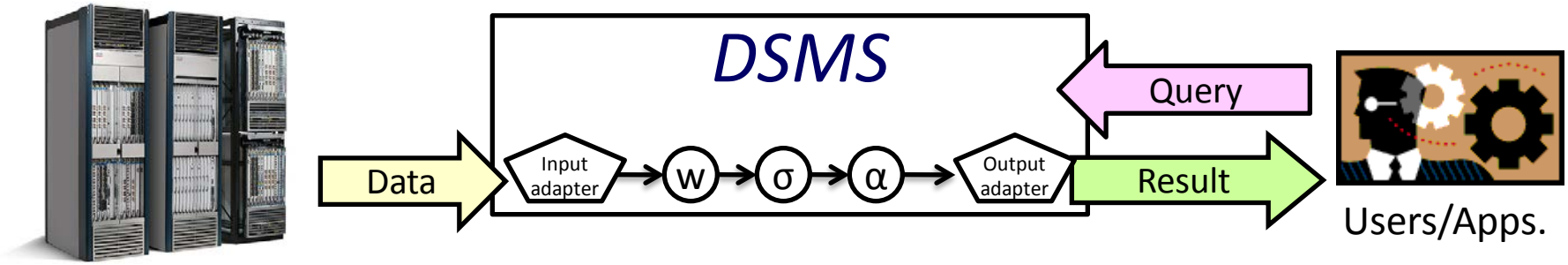
Relation eth0

DSMS

Relational schema

- Destination IP
- Source IP
- Destination Port
- Source Port
- Interface (e.g. eth0)
- Length
- Version（e.g. IPV4 ）
- Payload

```
SELECT  COUNT(*)
FROM    eth0[TIME 1 MIN]
WHERE  port = 80
```

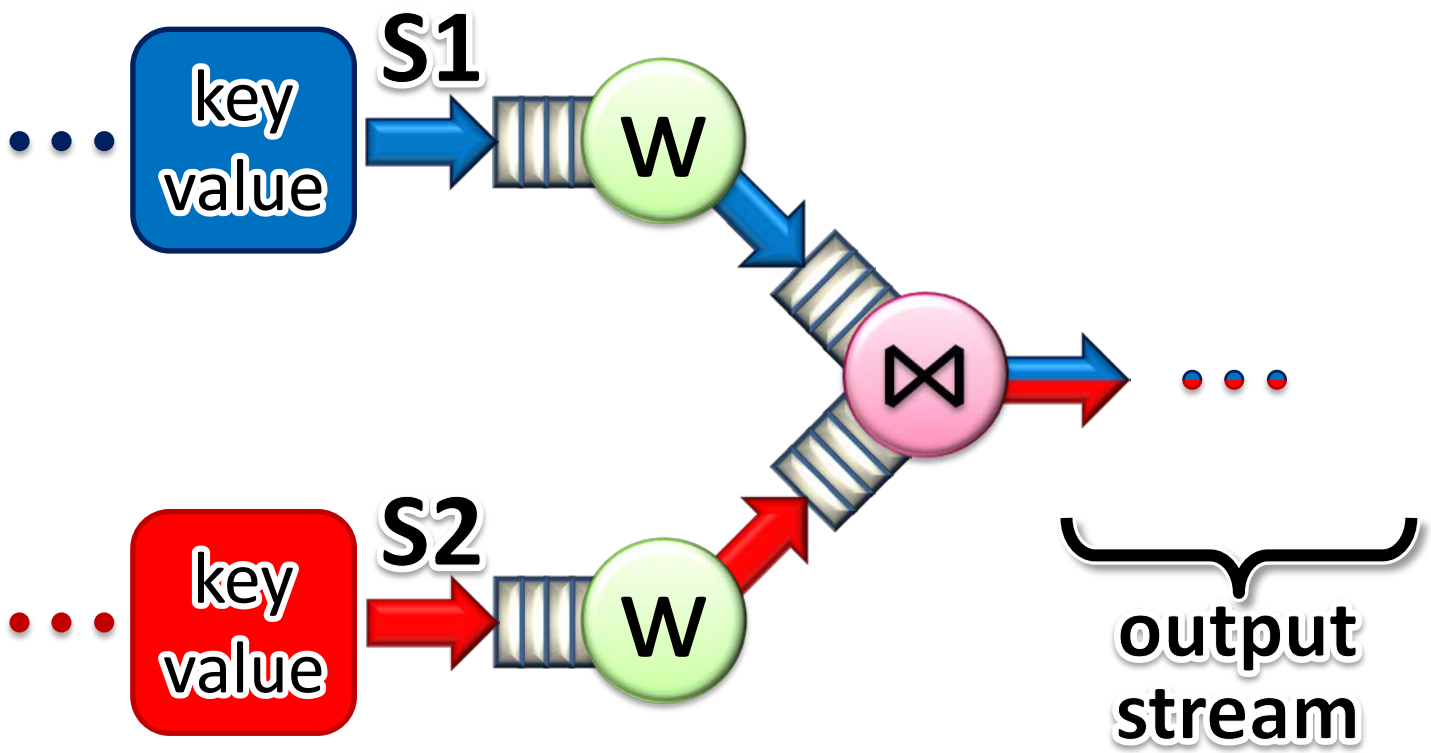


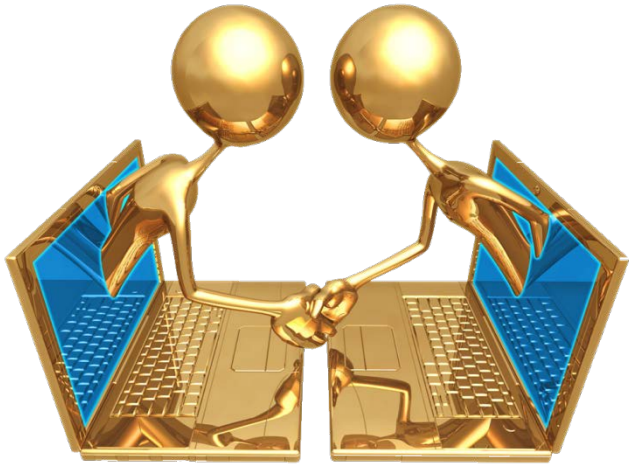
- SQL is translated to operator tree.
- On arrival of data, tree is evaluated.
- Operators are based on relational database
  - w(Window) : Cutting off relations from a stream
  - σ (Selection) : Filter
  - α (Aggregation) : such as AVG, MIN, MAX

Window Join

SELECT	*
FROM	S1 [Rows 100], S2 [Rows 100]
WHERE	S1.key = S2.key

CQL Query





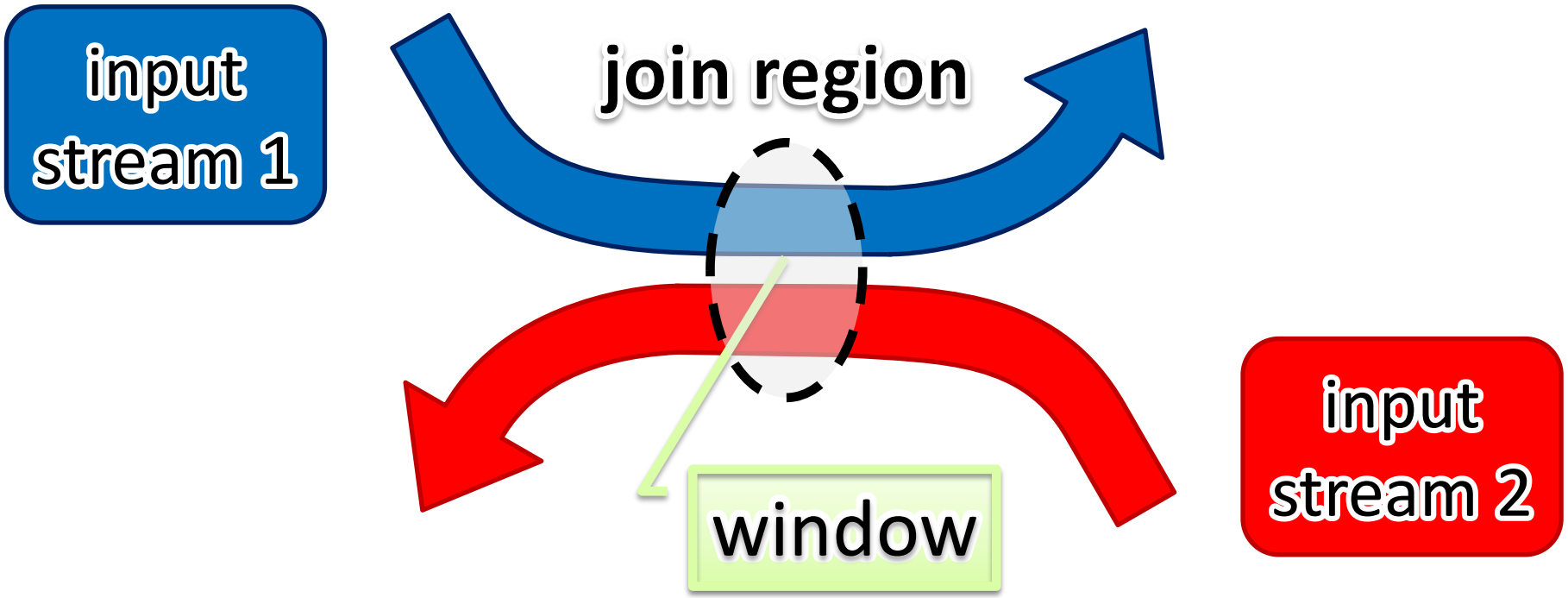
HANDSHAKE JOIN

J. Teubner and R. Müller, SIGMOD'11

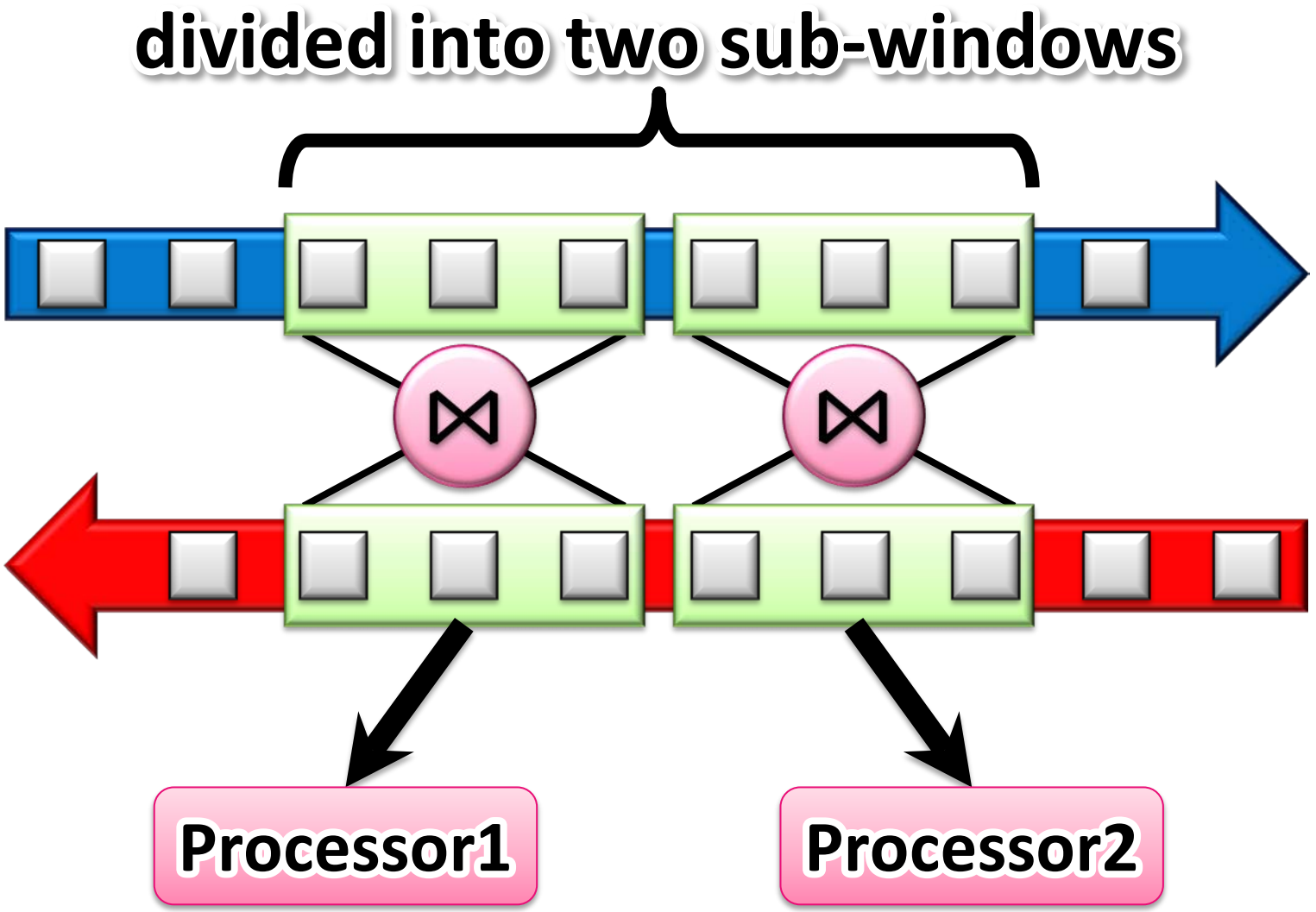
Handshake Join

basic idea Flow in opposite direction

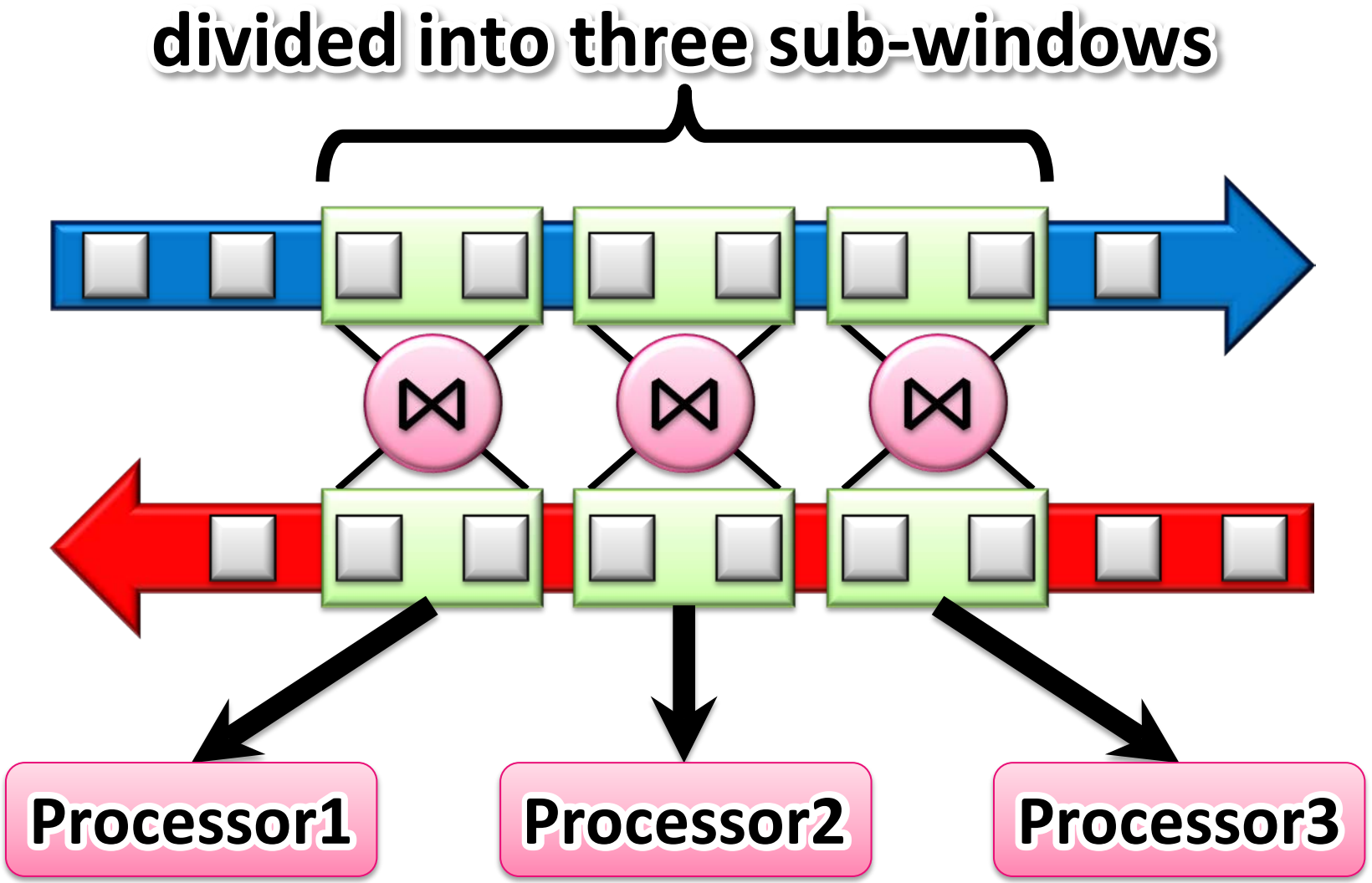
advantage Highly parallel evaluation

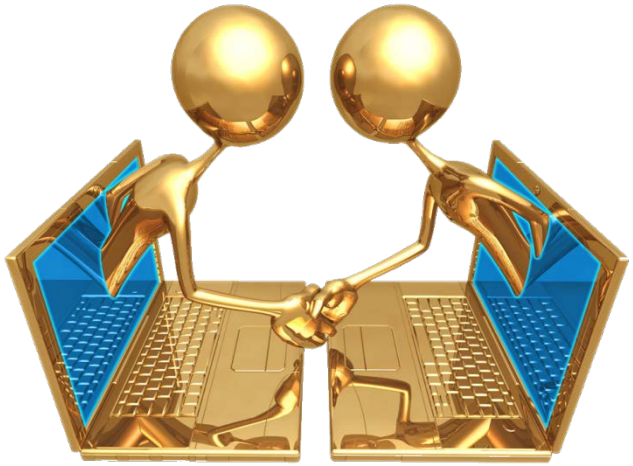


Handshake Join | Parallelization



Handshake Join | Parallelization

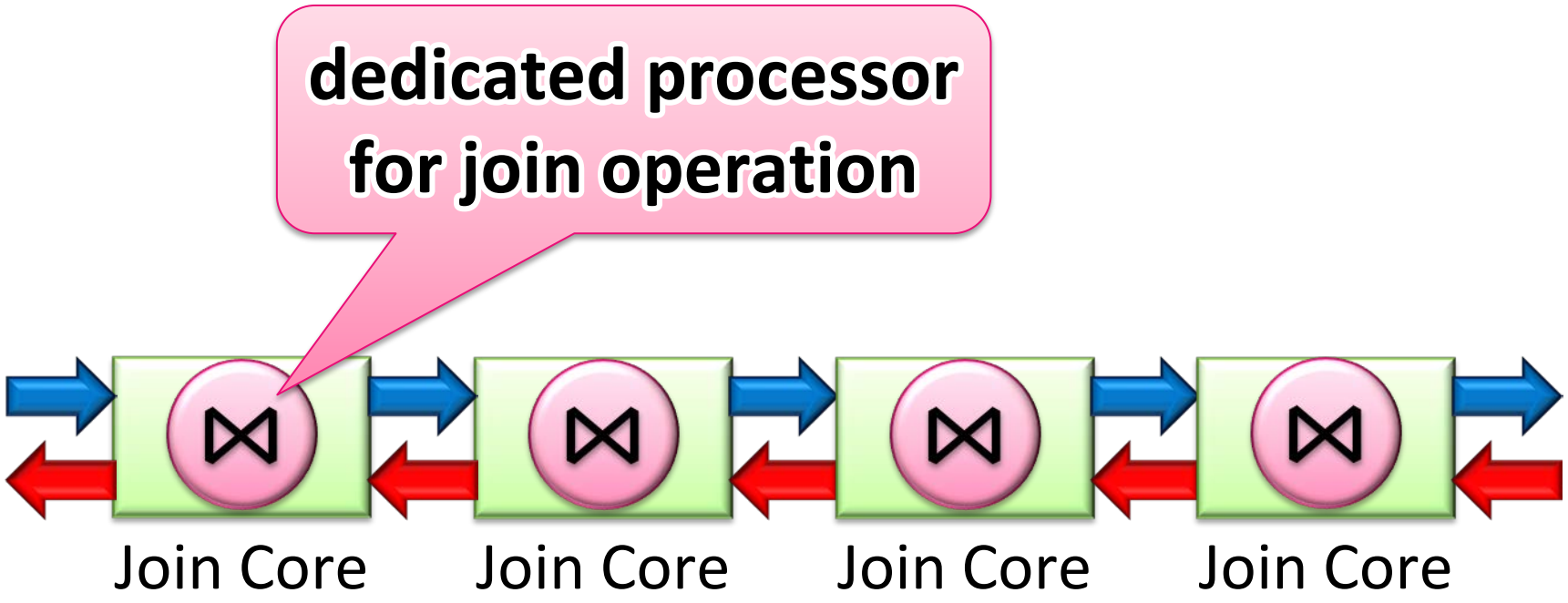




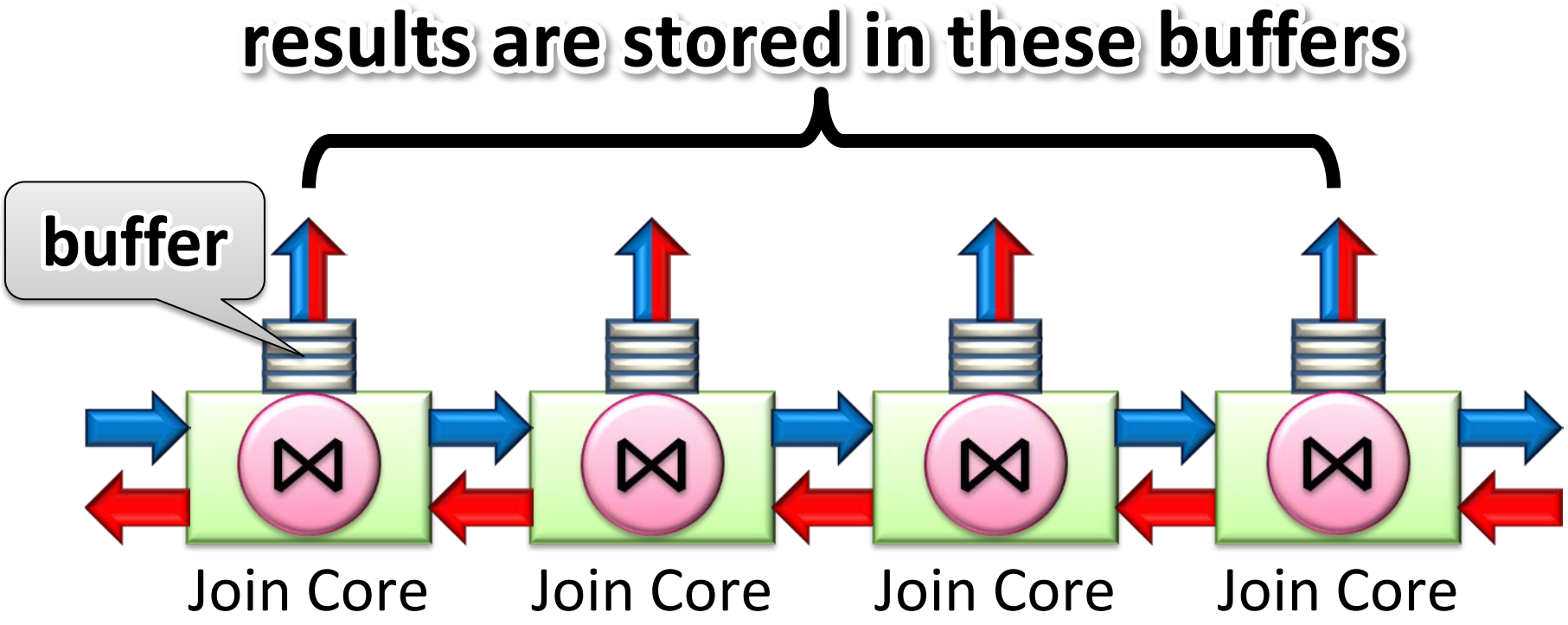
Naïve IMPLEMENTATION

HANDSHAKE JOIN、J. Teubner and R. Müller, SIGMOD'11

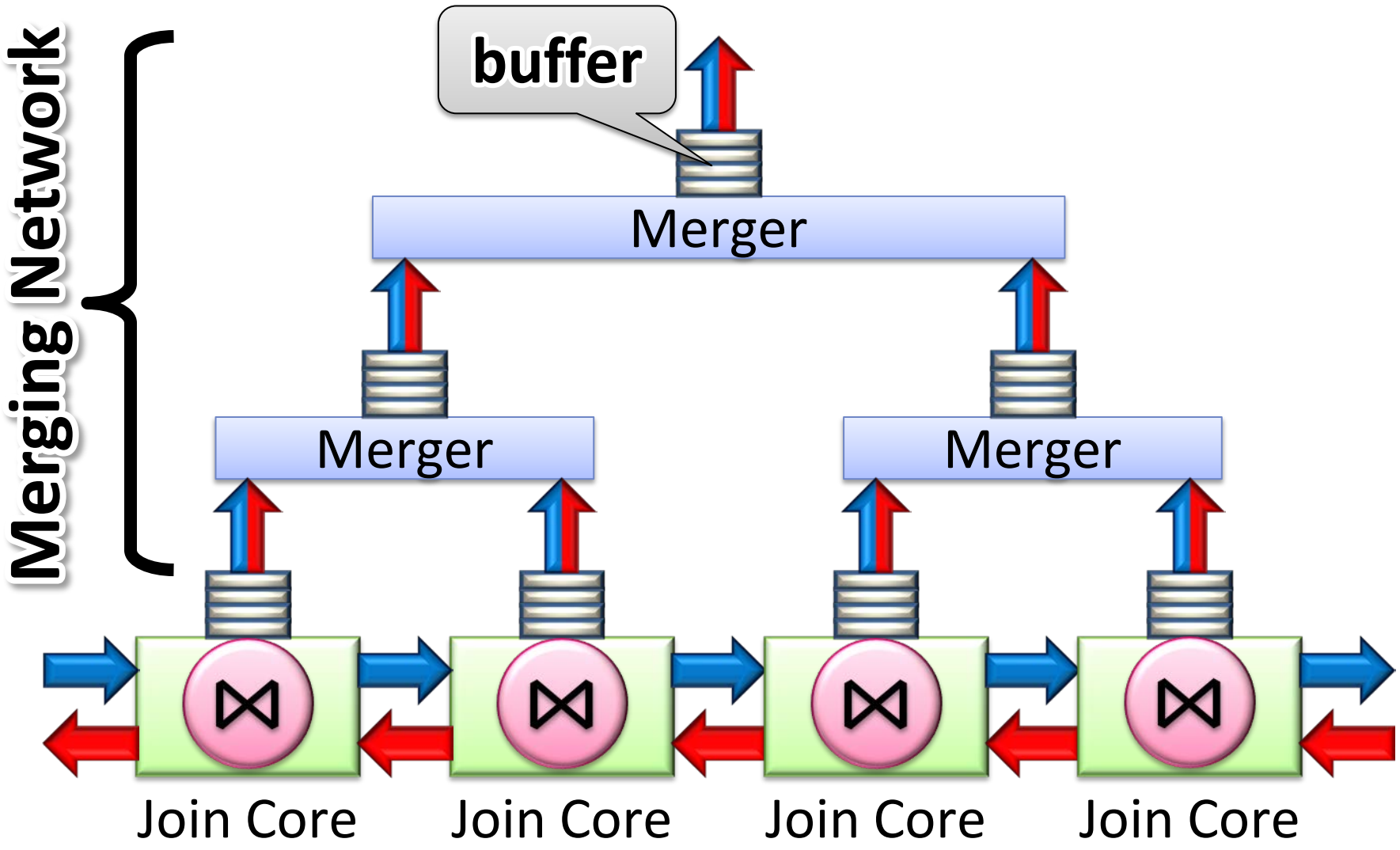
Baseline Implementation



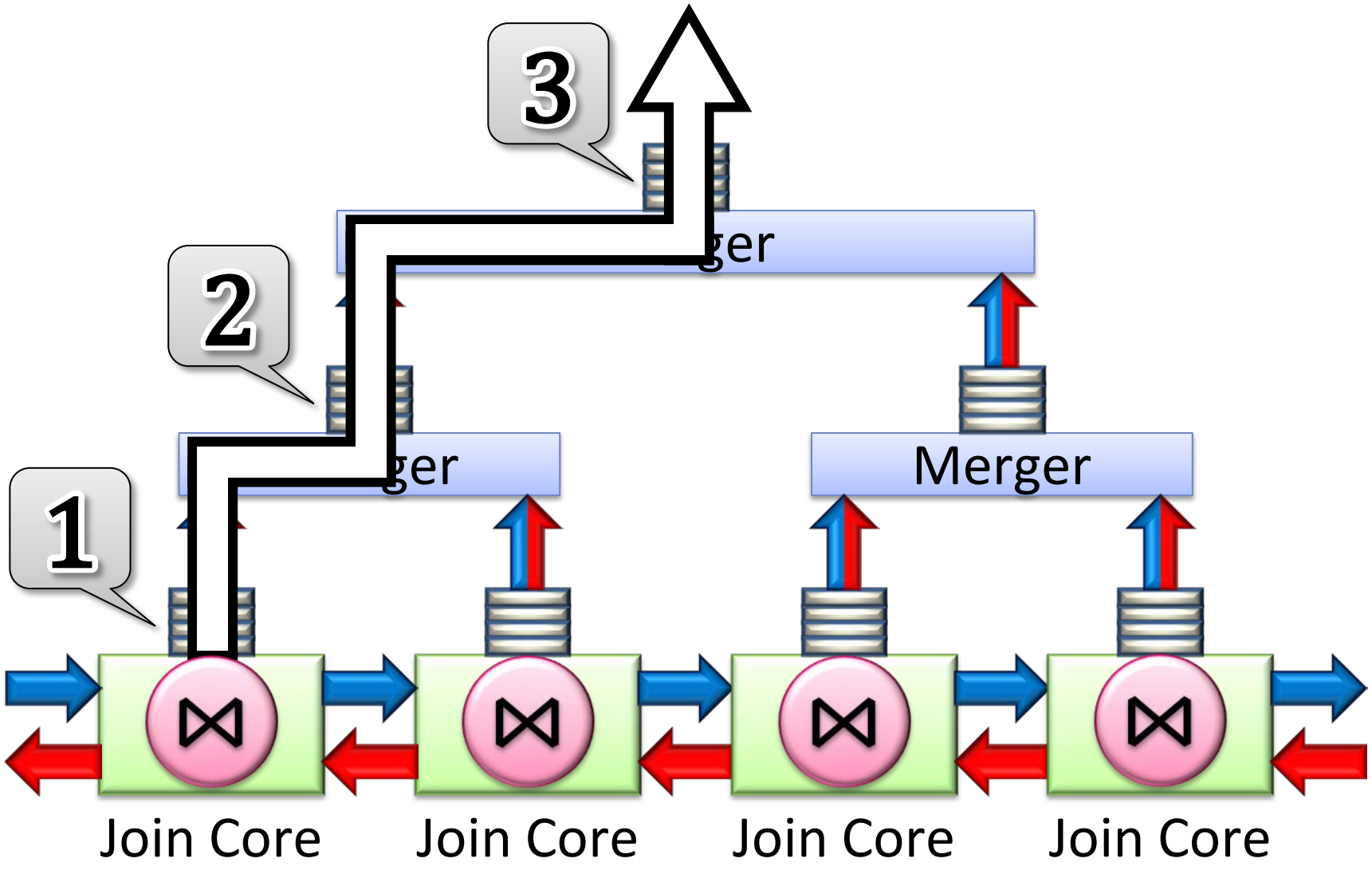
Baseline Implementation



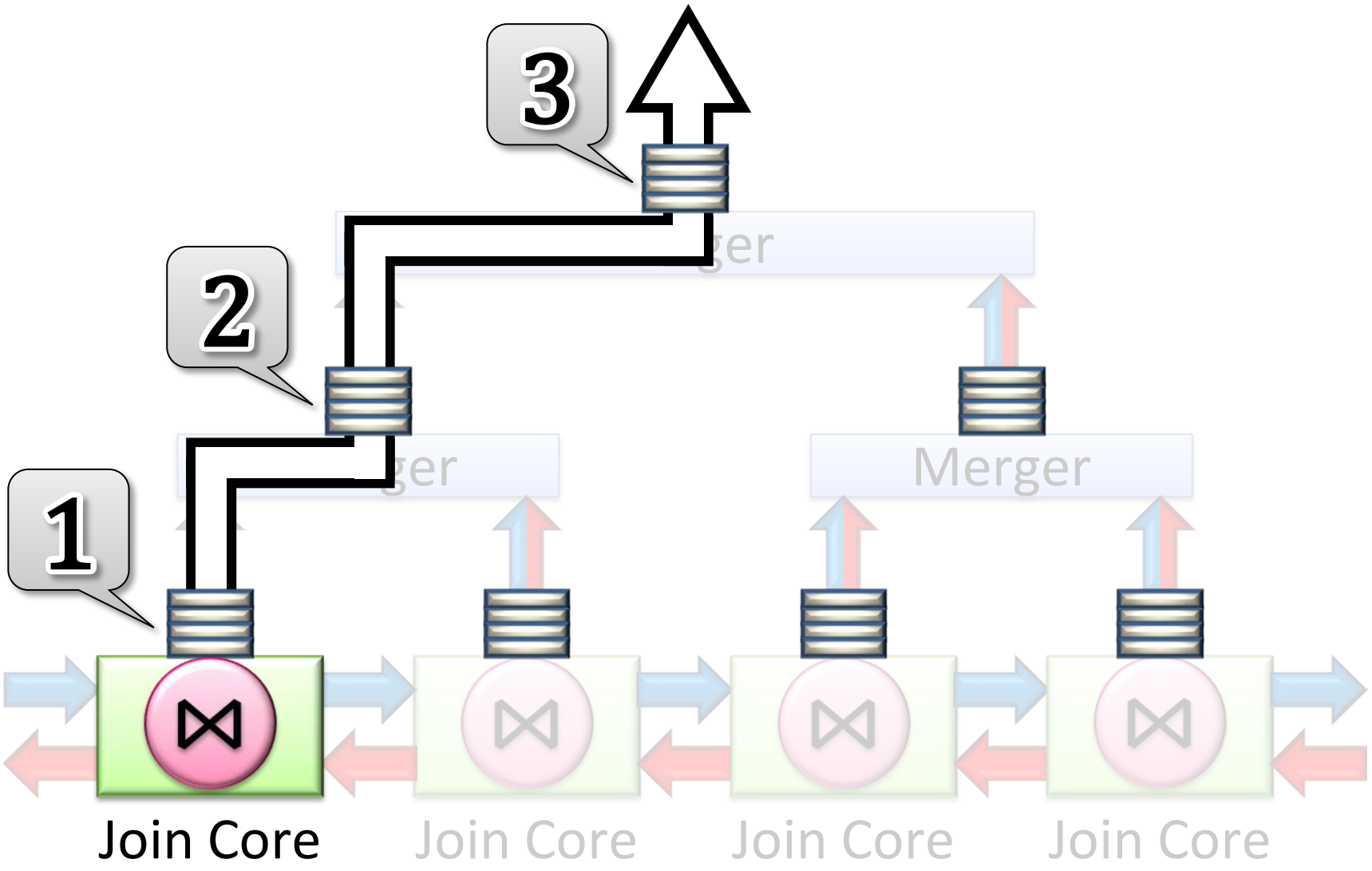
Baseline Implementation



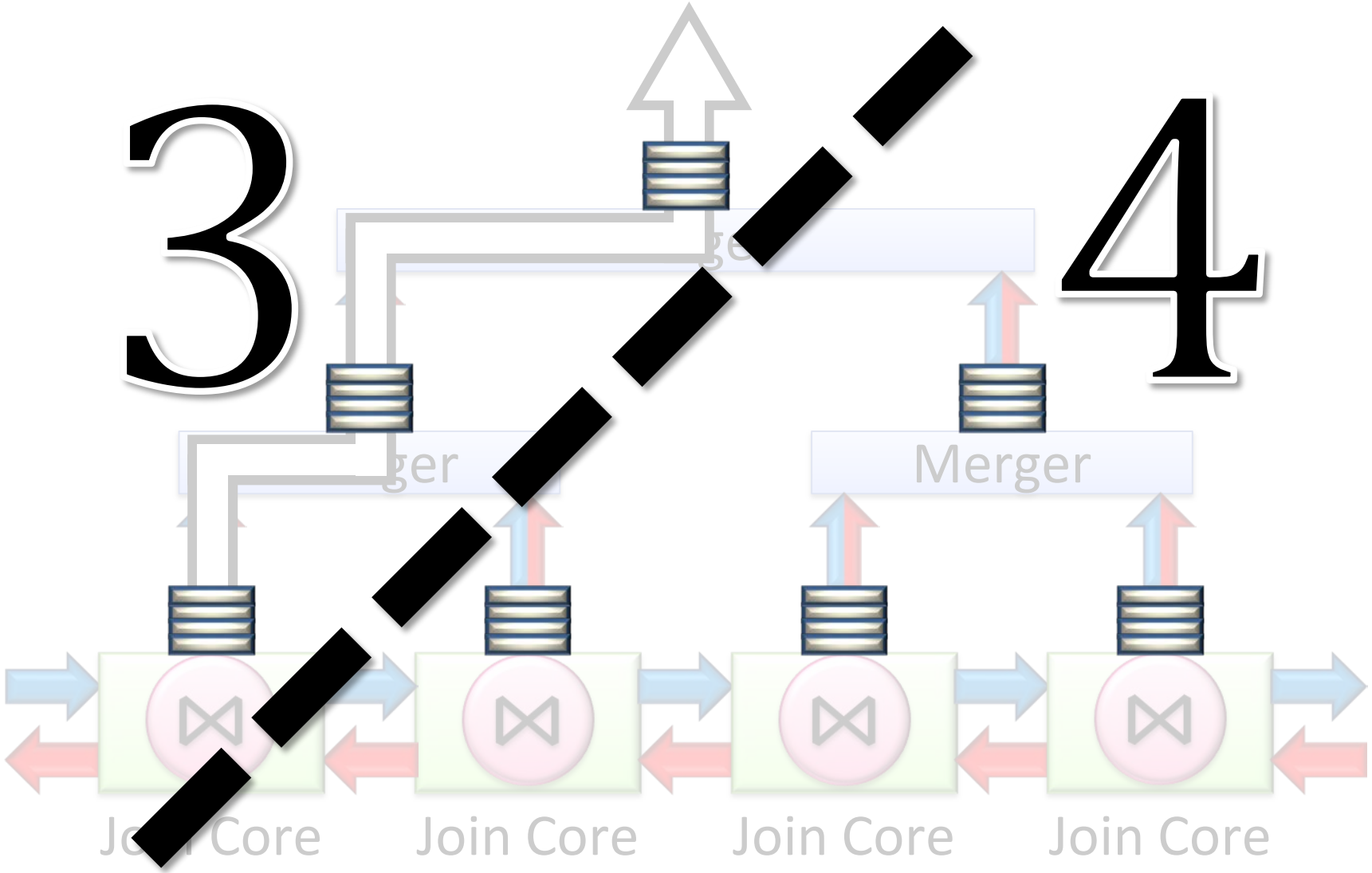
Output Data Flow



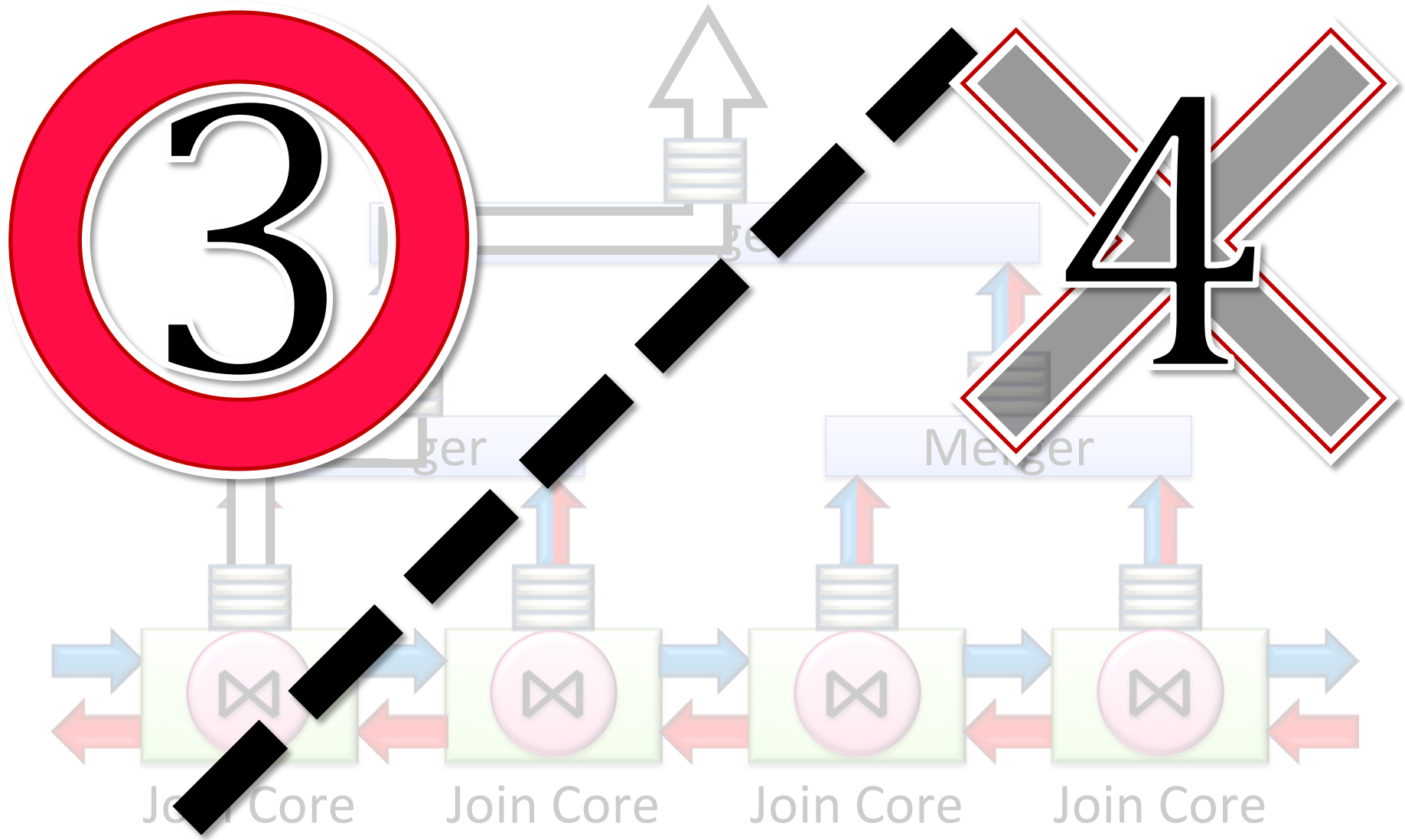
Issue | Inefficient Buffer Utilization



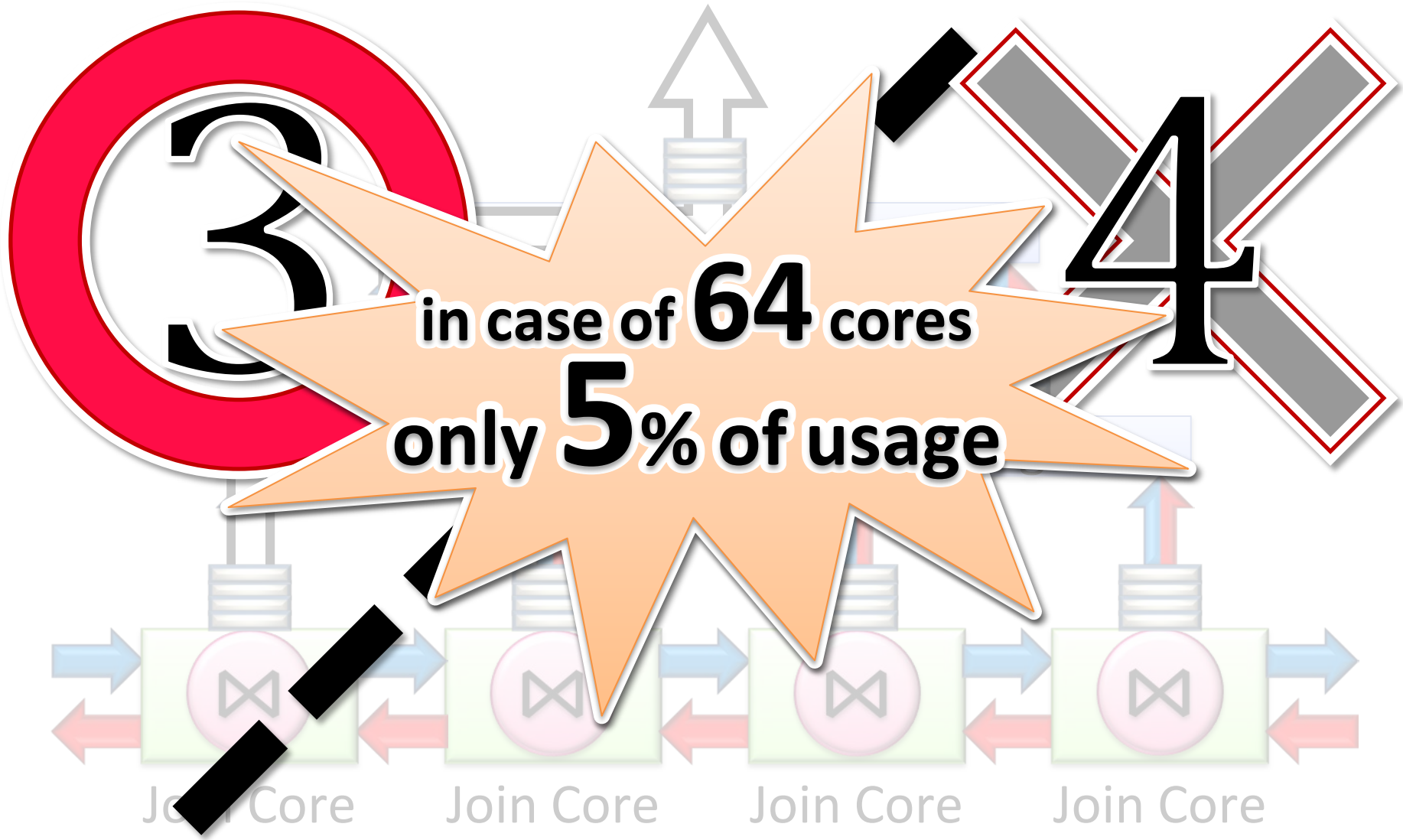
Issue | Inefficient Buffer Utilization

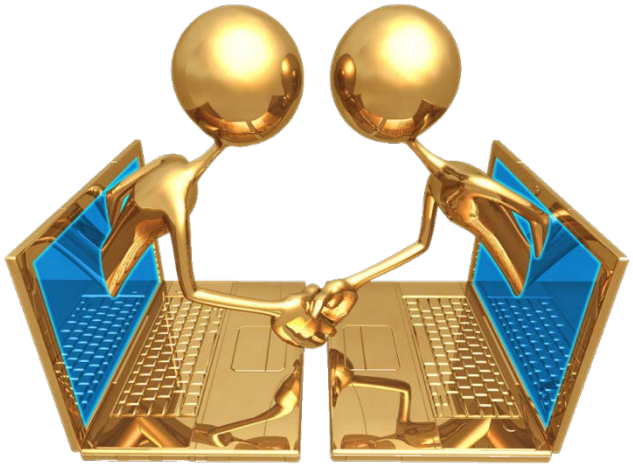


Issue | Inefficient Buffer Utilization



Issue | Inefficient Buffer Utilization

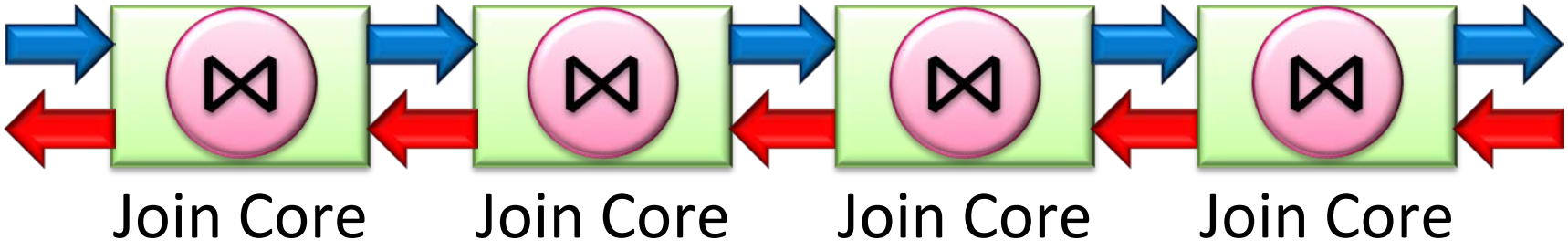




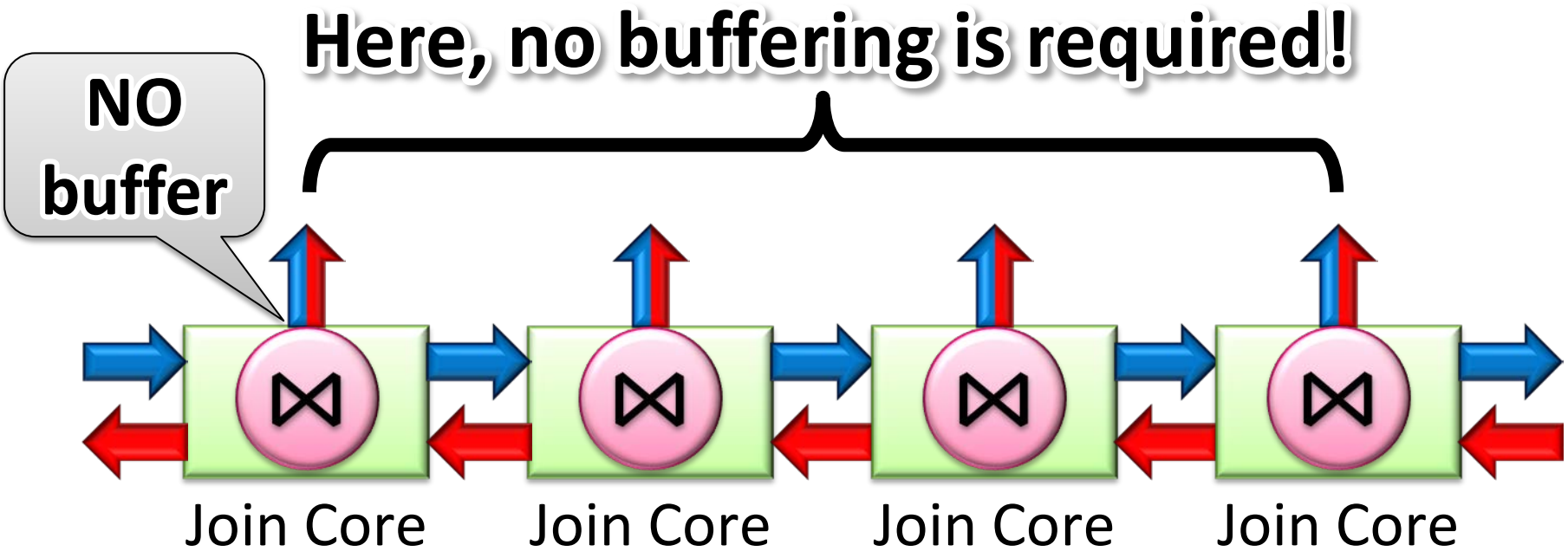
PROPOSED IMPLEMENTATION

Adaptive Merging Network

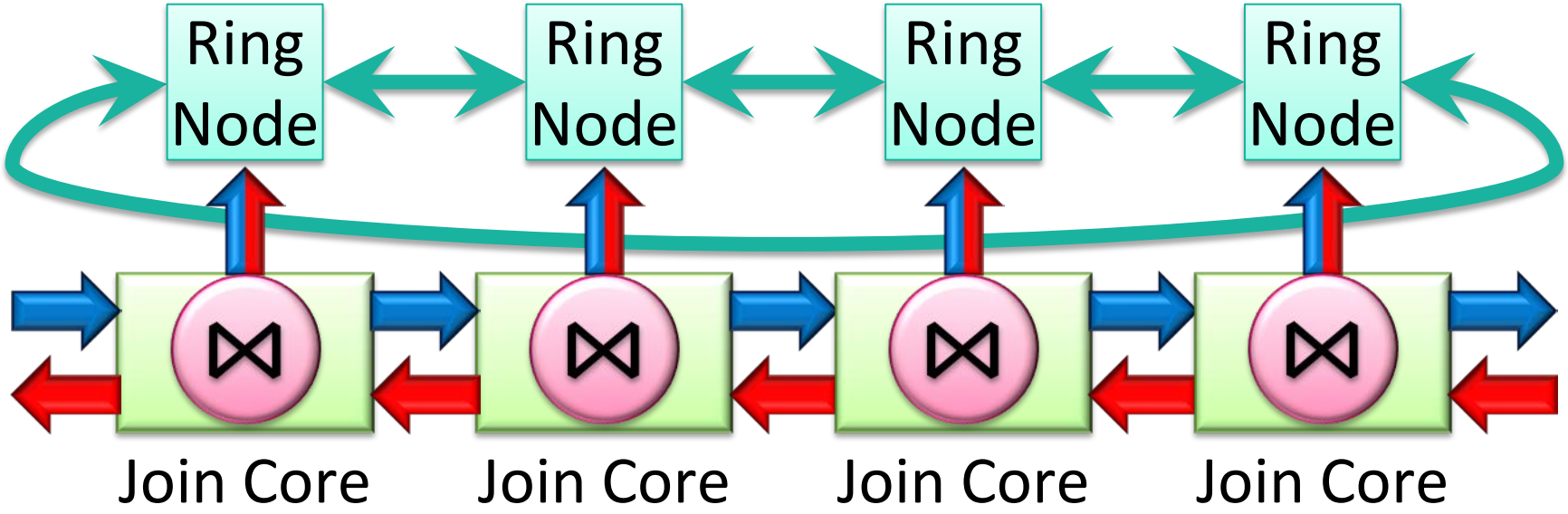
Proposed Implementation



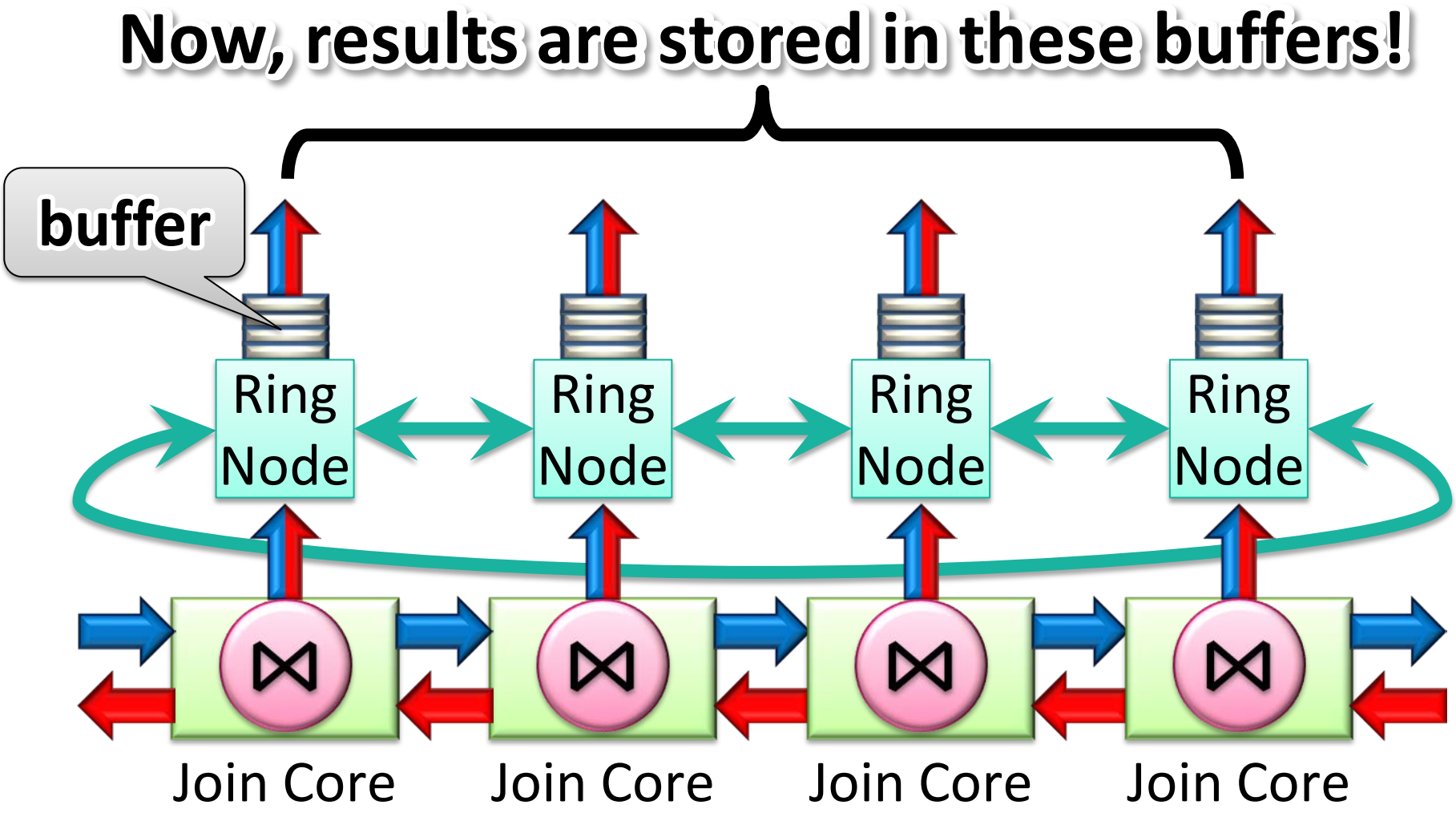
Proposed Implementation



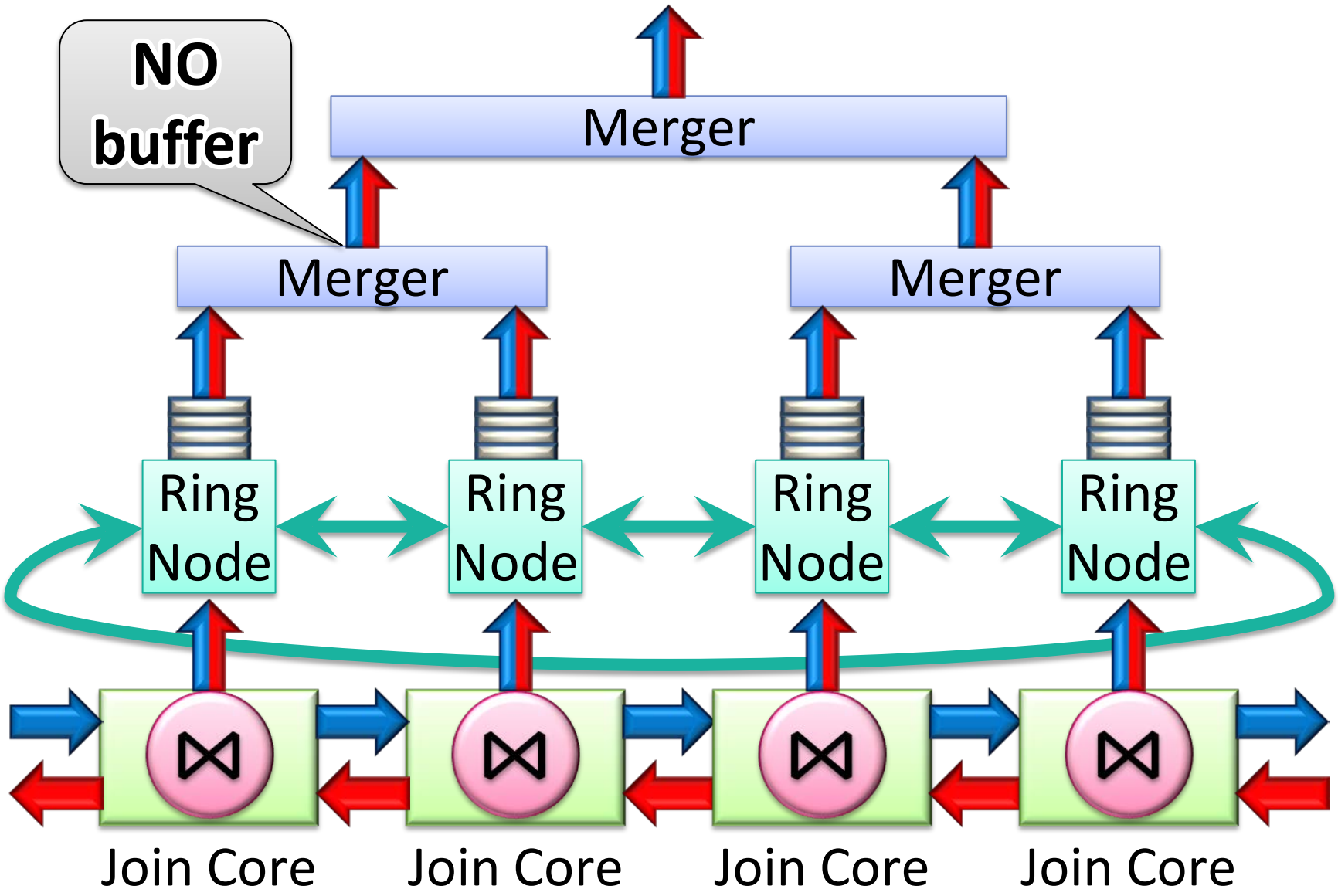
Proposed Implementation



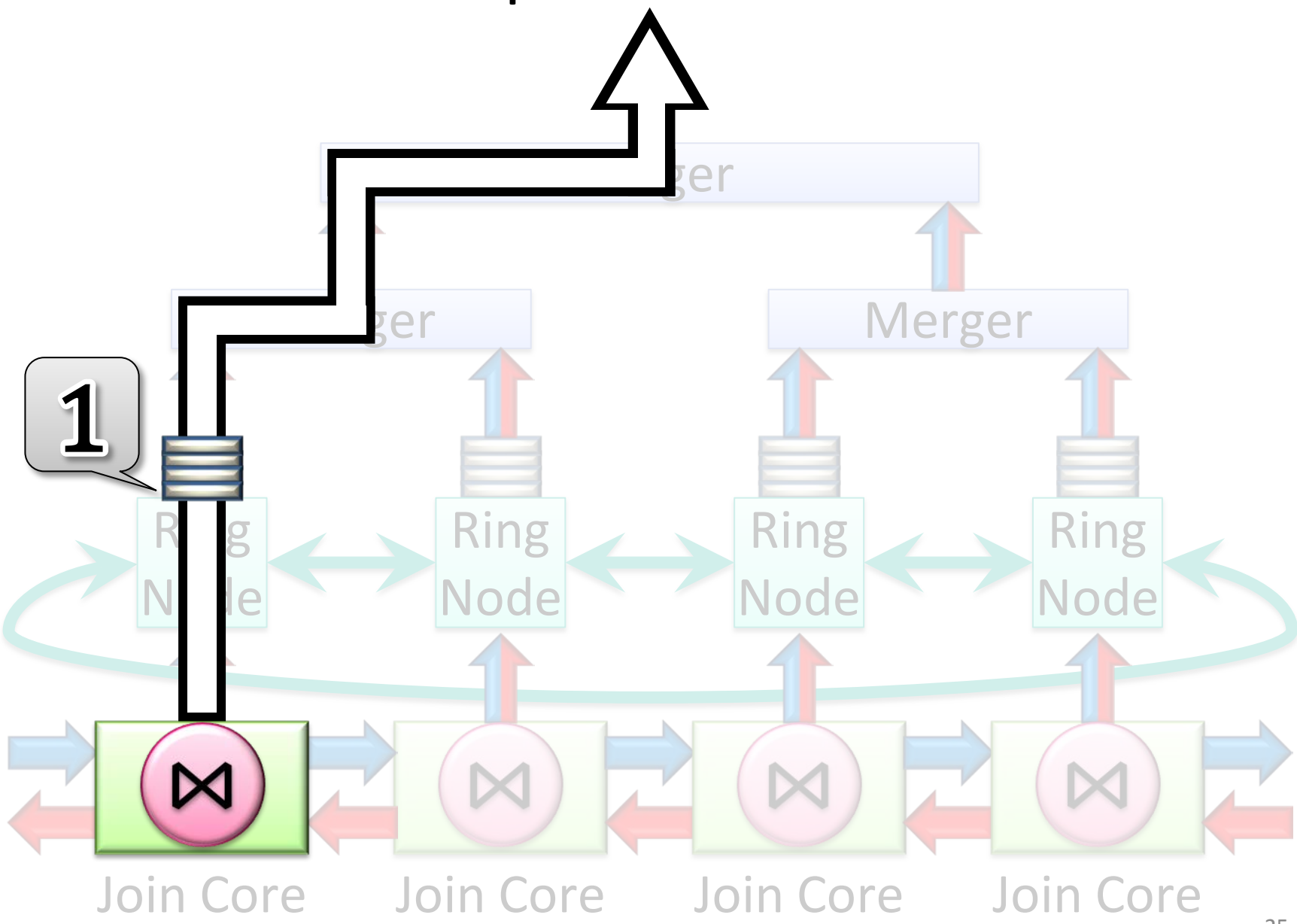
Proposed Implementation

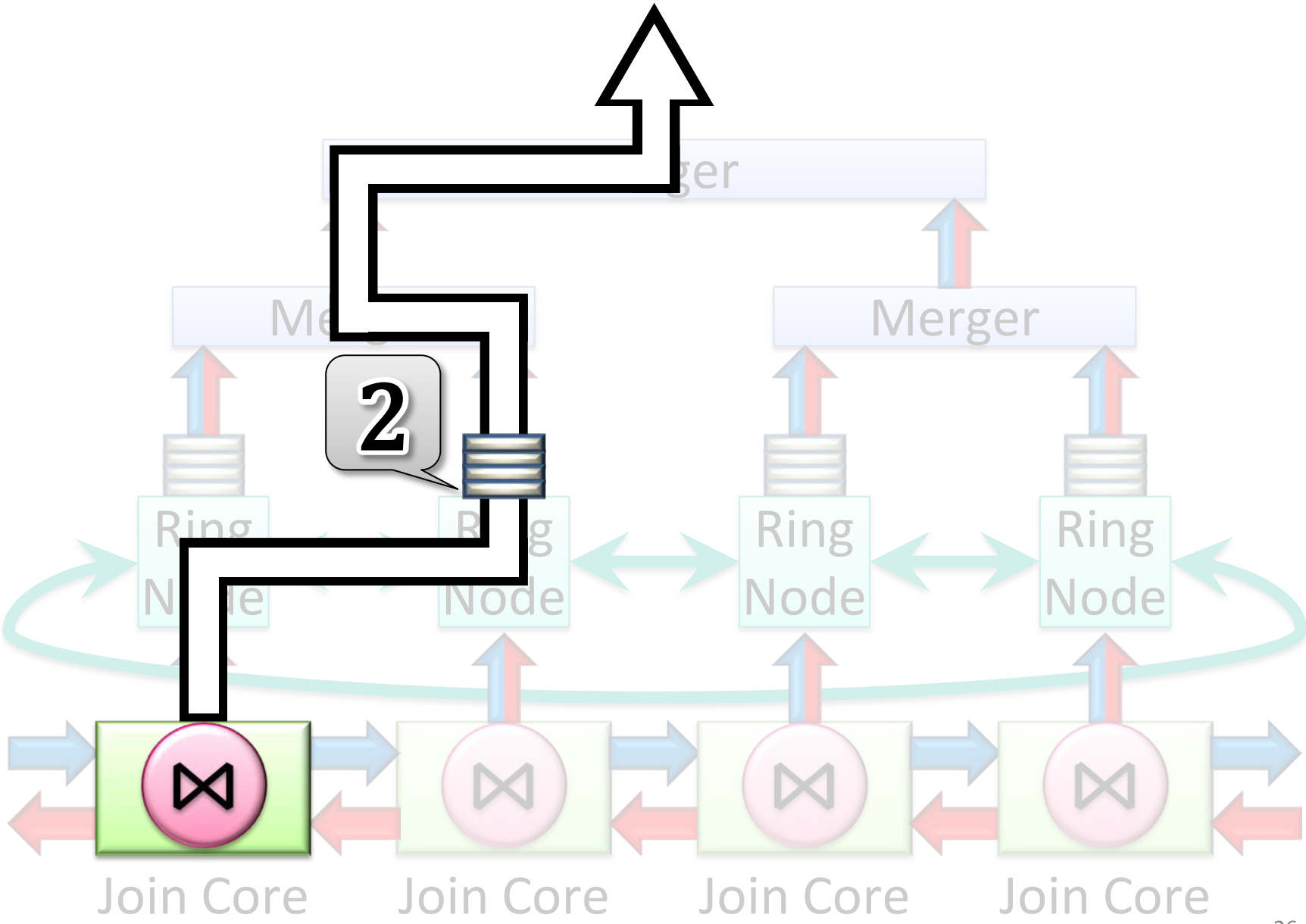


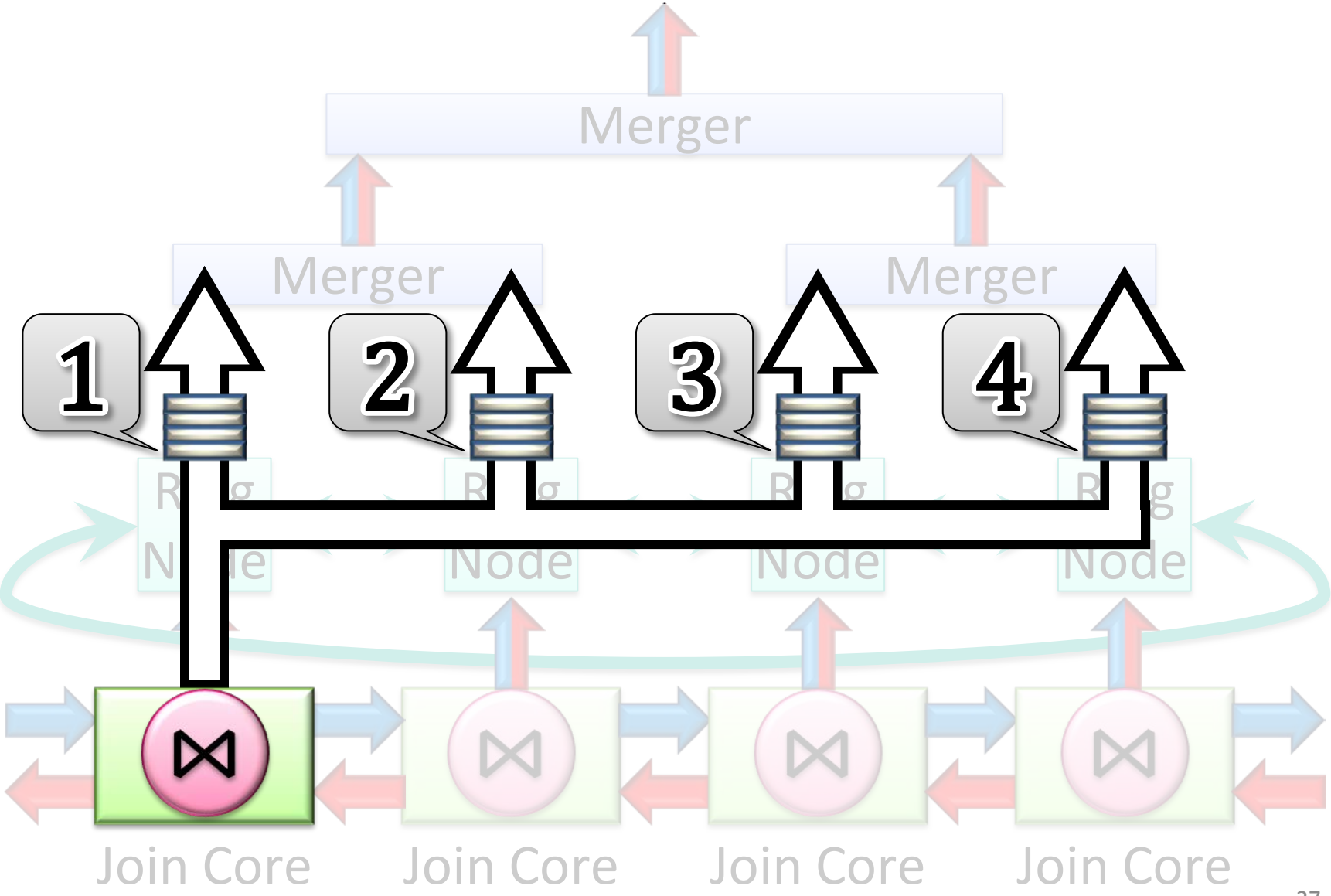
Proposed Implementation



Output Data Flow

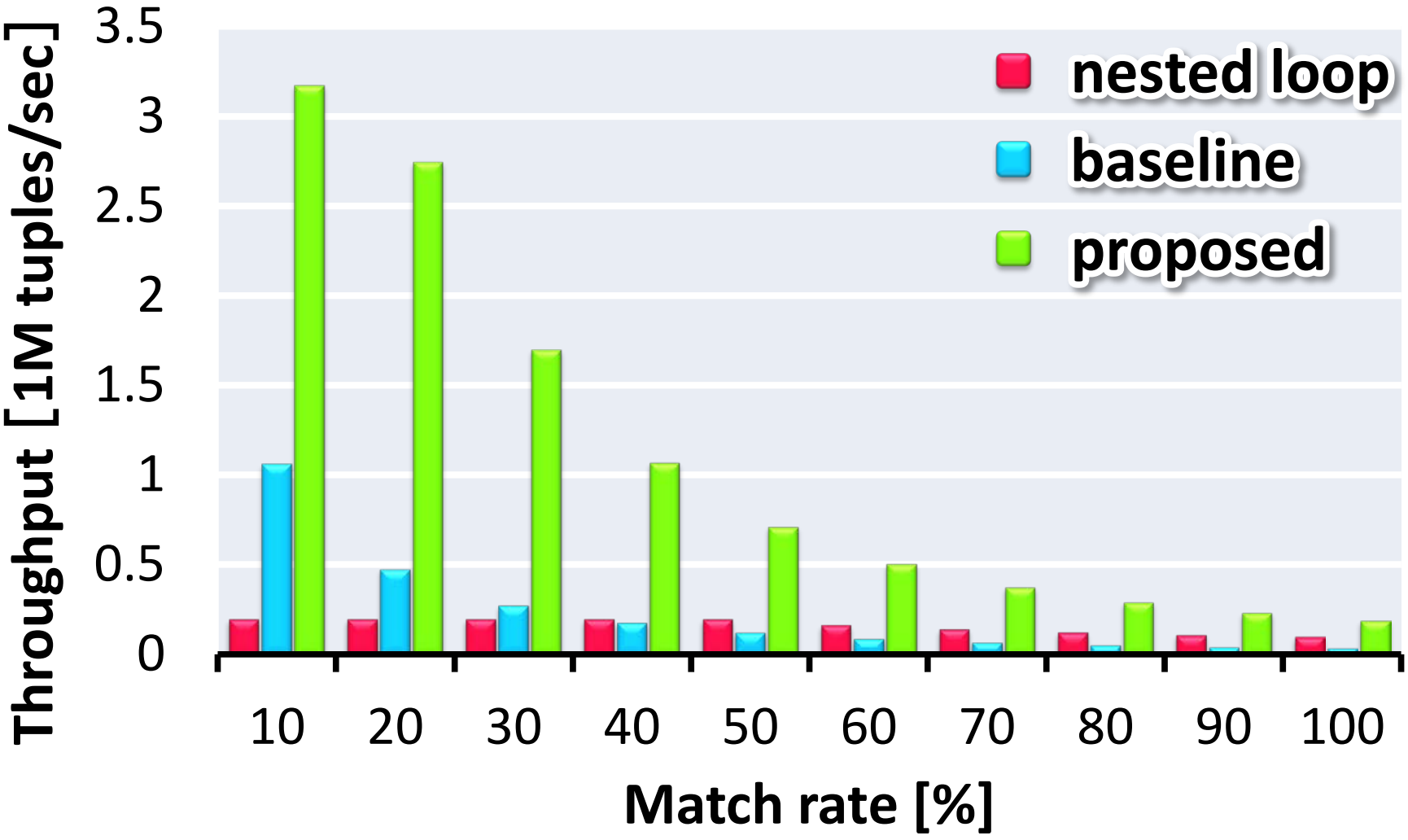



2
Merger
Merger
Ring Node
Ring Node
Ring Node
Ring Node
Join Core
Join Core
Join Core
Join Core

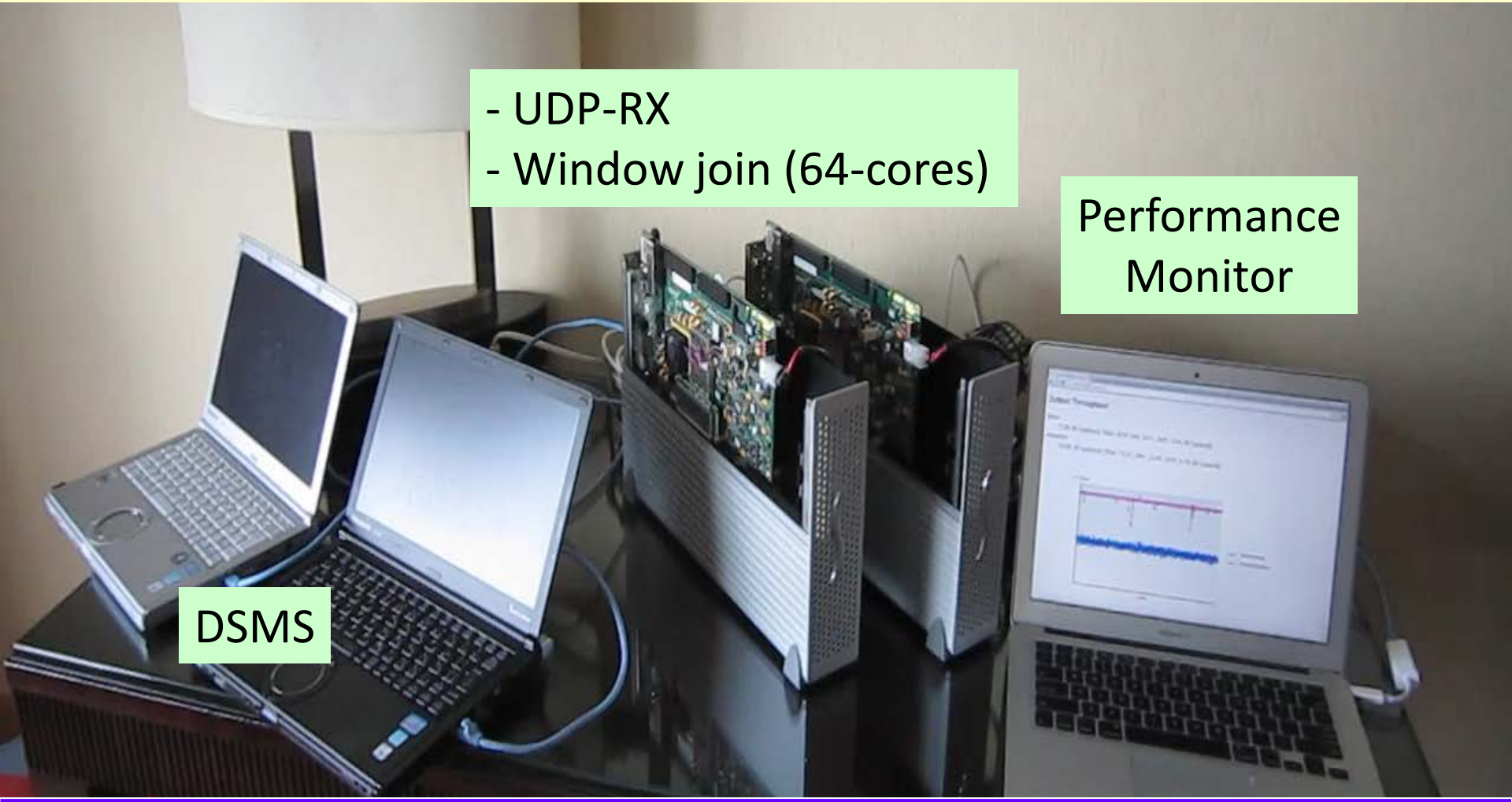

Merger
Merger
Merger
1
2
3
4
Node
Node
Node
Node
Join Core
Join Core
Join Core
Join Core


Merger
1
up to 100%
utilization
Node
Join Core
Join Core
Join Core
Join Core

Performance | Proposed (64 join cores)



Basic: 6.7 millions of tuples per second
Proposal: 14.6 millions of tuples per second



Demo @ SSDBM'13

Publications

- Yasin Oge, Takefumi Miyoshi, Hideyuki Kawashima, Tsutomu Yoshinaga: “A fast handshake join implementation on FPGA with adaptive merging network”. SSDBM 2013
- Yasin OGE, Takefumi MIYOSHI, Hideyuki KAWASHIMA, Tsutomu YOSHINAGA: "Design and Implementation of a Handshake Join Architecture on FPGA", IEICE TRANSACTIONS(Journal) Vol.E95-D No.12 pp.2919-2927
- Yasin Oge, Takefumi MIYOSHI, Hideyuki Kawashima, Tsutomu Yoshinaga, "Design and Implementation of a Merging Network Architecture for Handshake Join Operator on FPGA", IEEE MCSoC, pp.84-91, Sep. 2012.
- Yasin Oge, Takefumi Miyoshi, Hideyuki Kawashima, and Tsutomu Yoshinaga, "An Implementation of Handshake Join on FPGA", ICNC, pp.95-104, Dec. 2011.
- Takefumi Miyoshi, Hideyuki Kawashima, Yuta Terada and Tsutomu Yoshinaga, “A Coarse Grain Reconfigurable Processor Architecture for Stream Processing Engine”, FPL, Sep. 5-7, 2011

Cited by ISCA’13 Paper

Top tier conf. on FPGA