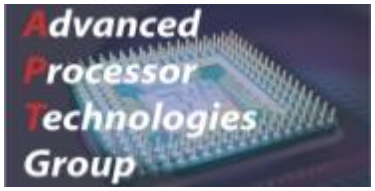
MANCHESTER 1824

The University of Manchester



EPSRC
Pioneering research and skills

An Asynchronous Routing Algorithm for Clos Networks

Wei Song and Doug Edwards

The Advanced Processor Technologies Group (APT)
School of Computer Science
The University of Manchester
{songw, doug}@cs.man.ac.uk

Outline

- Introduction of Clos networks and asynchronous circuits
- Classic Synchronous dispatching algorithms
  - Random dispatching (RD)
  - Concurrent Round-Robin Dispatching (CRRD)
- Asynchronous Dispatching (AD) algorithm
- Implementation
- Outcome
  - 32-port Clos network. Setup 6.2 ns, release 3.9 ns.

Switching Networks

- Telephone networks
- Asynchronous transfer mode (ATM) and IP networks
  - ATLANTA chip (622Mb/s/port, 1997)
  - Petastar Optical Switch (160Gb/s/port, 2003)
- ***Intra/inter chip interconnect***
  - High-radix router with many narrow ports
  - SDM: delay guaranteed services

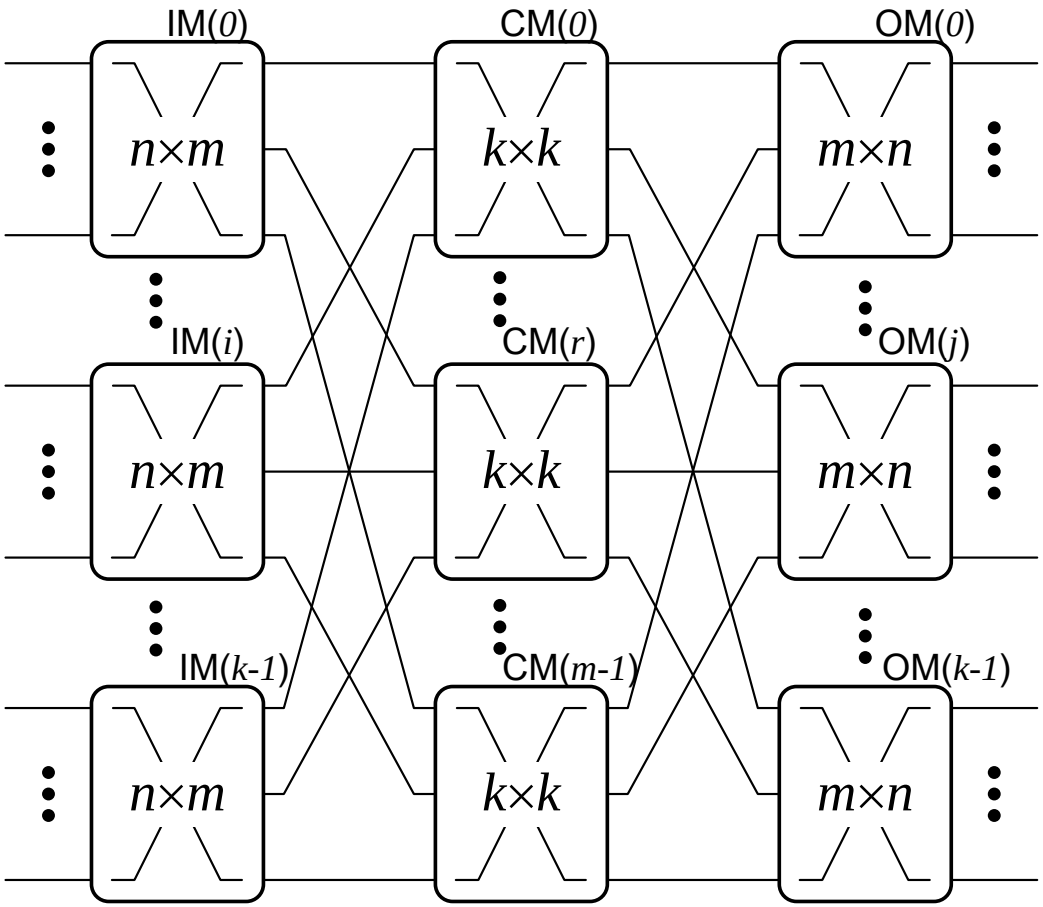
Asynchronous Circuit

- Asynchronous vs. synchronous
  - ***Low power (clockless)***
  - Possible performance boost (average latency)
  - ***Tolerance to process variation***
- Implementation style
  - 2-phase vs. 4-phase
  - Bundled-data vs. quasi-delay insensitive (QDI)
  - ***4-phase QDI***

Research Target

- An area-efficient and high-speed switching network for on-chip networks
  - High-radix router
  - Low-power consumption
  - Tolerance to process variation
  - Area efficient

Clos Switching Network



3 stages:
IMs, CMs and OMs

C(n, k, m):
k IM/OMs
n ports per IM/OM
m CMs

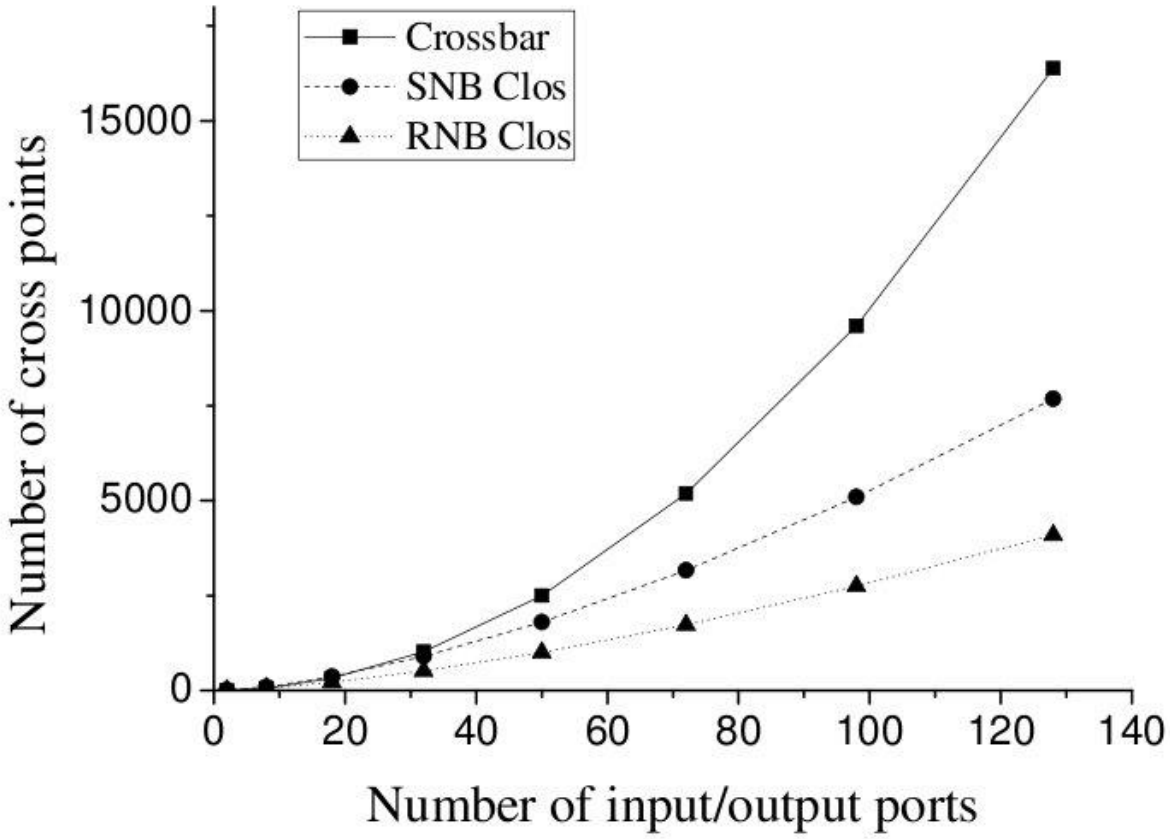
LI links between IM/CM
LO links between CM/OM

SMS: buffer CM
MSM: buffer IM/OM
S³: no buffer

Non-Blocking

- Blocking
  - An available input/output pairs may not be connected due to internal blocking.
- Strict Non-Blocking (SNB)
  - All available input/output pairs are connectable.
  - $m \geq 2n-1$
- Rearrangeable Non-Blocking (RNB)
  - Connecting available input/output pairs may require internal permutation.
  - $n \leq m < 2n-1$

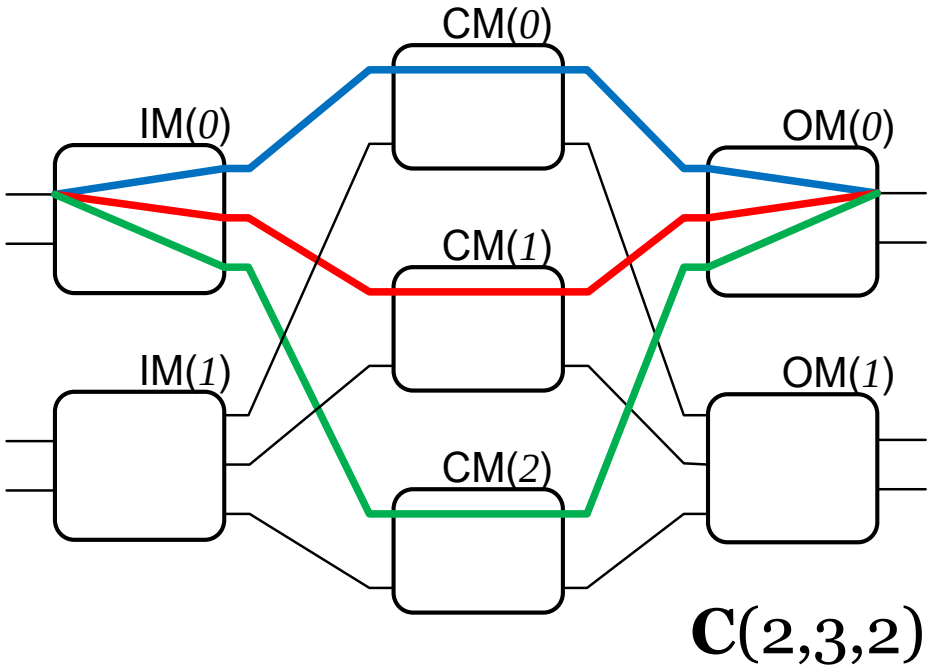
Area Consumption



Routing Algorithms

- Optimal algorithms
  - Algorithms provide guaranteed results for all matches but with a higher complexity in time and implementation.
- Heuristic algorithms
  - Algorithms provide all or partial connections in much lower time complexity.
- Most dynamically reconfigurable Clos networks use heuristic algorithms

Dispatching Algorithms

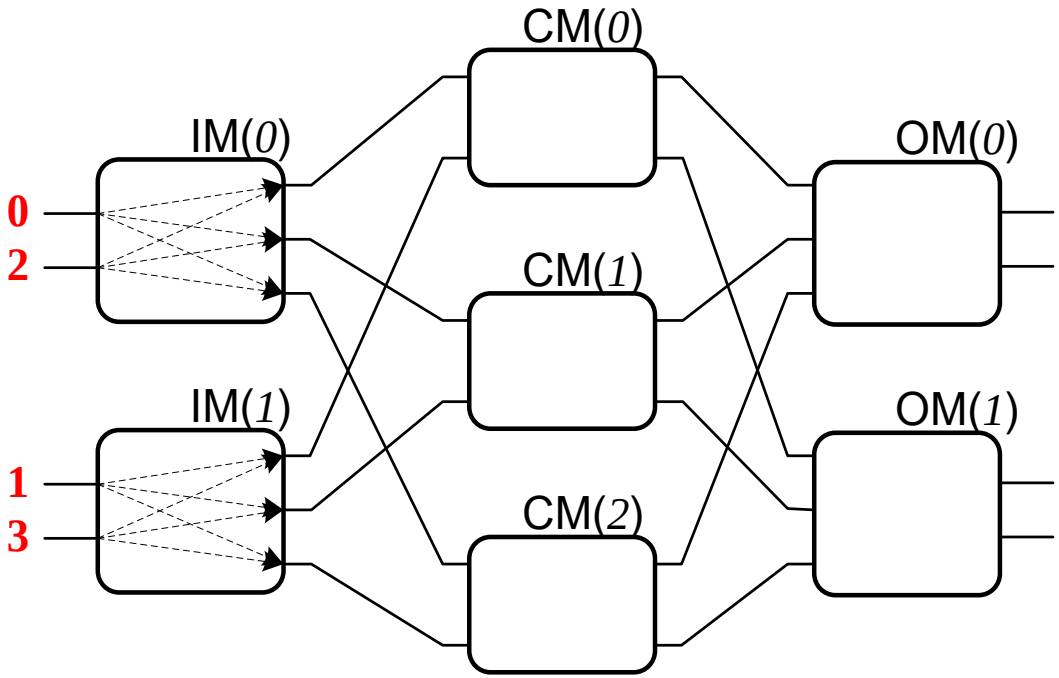


Every Input/output pair has *m* paths. Packets are dispatched evenly to all CMs.

Dispatching algorithm: the algorithm used in IM to CM packet dispatching.

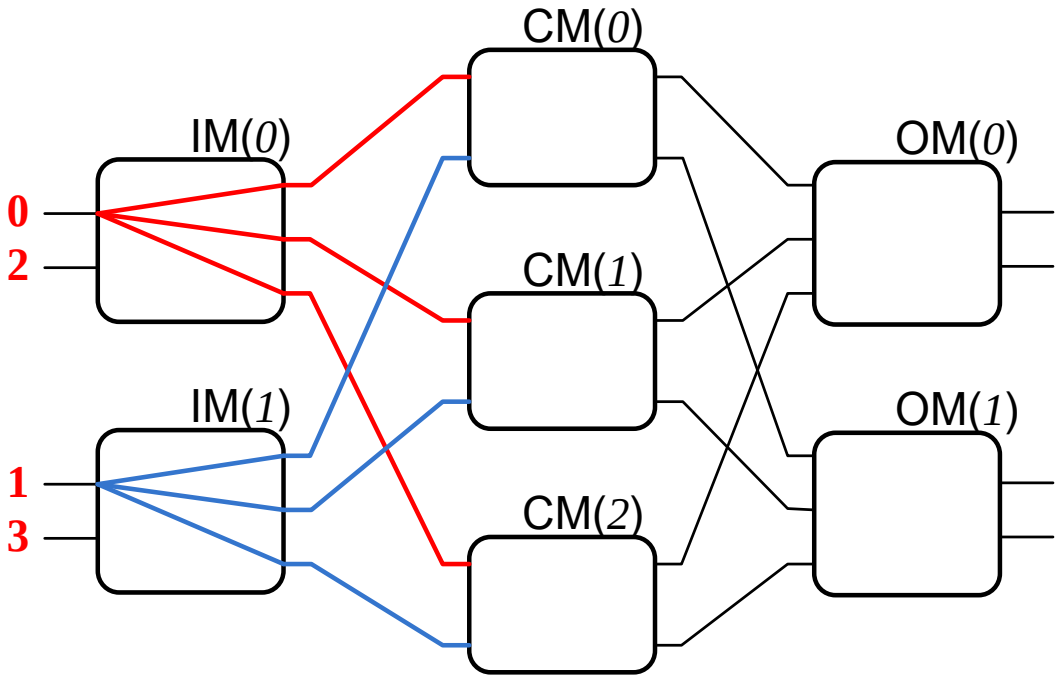
Random Dispatching (RD)

- Phase 1
  - **IPs send requests to LIs**
  - LIs select IPs
- Phase 2
  - CMs select LIs
  - Configure granted IPs



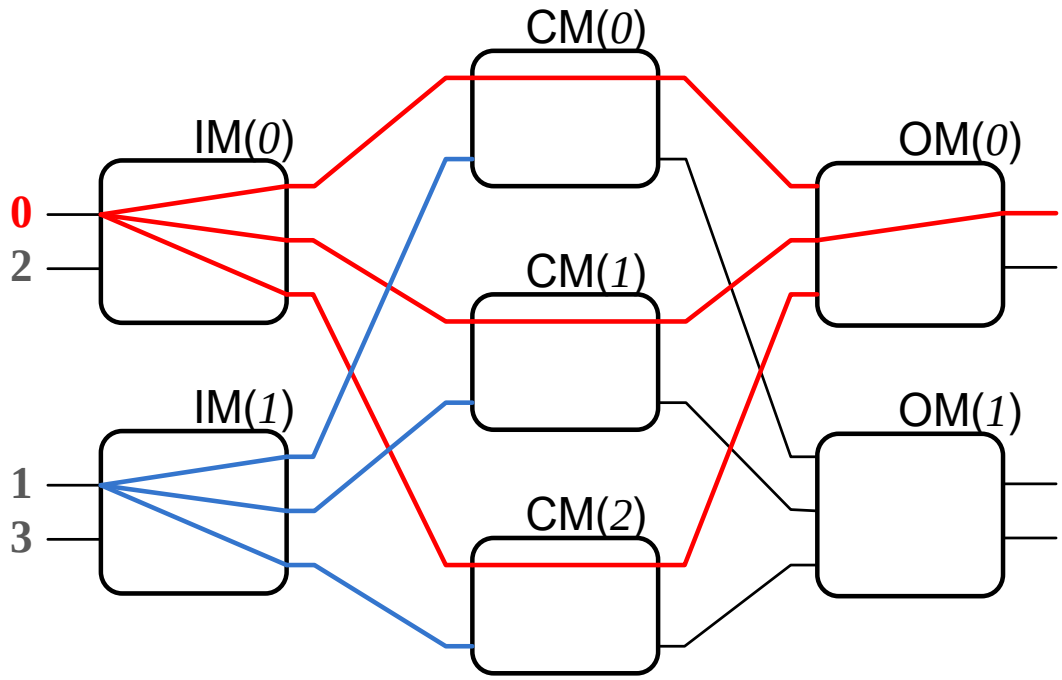
Random Dispatching (RD)

- Phase 1
  - IPs send requests to LIs
  - **LIs select IPs**
- Phase 2
  - CMs select LIs
  - Configure granted IPs



Random Dispatching (RD)

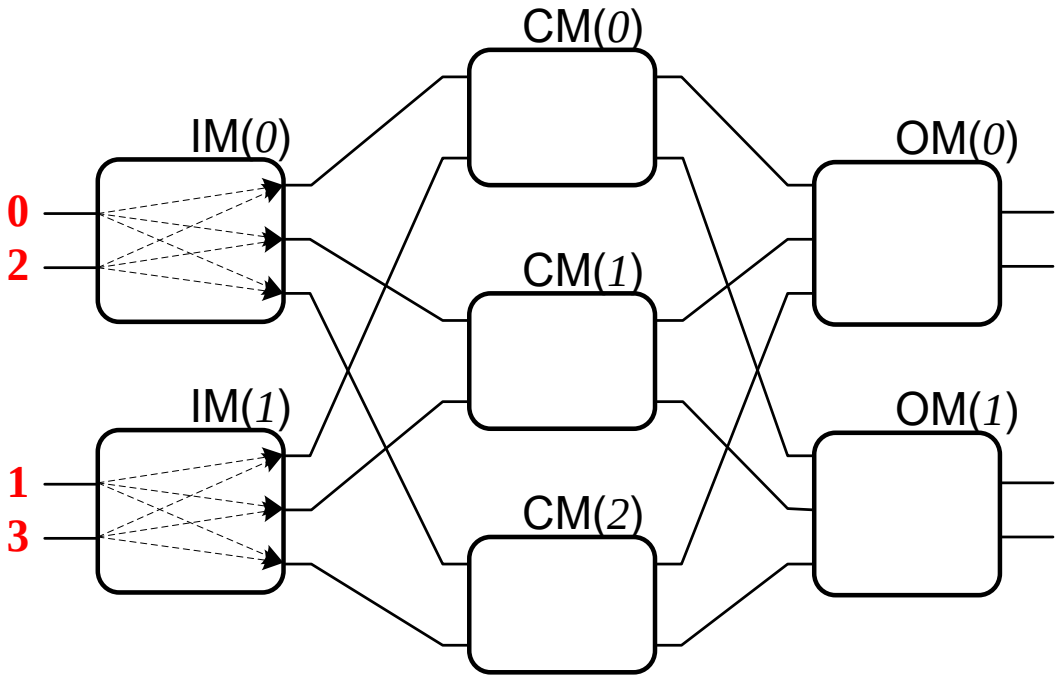
- Phase 1
  - IPs send requests to LIs
  - LIs select IPs
- Phase 2
  - **CMs select LIs**
  - **Configure granted IPs**



RD cannot resolve contention in IMs.

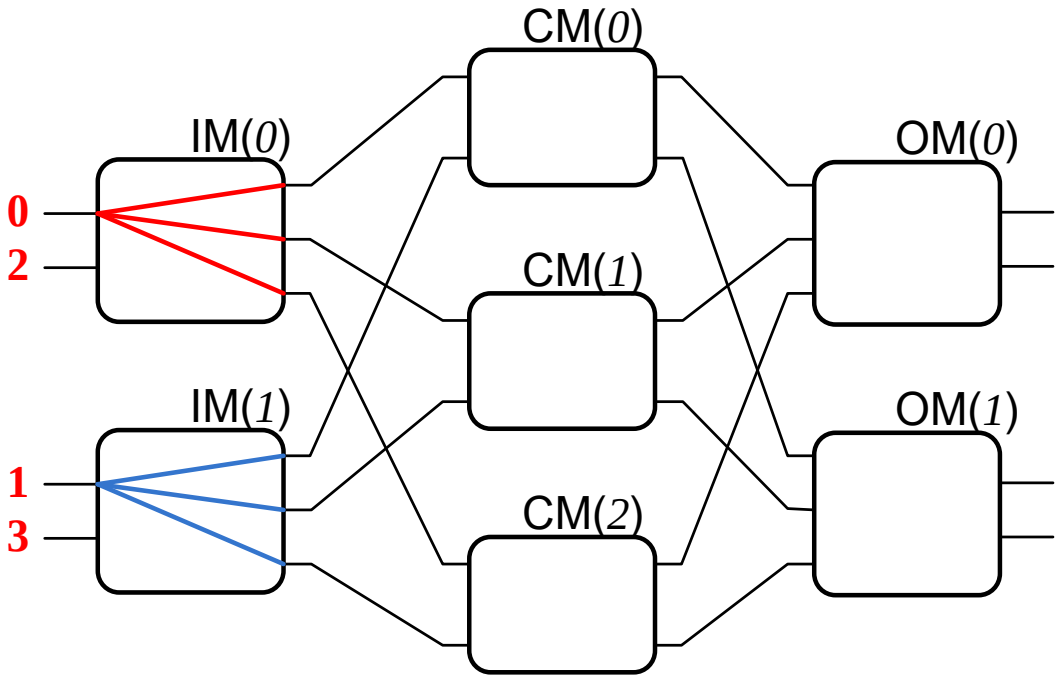
Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - **IPs request**
  - LIs select
  - IPs select
  - Go back (iteration)
- Phase 2
  - Same as RD



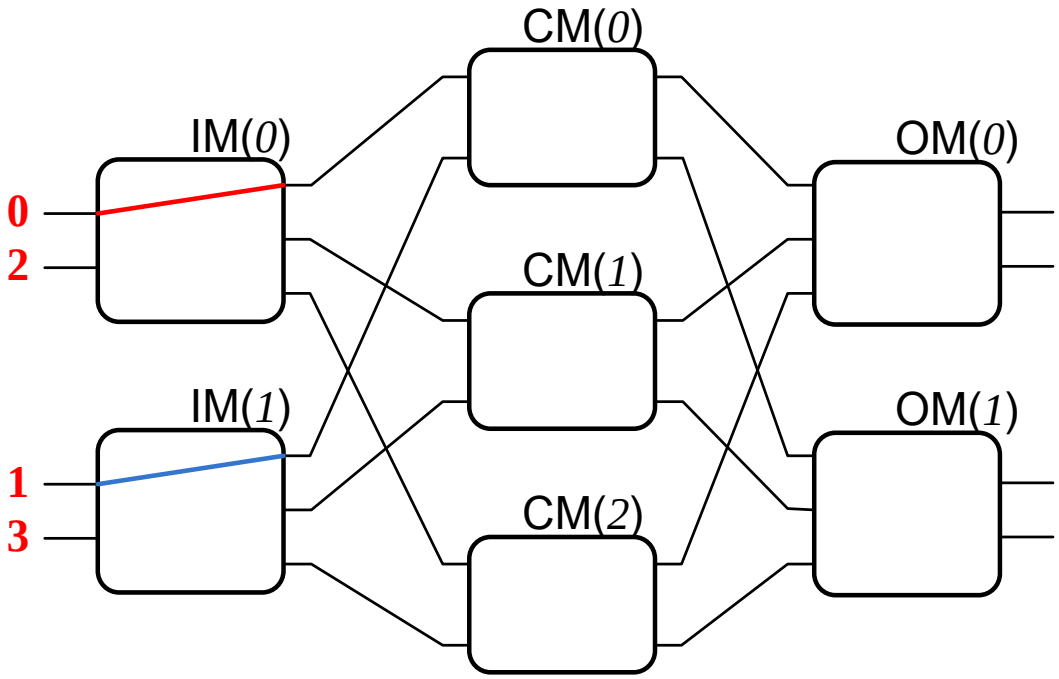
Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - IPs request
  - **LIs select**
  - IPs select
  - Go back (iteration)
- Phase 2
  - Same as RD



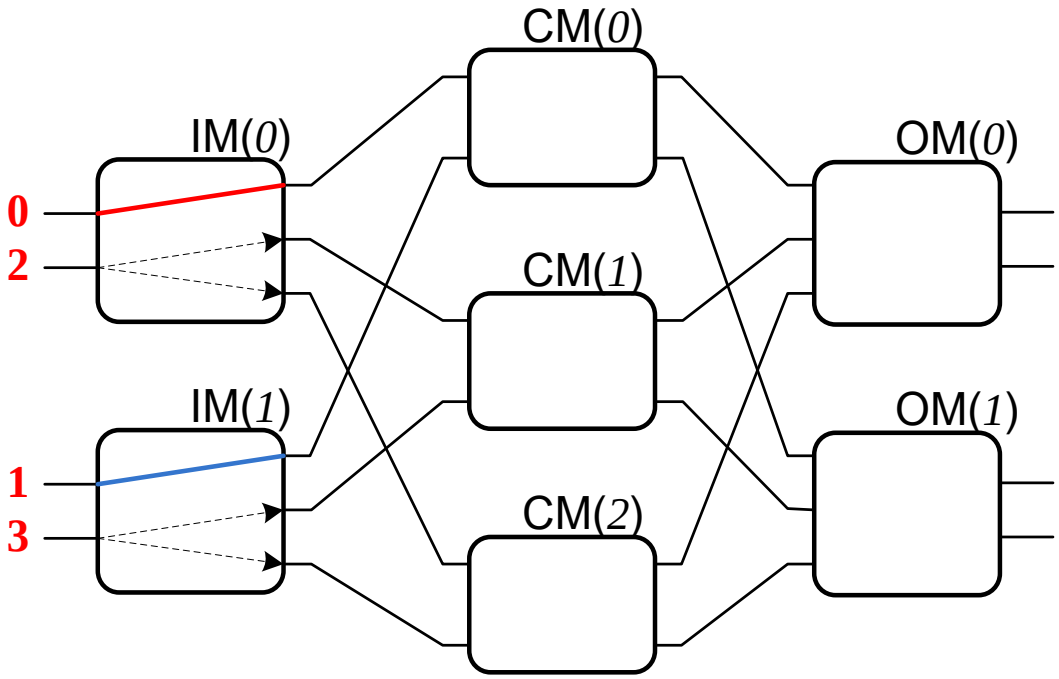
Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - IPs request
  - LIs select
  - **IPs select**
  - Go back (iteration)
- Phase 2
  - Same as RD



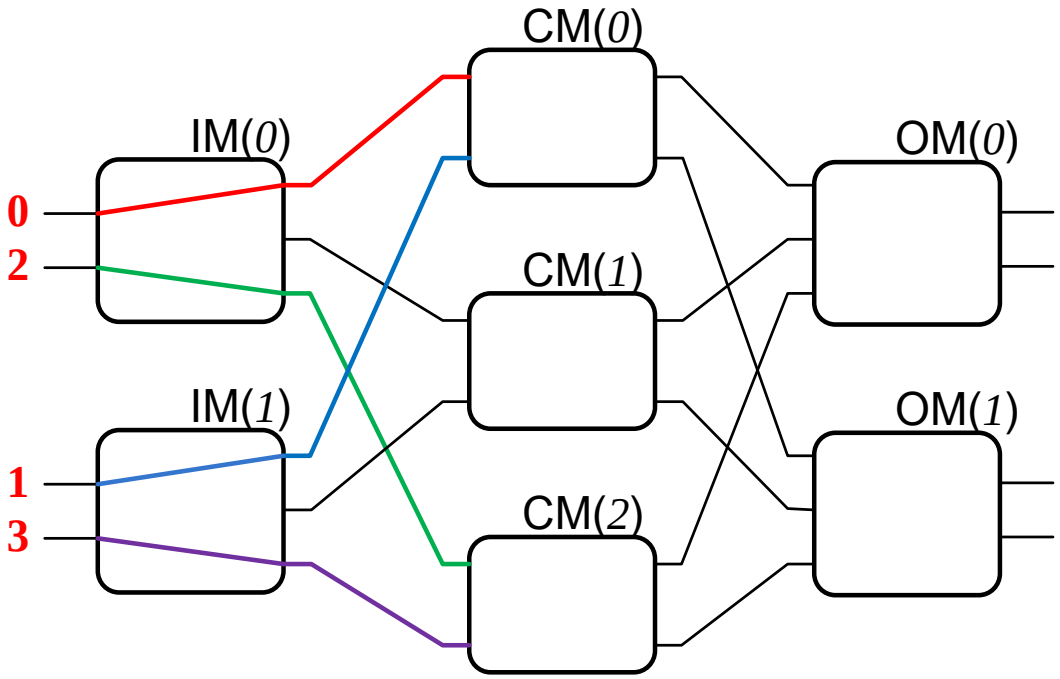
Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - IPs request
  - LIs select
  - IPs select
  - **Go back (iteration)**
- Phase 2
  - Same as RD



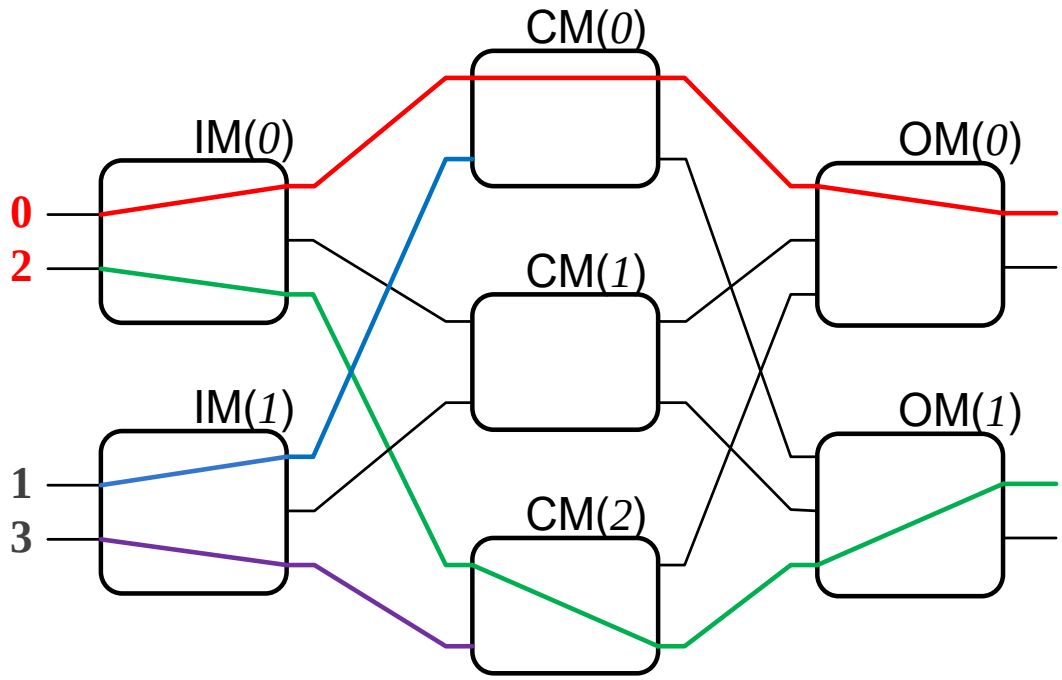
Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - IPs request
  - LIs select
  - IPs select
  - **Go back (iteration)**
- Phase 2
  - Same as RD



Concurrent Round-Robin Dispatching (CRRD)

- Phase 1
  - IPs request
  - LIs select
  - IPs select
  - Go back (iteration)
- Phase 2
  - **Same as RD**



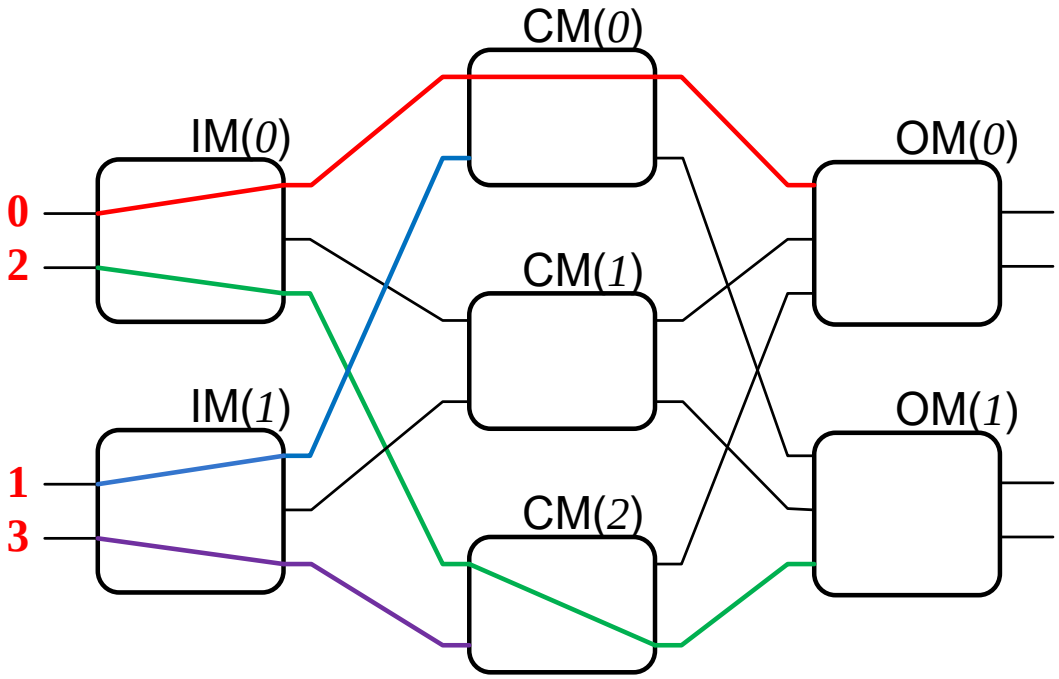
CRRD cannot resolve contention in CMs.

Asynchronous Dispatching (AD)

- Difficulties
  - Packets arrive asynchronously
  - Modules are event-driven
  - Orphans are generated by failed requests
- Solution
  - Independent algorithms (allocators) run in IMs and CMs
  - Failed requests use status feedback from CMs as acknowledge.

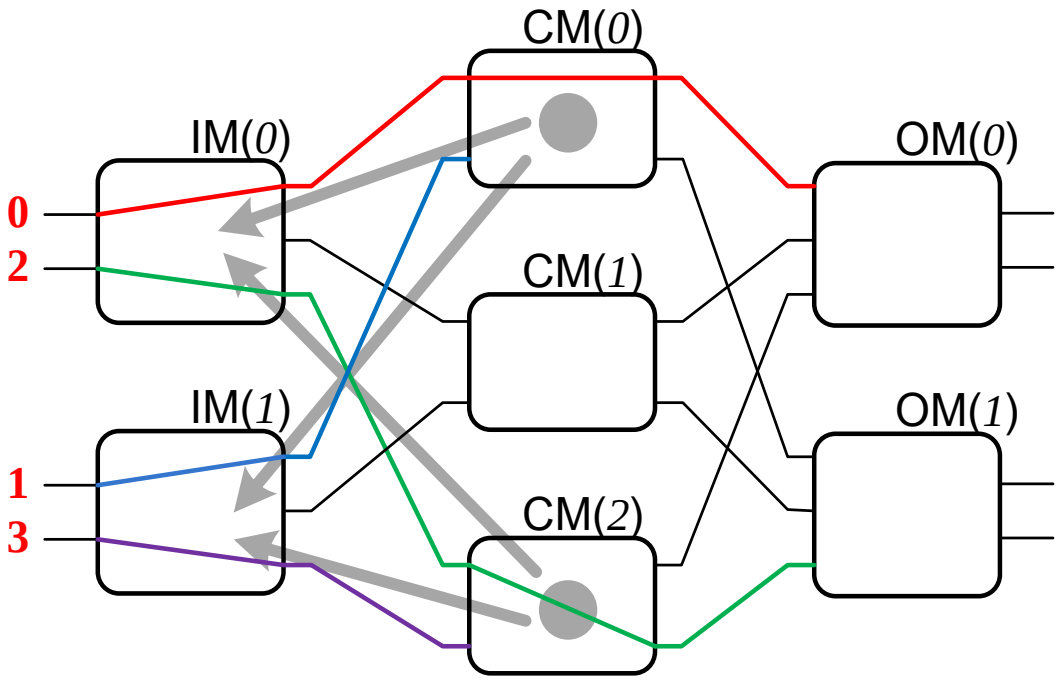
Asynchronous Dispatching (AD)

- IM alg.
  - **IPs request**
  - **LIs select**
  - OK? Send data
  - Fail? Go back
- CM alg.
  - **CMs grant LIs**
  - Update status feedback



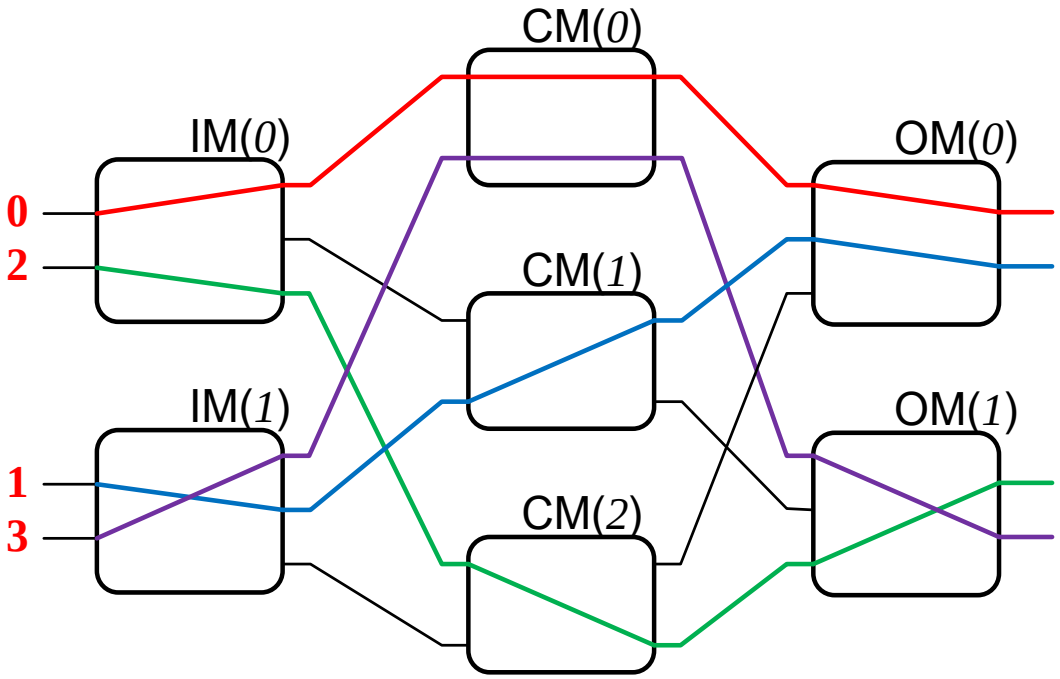
Asynchronous Dispatching (AD)

- IM alg.
  - IPs request
  - LIs select
  - OK? Send data
  - Fail? Go back
- CM alg.
  - **CMs grant LIs**
  - **Update status feedback**



Asynchronous Dispatching (AD)

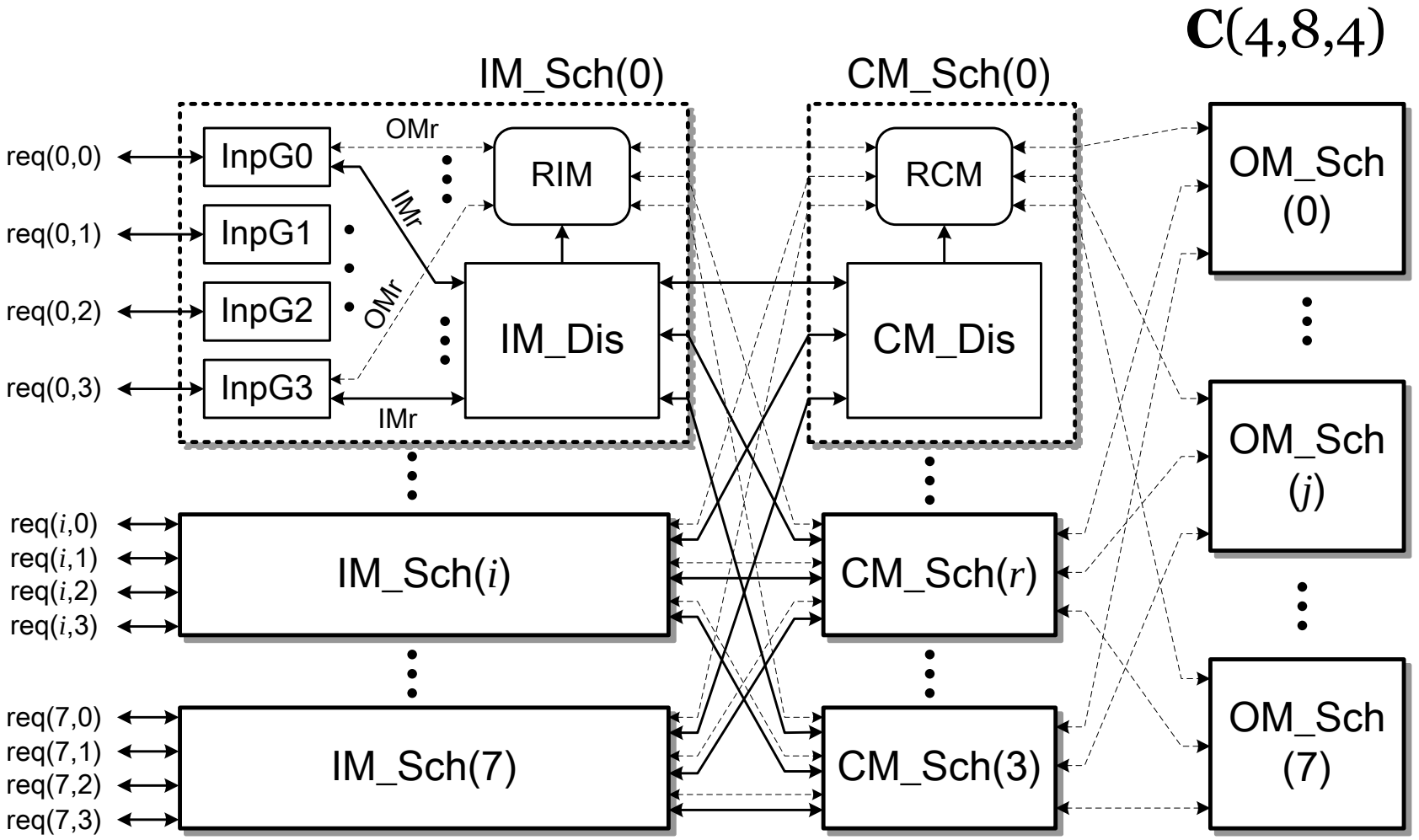
- IM alg.
  - IPs request
  - LIs select
  - **OK? Send data**
  - **Fail? Go back**
- CM alg.
  - CMs grant LIs
  - Update status feedback



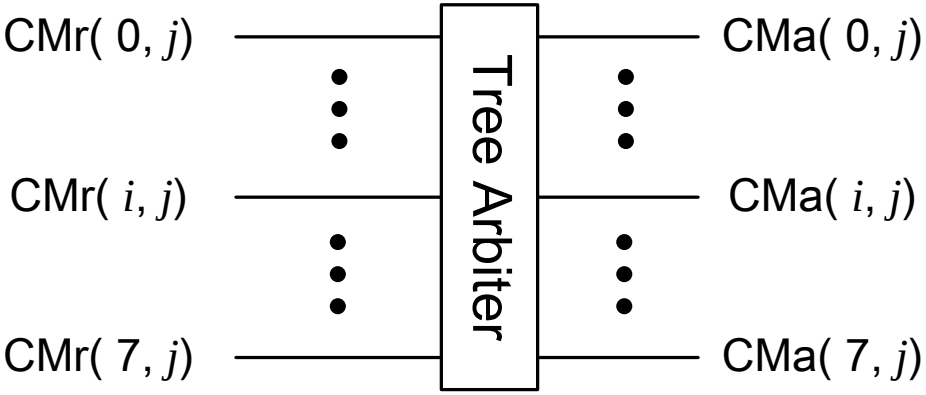
Comparison of Algorithms

- Random Dispatching (AD)
  - Cannot resolve contention in IMs or CMs.
- Concurrent Round-Robin Dispatching (CRRD)
  - Handle contention in IMs using iterations.
  - Cannot resolve contention in CMs.
- Asynchronous Dispatching (AD)
  - Contention in IMs is handled by asynchronous arbiter naturally.
  - Handle contention in CMs using status feedback.

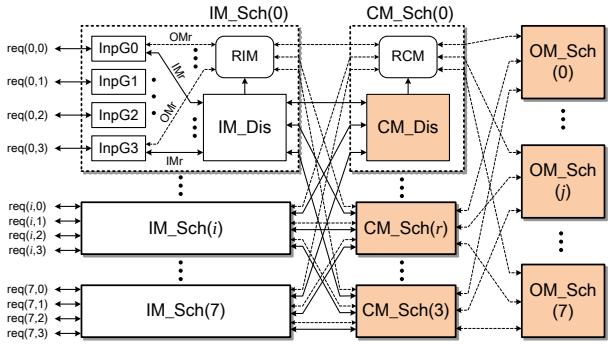
Scheduler Architecture

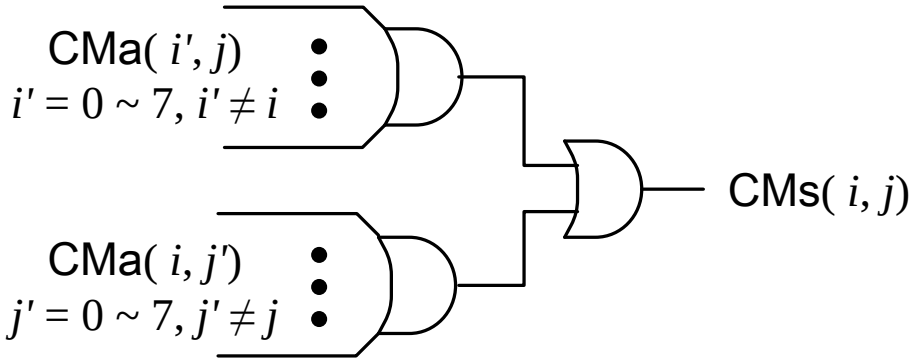


CM Dispatcher/OM Scheduler



C(4,8,4)



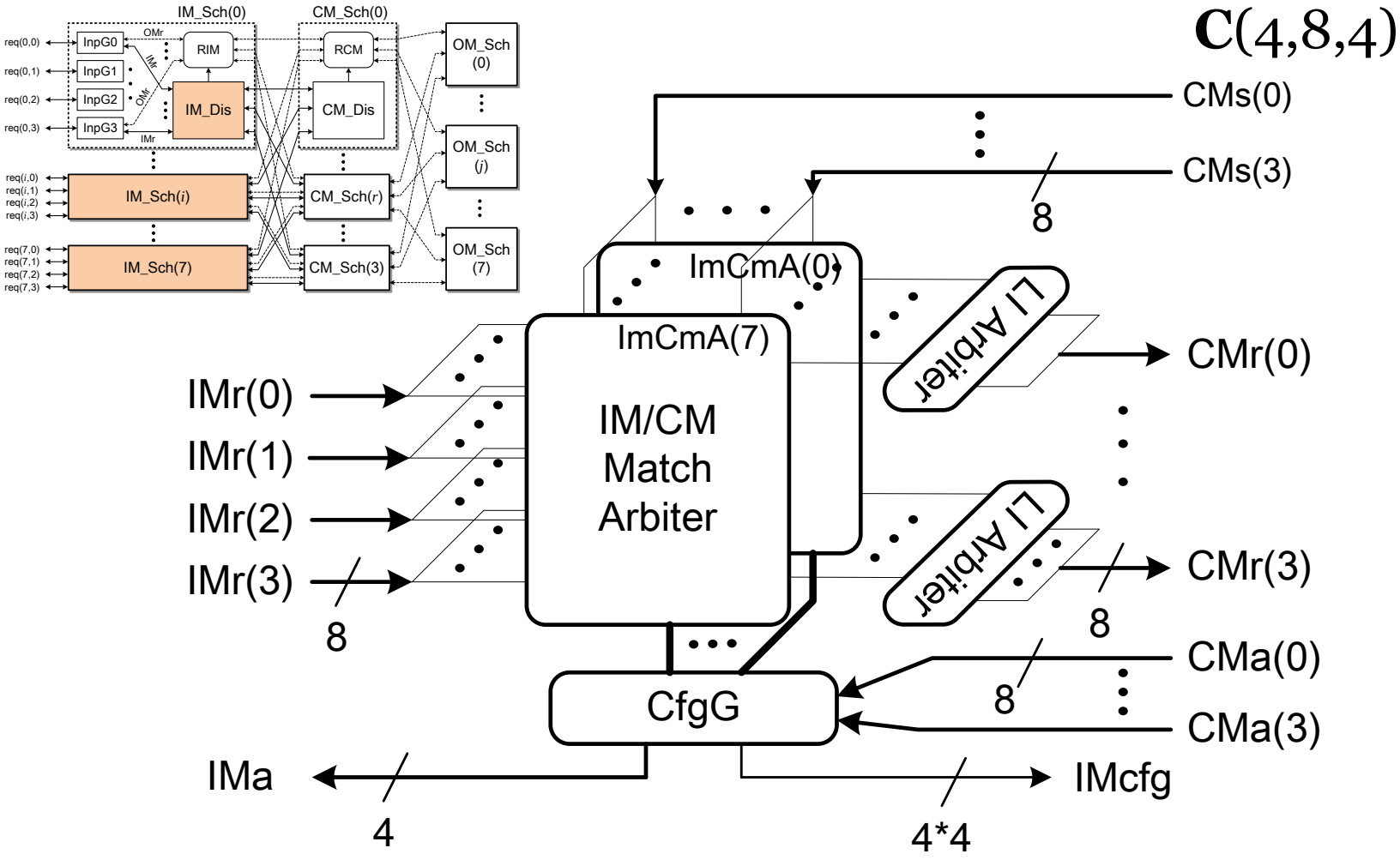


CMa: CM ack

CMs: CM status feedback

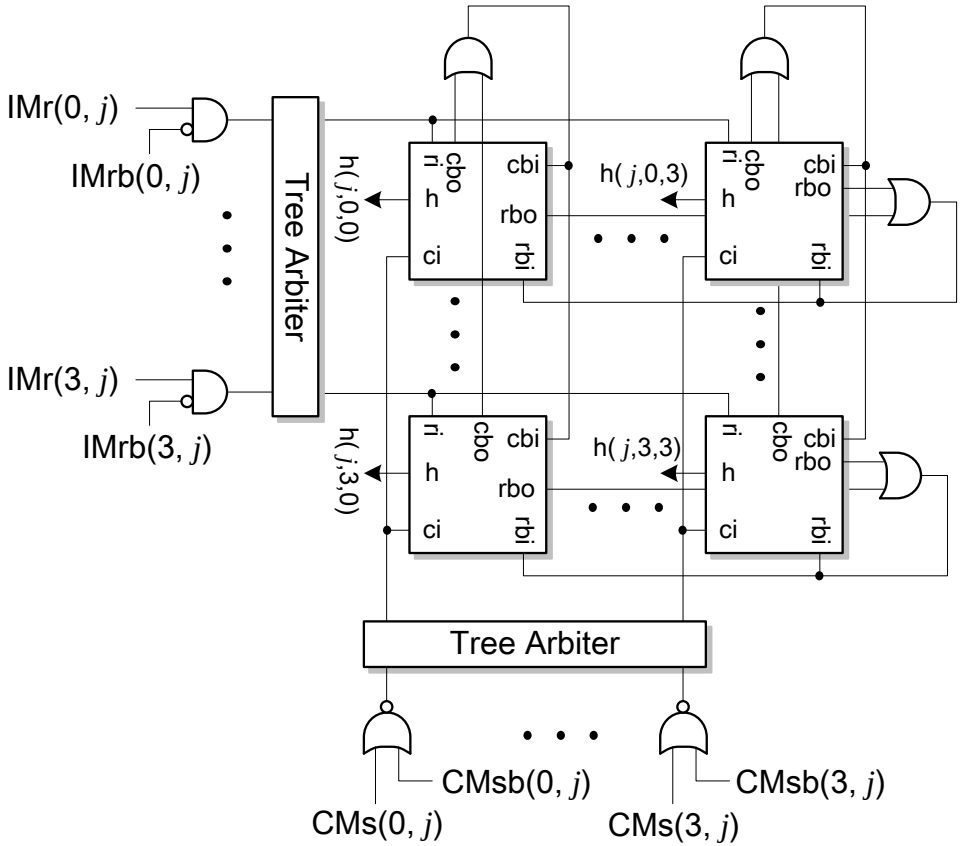
CMa and CMs are mutually exclusive

IM Dispatcher



IM/CM Match Arbiter (Multi-resource Arbiter)

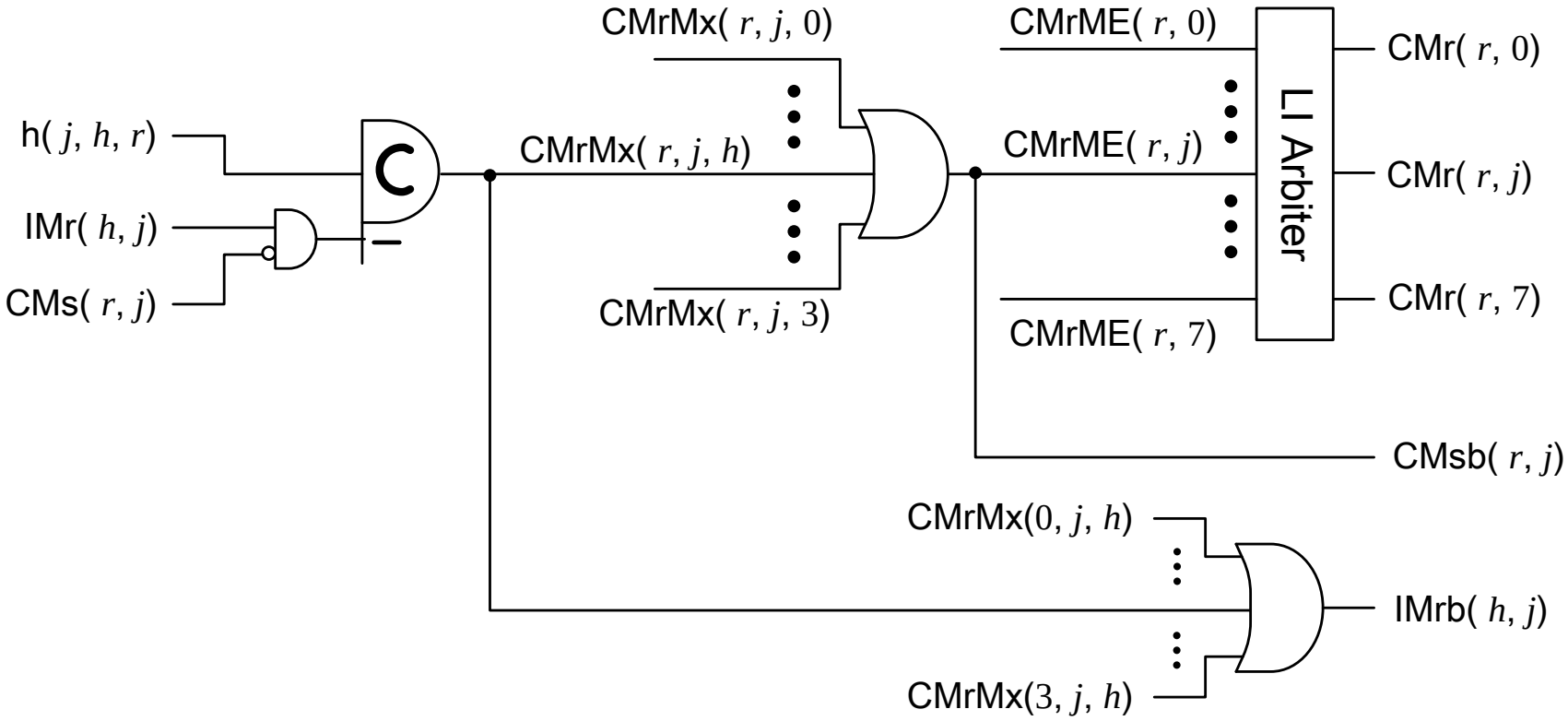
C(4,8,4)



- S. Golubcovs, D. Shang, F. Xia, A. Mokhov, and A. Yakovlev, “Modular approach to multi-resource arbiter design,” ASYNC 2009.

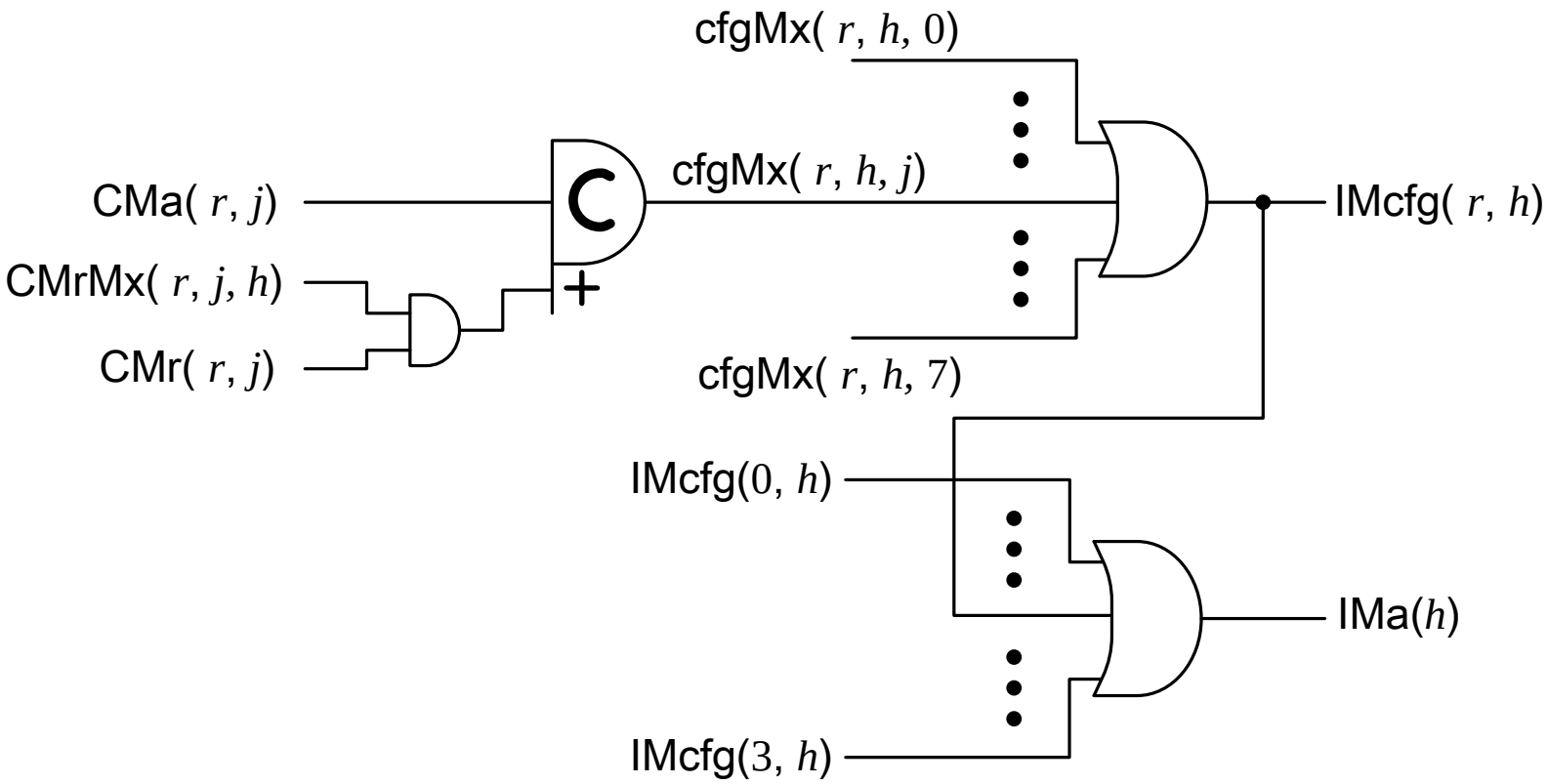
Request Generation

C(4,8,4)



Configuration Generation

C(4,8,4)

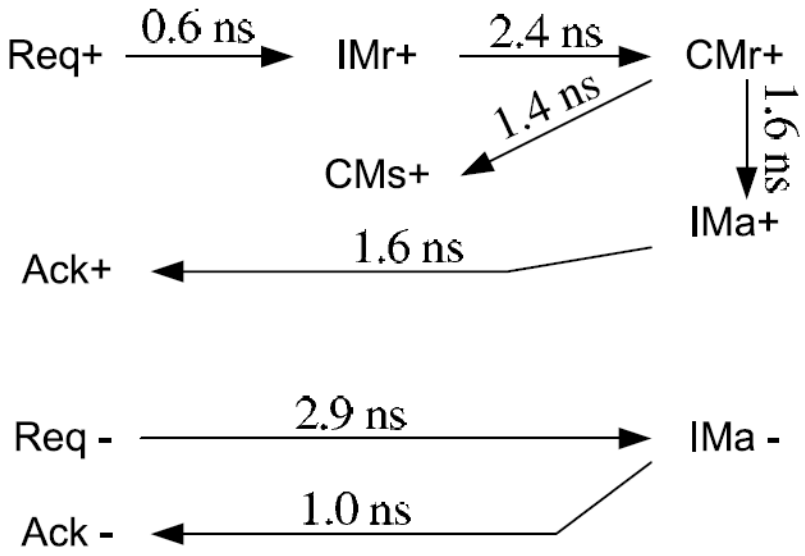


Area and Speed

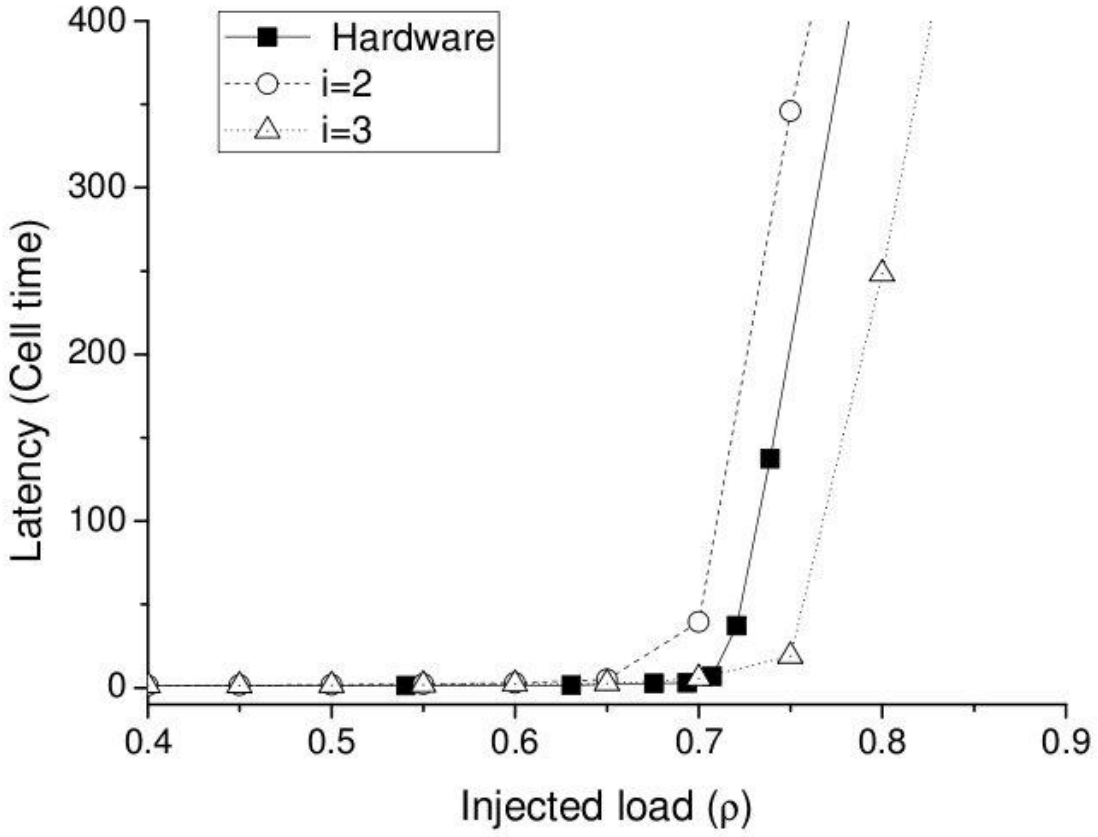
	area (um^2)	gate count (NAND2X1)	percent
InpG	6.26K	1.6K	2.5%
IM_Dis	175.06K	43.8K	67.2%
CM_Dis	33.75K	8.4K	12.9%
OM_Sch	10.06K	2.5K	3.8%
RIM & RCM	13.46K	3.4K	5.2%
other	22.15K	5.5K	8.4%
Total	260.74K	65.2K	

C(4,8,4)

Dispatching:
setup 4.6 ns release 2.9 ns
OM Scheduler:
setup 1.6 ns release 1.0 ns
Total:
setup 6.2 ns release 3.9 ns



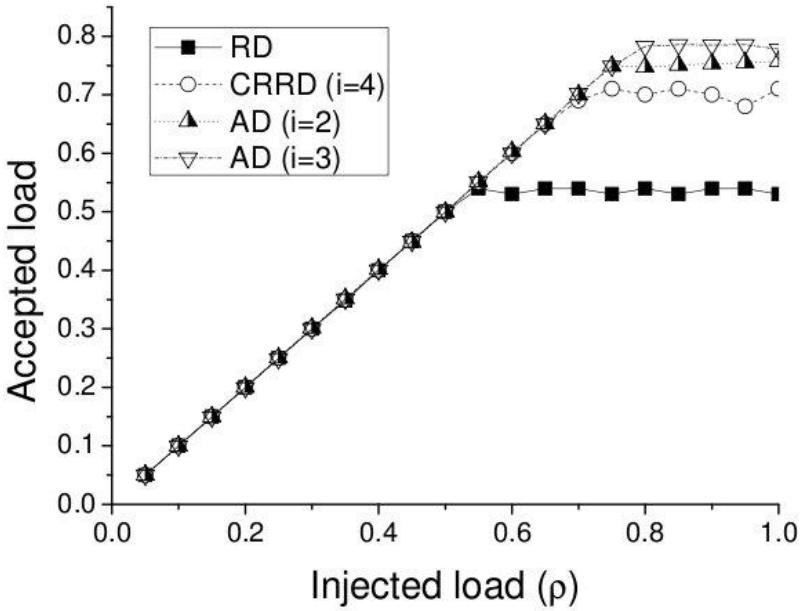
MATLAB Simulation Model

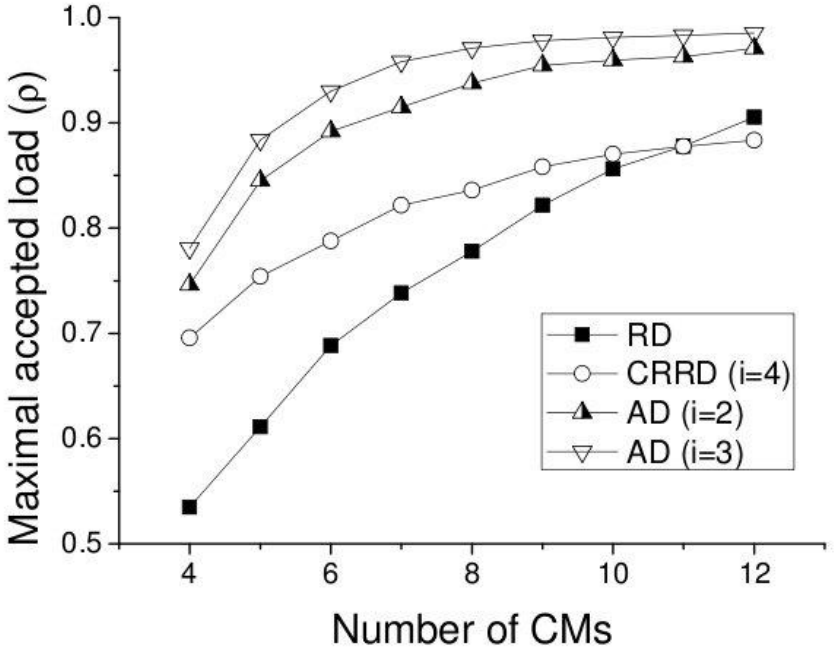


MANCHESTER 1824

The University of Manchester

Throughput Analysis





C(4,8,4) non-blocking load
RD 55%
CRRD (i=4) 70%
AD (i=3) 76%

C(4,8,*m*) non-blocking load
RD (*m*=7) 74%
CRRD (*m*=7) 82%
AD (*m*=7) 96%

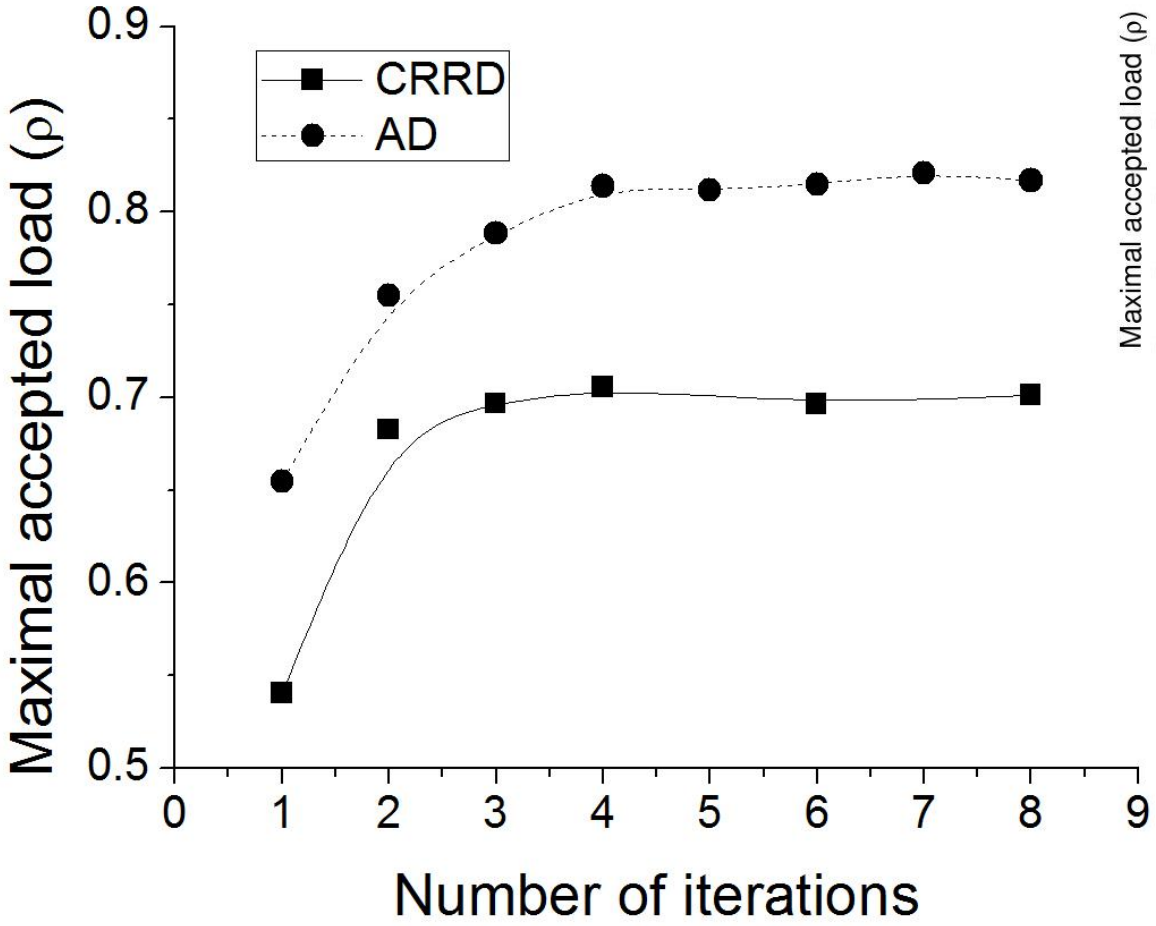
Conclusions

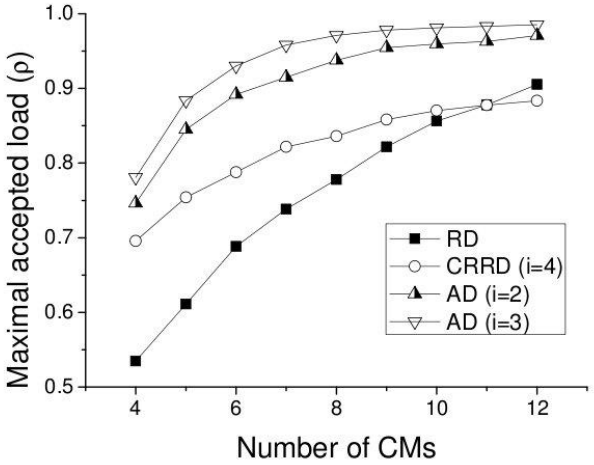
- The first asynchronous routing algorithm for general 3-stage Clos networks.
- Better throughput performance than RD and CRRD
- Future works
  - Optimize the Clos structure
  - Reduce area and latency

Thank you!

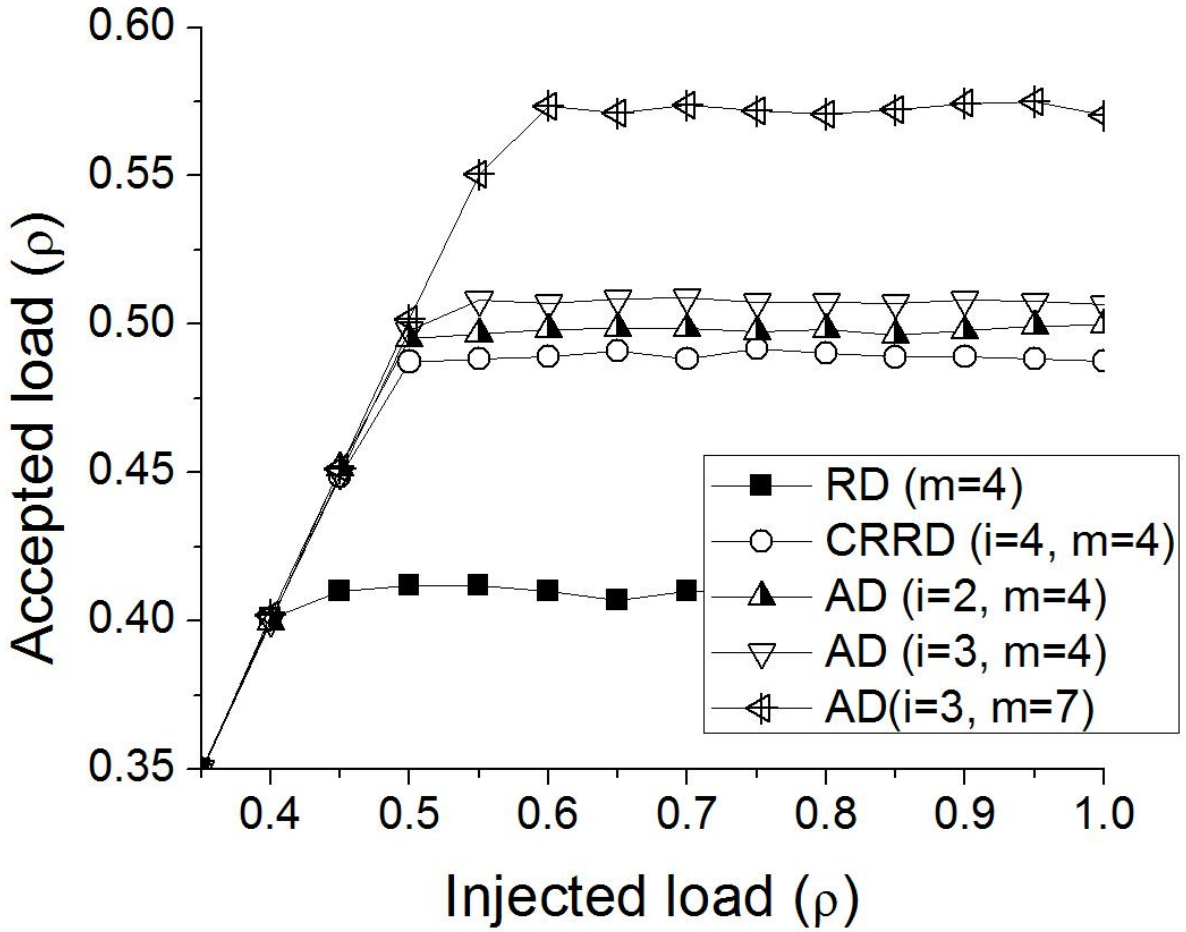
Questions?

Iterations


CRRD
AD
Maximal accepted load (ρ)
Number of iterations
0.5
0.6
0.7
0.8
0.9
0
1
2
3
4
5
6
7
8
9


RD
CRRD (i=4)
AD (i=2)
AD (i=3)
Maximal accepted load (ρ)
Number of CMs
0.5
0.6
0.7
0.8
0.9
1.0
4
6
8
10
12

Uniform Traffic



Results after Optimization

- Async
  - Setup 3.7 ns
  - Reset 3.2 ns
  - Period ~ 7 ns (140M)
  - 28K Gates
- Sync
  - 350 MHz
  - Cell time = Iter+1 =5 (70M)
  - 22K Gates
- Optimization
  - Replace multi-resource arbiter
  - Eager request (InpG)
  - MUTEX arbiter
  - Optimized tree arbiter