



Hewlett Packard
Enterprise

Practical and Efficient Incremental Adaptive Routing for HyperX Networks

Nic McDonald, Mikhail Isaev, Adriana Flores, Al Davis, John Kim

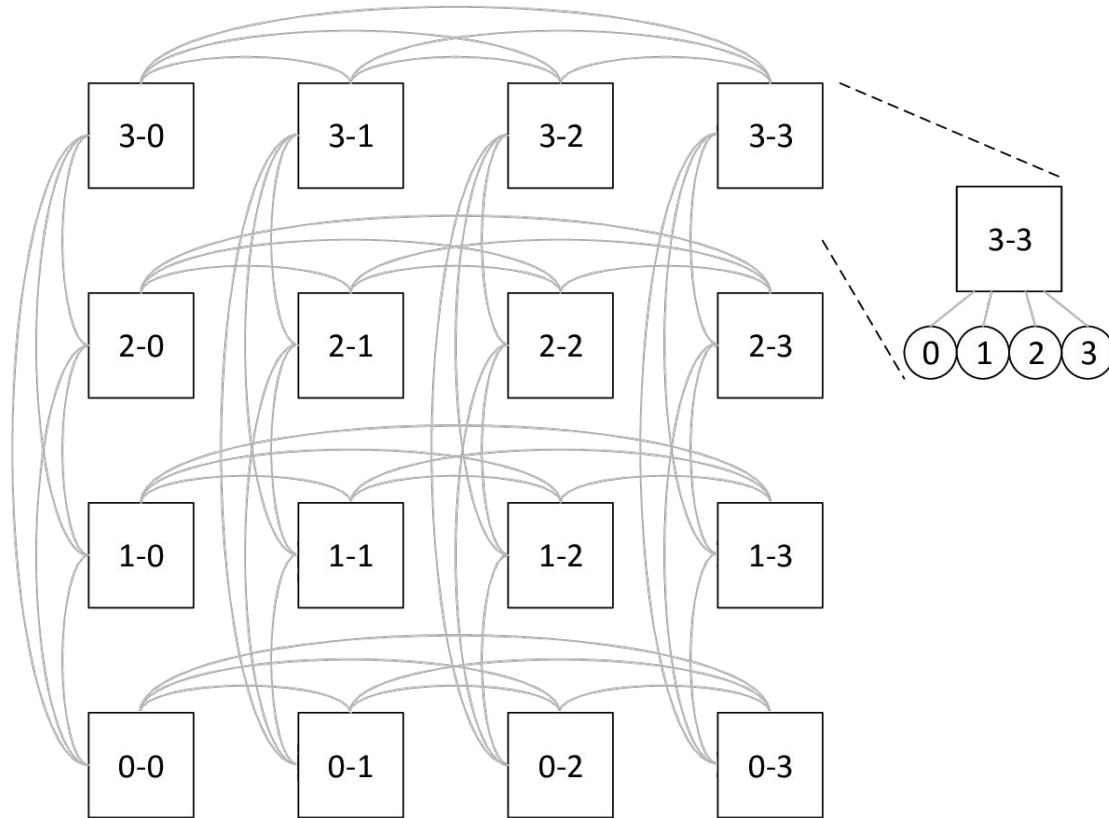
SC19 - November 19, 2019



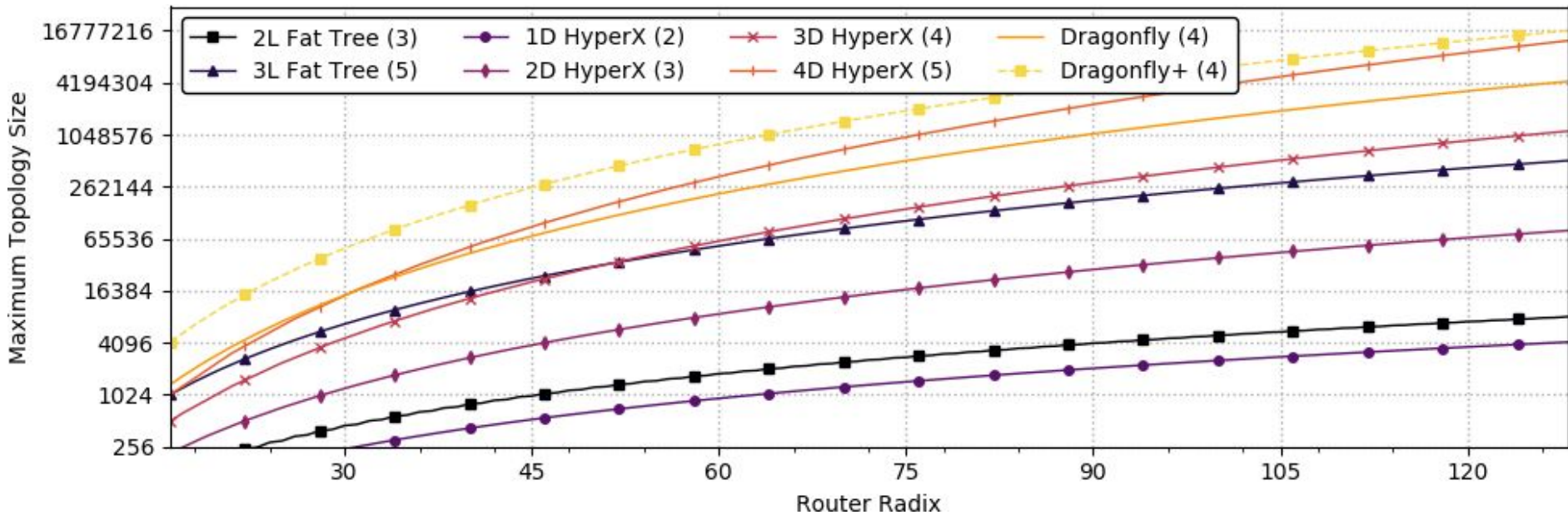
HyperX Networks



What is a HyperX Network?



Why HyperX Networks?



Why HyperX Networks?

2008 - AOCs

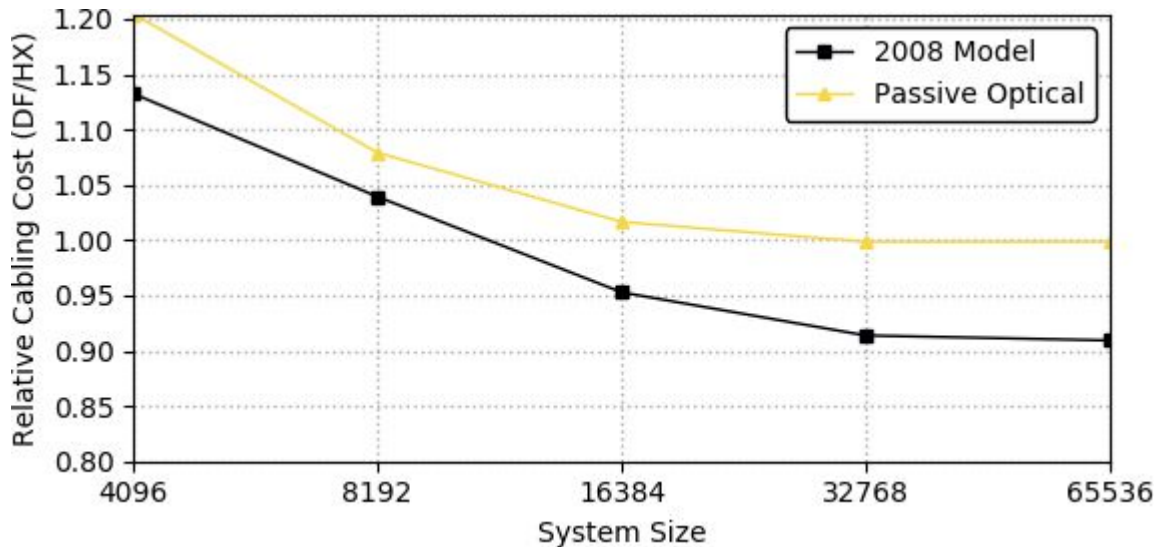
Dragonfly is ~10% cheaper

2019 - AOCs

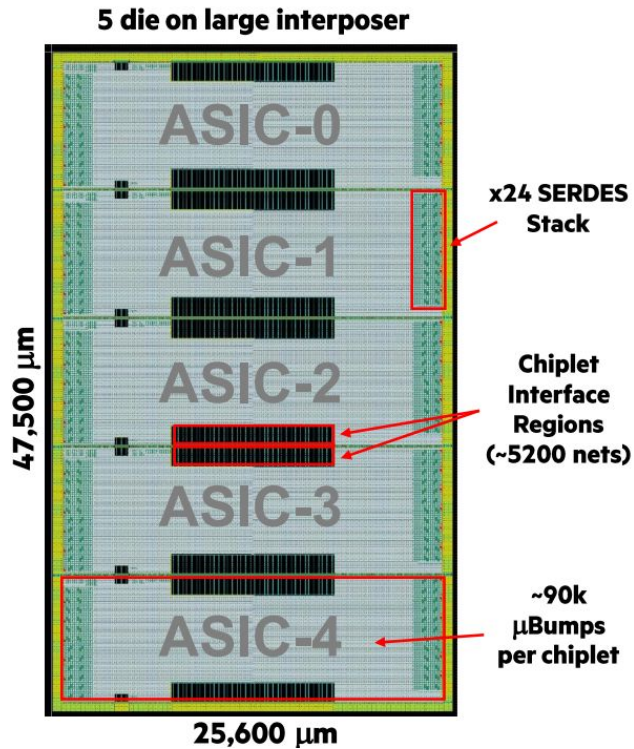
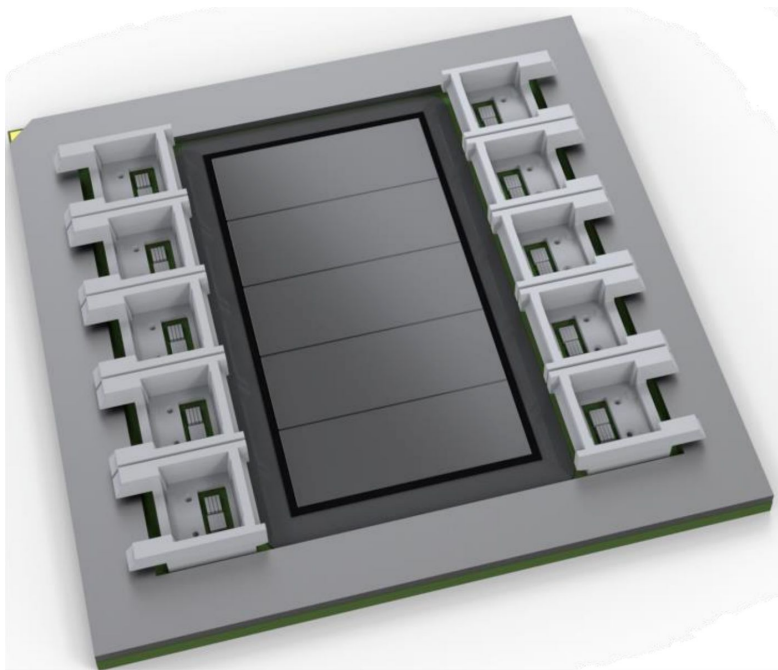
Dragonfly is ~3-5% cheaper

2019 - POCs

Dragonfly and HyperX are same cost



Switch with Integrated Optics



Why HyperX Networks?

Fat Tree

1:1 is performant yet expensive

2:1 is not performant yet cheap

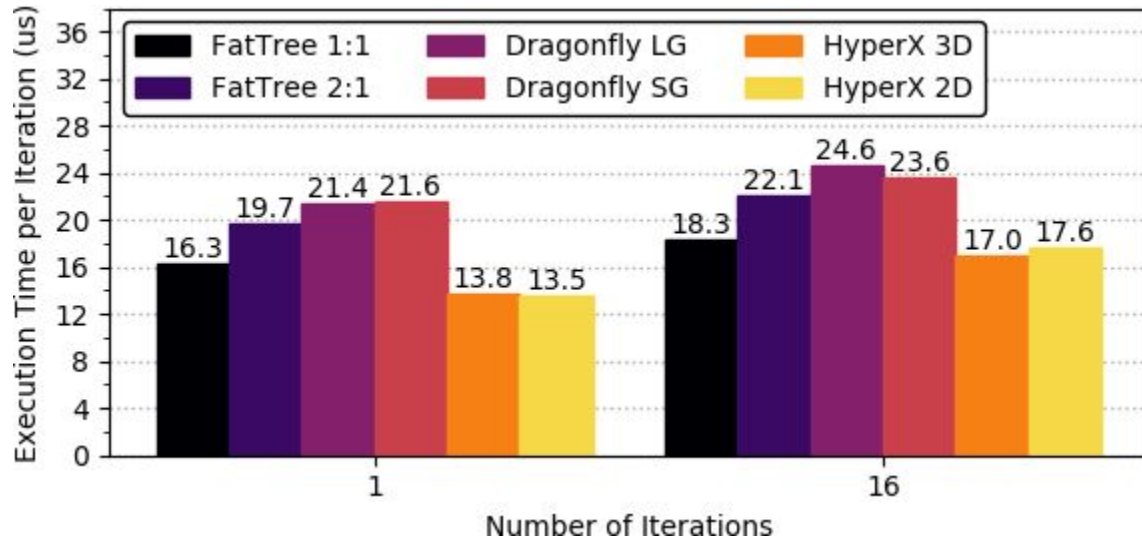
Dragonfly

Cost similar to 2:1 Fat Tree but worse performance

HyperX

Cost similar to 2:1 Fat Tree and better performance than 1:1 Fat Tree

27-Point Stencil Workload*



* workload explained in later slides

Adaptive Routing



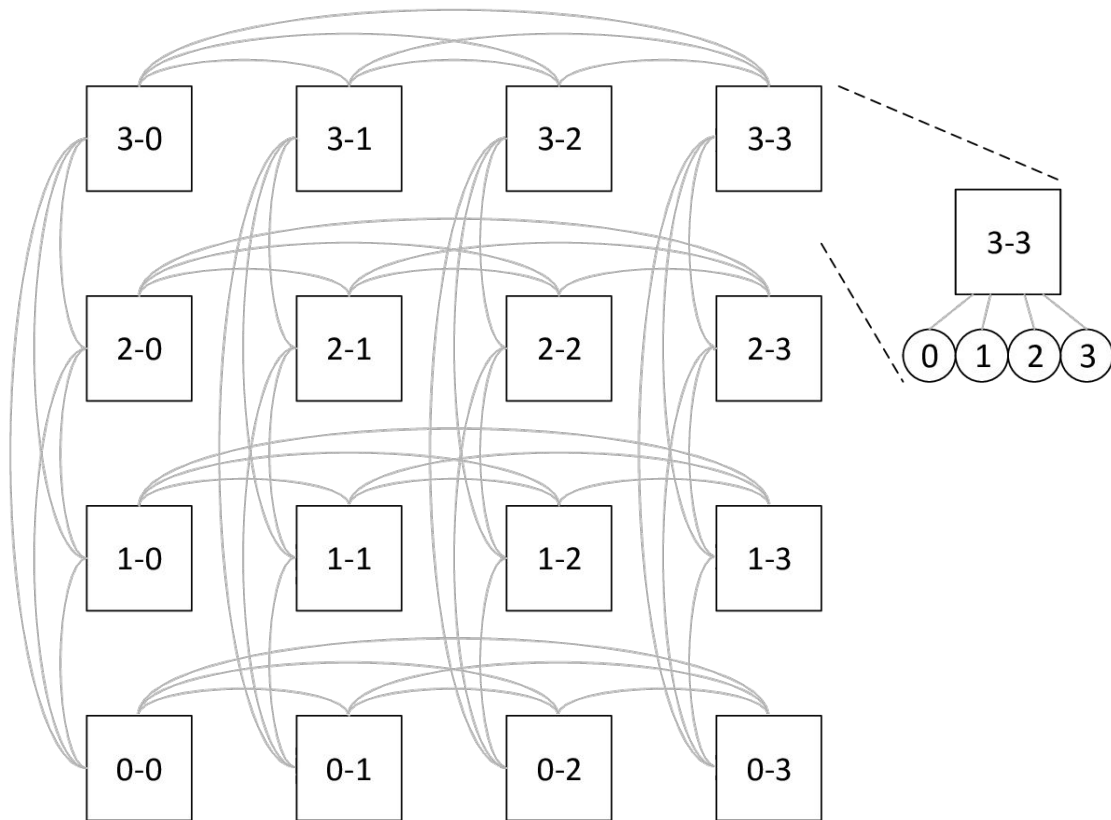
Adaptive Routing

Definition

Route packets using network state information.

Goals

Increase throughput
Decrease latency



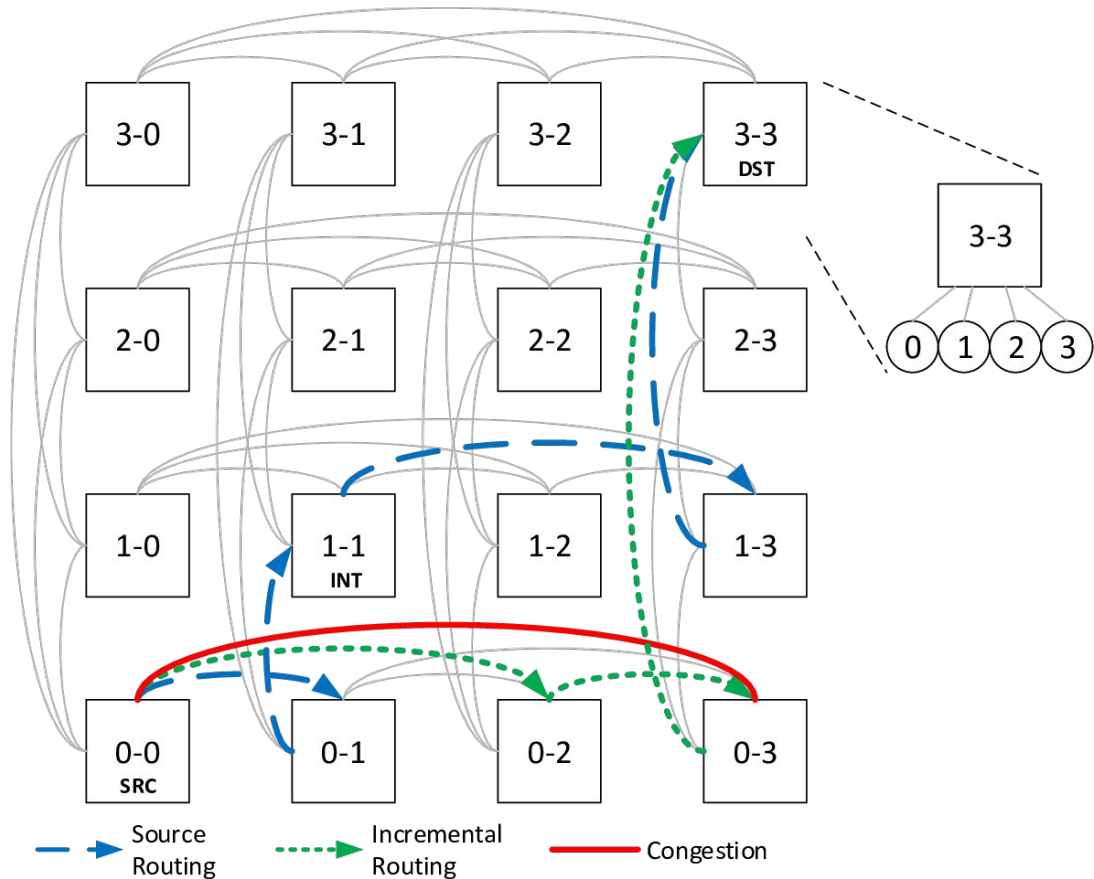
Non-Minimal Adaptive Routing

Source

One big decision at the source router

Incremental

A small decision at every router



Prior Work

UGAL

De facto standard for global adaptive routing. Choose between DOR and VAL.

Source adaptive.

Clos-AD

Flattened Butterfly paper proposed Clos-AD as UGAL with 3 optimizations.

Not implementable in high radix switch architectures due to sequential allocation.

Source adaptive.

DAL

HyperX paper proposed dimensionally adaptive load balancing (DAL).

Not implementable in high radix switch architectures due to reliance on escape path deadlock avoidance or atomic queue allocation.

Incremental adaptive.





DimWAR & OmniWAR



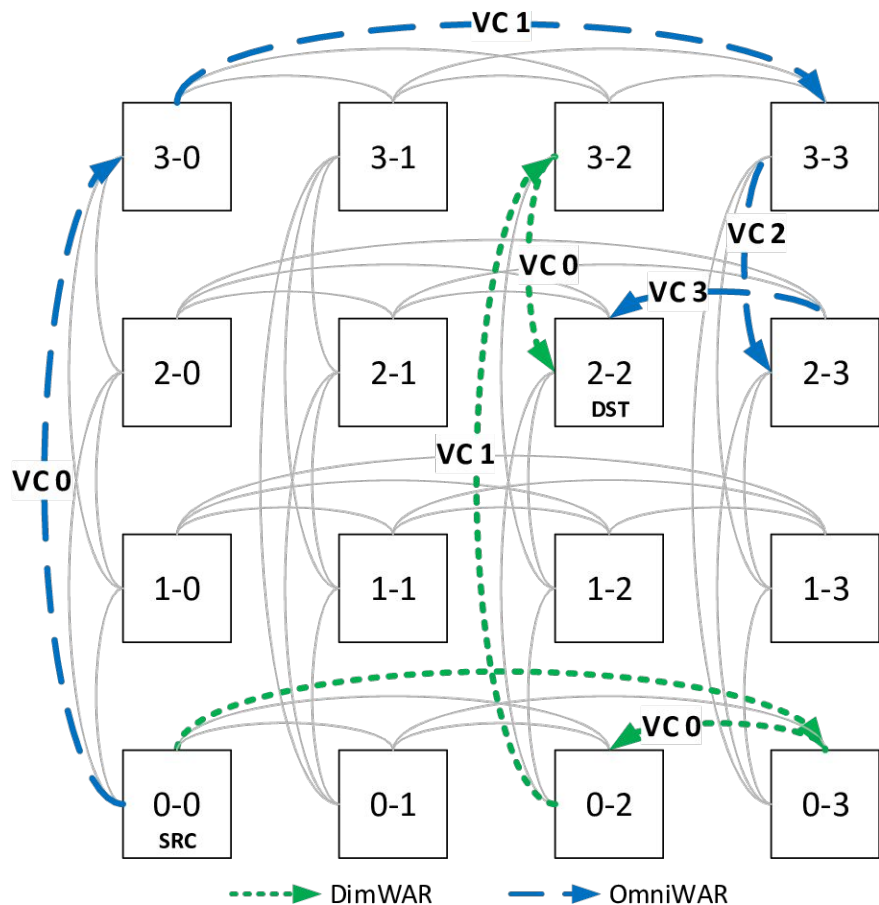
Weighted Adaptive Routing (WAR)

DimWAR

Traverse the network in dimension order.
Use weighted decision in every dimension.
Deroute at most once on any dimension.
Requires 2 VCs.

OmniWAR

Traverse any dimension at any time.
Use weighted decision in every dimension.
Reserve N total deroutes. Take them anytime.
Requires 2N VCs.
*N is # of dimensions



Implementation Analysis

Algorithm	Dimension Ordered	Routing Style	VCs Required	Deadlock Handling	Architecture Requirements	Packet Contents
UGAL	yes	source	2	RR & RC	-	int. addr.
Clos-AD	yes	source	2	RR & RC	seq. alloc.	int. addr.
DAL	no	incremental	1+1e	escape paths	escape paths	N-bit field
DimWAR	yes	incremental	2	RR & RC	-	-
OmniWAR	no	incremental	2N	RR & DC	-	-

Both DimWAR and OmniWAR do not require any special architectural features or special packet contents



Experimentation



Experimentation

SuperSim - github.com/nicmcd/supersim

Network

4,096 node 3D HyperX - 8x8x8 concentration of 8

8 virtual channels

10m (50ns) router-to-router channels

1m (5ns) router-to-terminal channels

Packet buffer flow control

Router

Combined input/output queued (CIOQ) architecture

Age-based virtual channel allocation

Age-based crossbar scheduling

56ns fall-through latency

Algorithms Under Investigation

Name	Description
DOR	Dimension Order Routing
VAL	Valiant's Randomized Routing
UGAL	Universal Global Adaptive Load-balancing
Clos-AD	Universal Global Adaptive Load-balancing optimized for HyperX
DimWAR	Dimensionally-ordered Weighted Adaptive Routing
OmniWAR	Omni-dimensional Weighted Adaptive Routing





Synthetic Workload Analysis

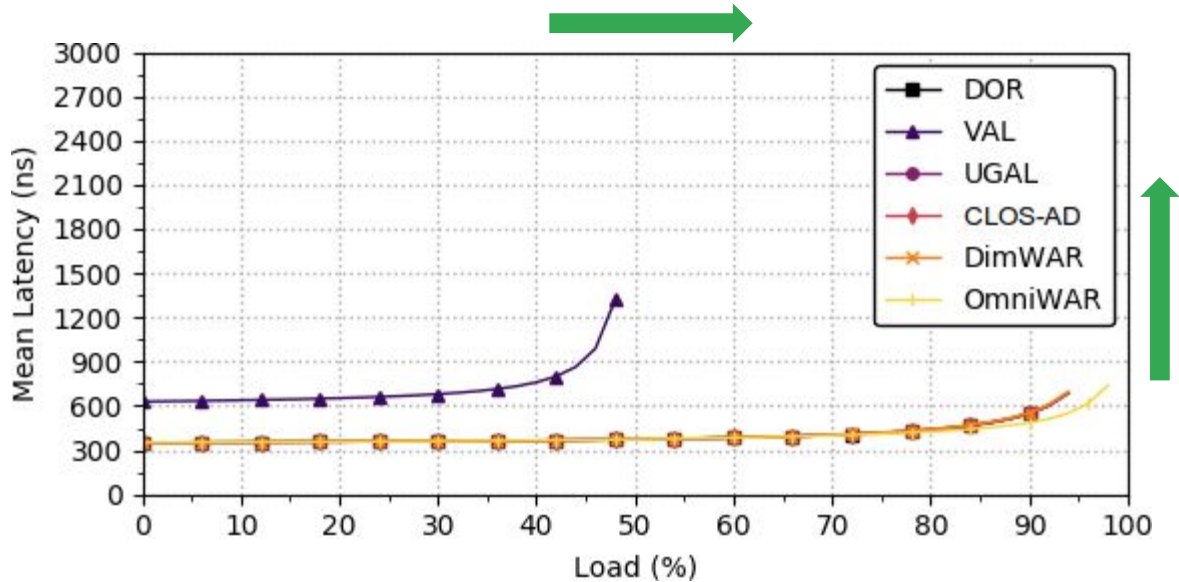


Synthetic Workload Analysis

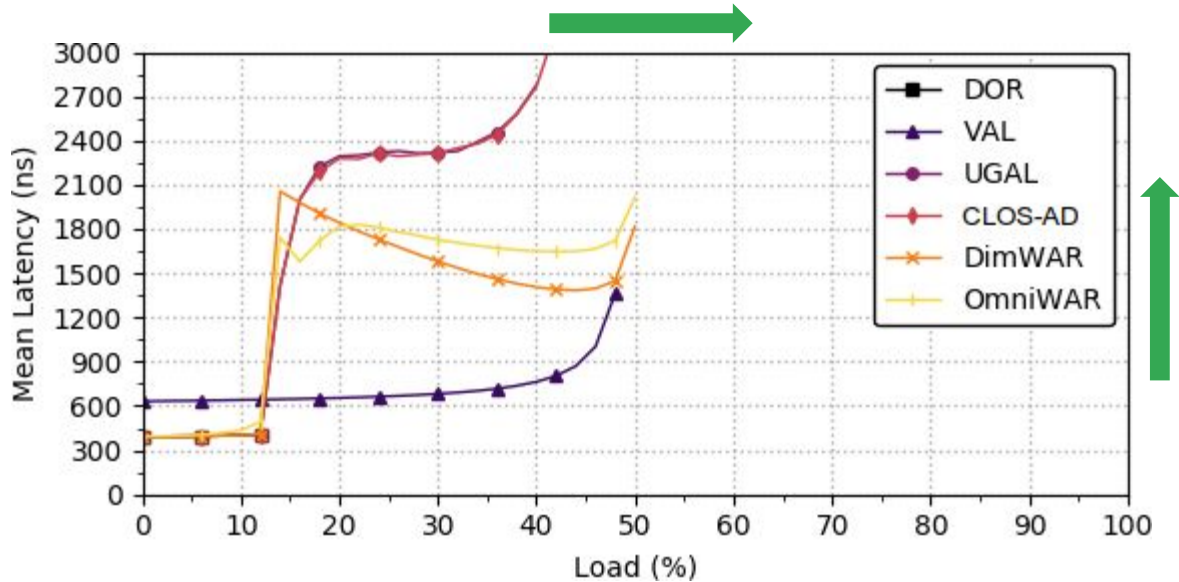
Traffic Patterns Under Investigation

Name	Description
UR	Uniform Random
BC	Bit Complement
URB	Uniform Random Bisection - destination selected using BC in the targeted dimension and UR in all other dimensions. This traffic results in one dimension being non-load-balanced and all other dimensions are load balanced. For example, URBy has UR in the X and Z dimensions and BC in the Y dimension.
S2	Swap 2 - destination selected like adversarial BC-like way but only along one dimension. Even numbered terminals use the X dimension and odd numbered terminals use the Y dimension. This presents non-load balanced traffic but there is a lot of unused bandwidth in total.
DCR	Dimension Complement Reverse - destination selected across the furthest dimensional instance of the network. All terminals in each X dimension instance distribute their traffic across a complement Z dimension instance. Worst-case admissible traffic for 3D.

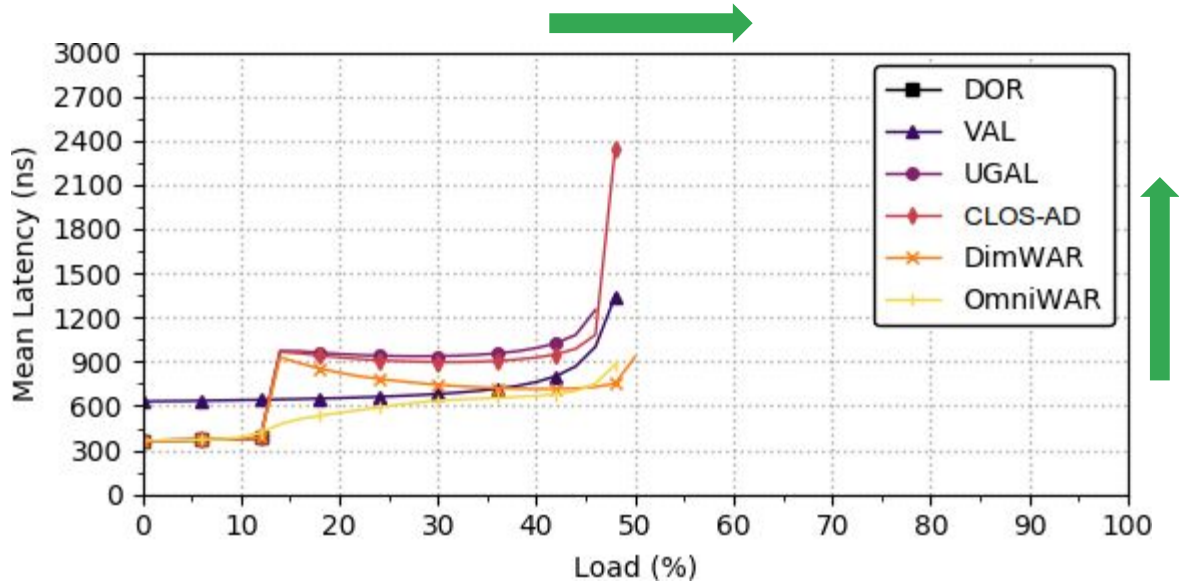
Synthetic Workload Analysis - UR



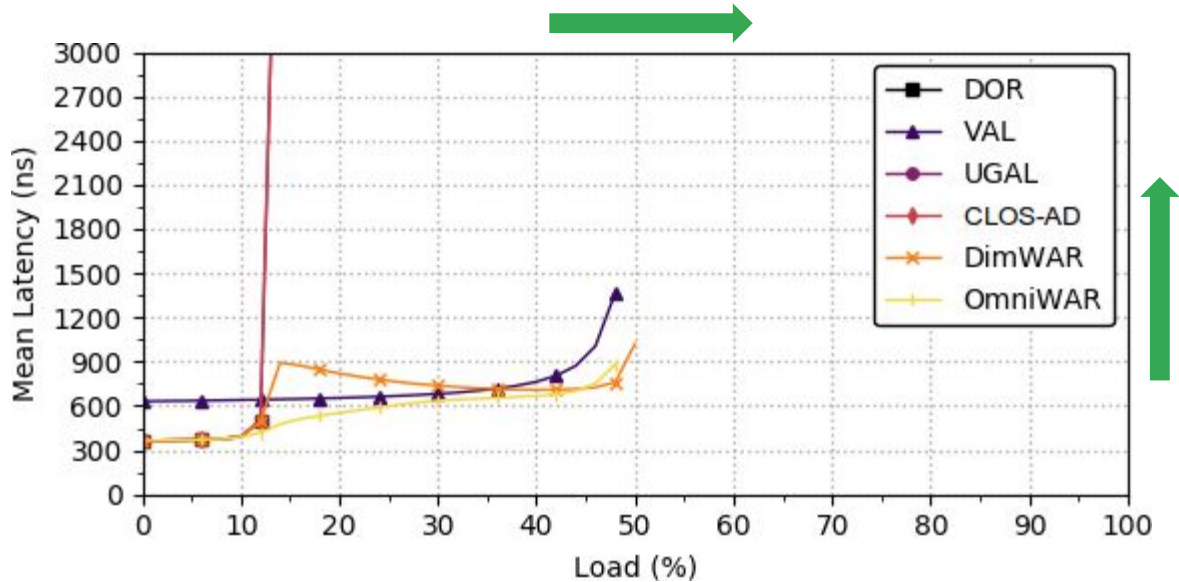
Synthetic Workload Analysis - BC



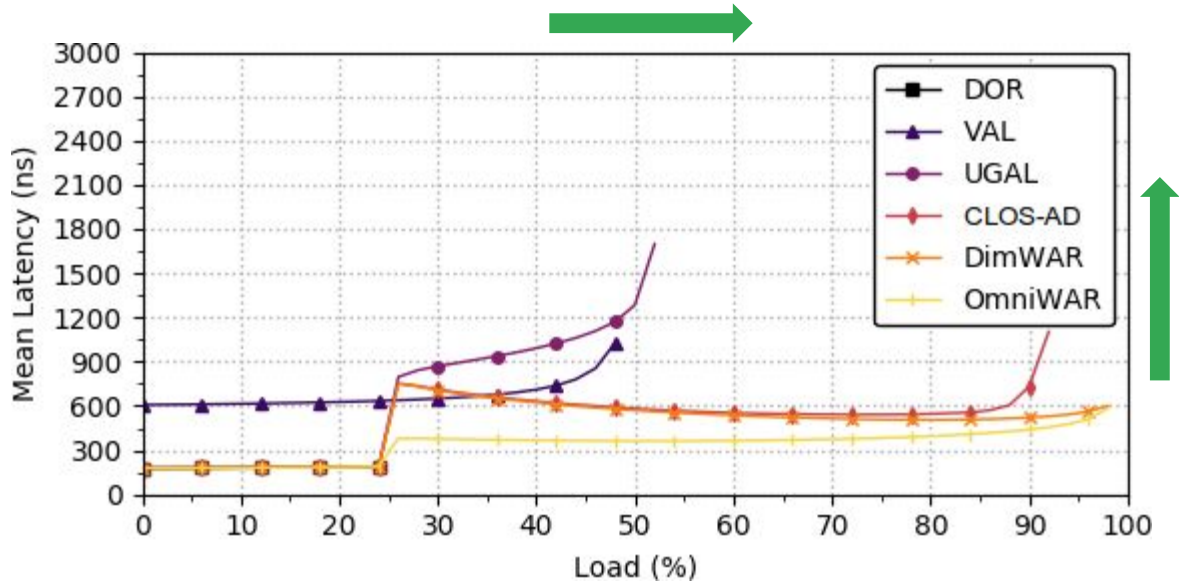
Synthetic Workload Analysis - URBx



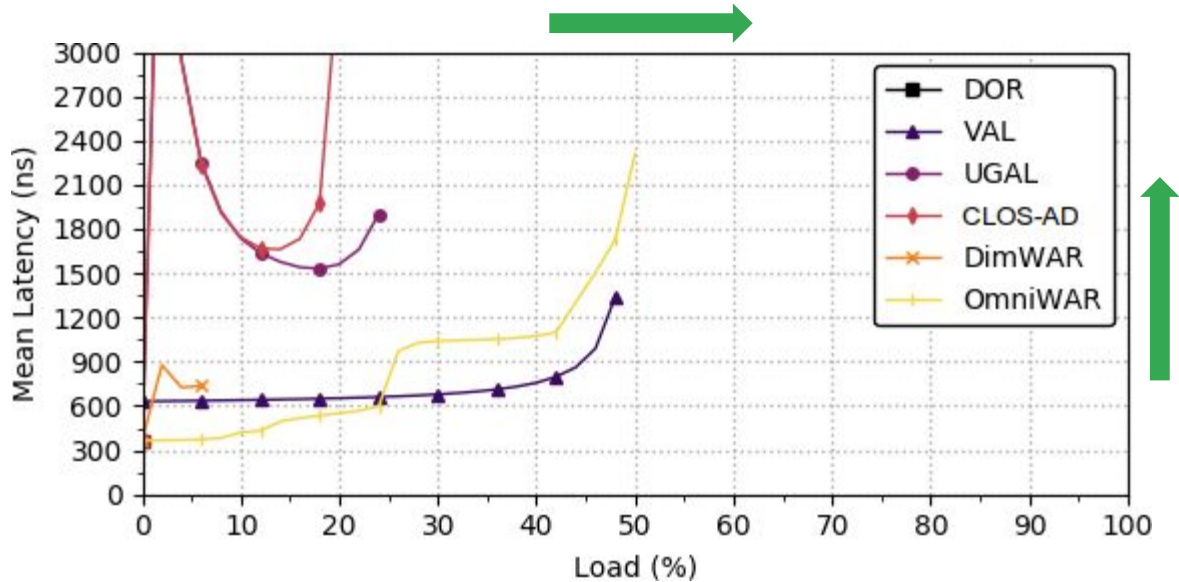
Synthetic Workload Analysis - URBy



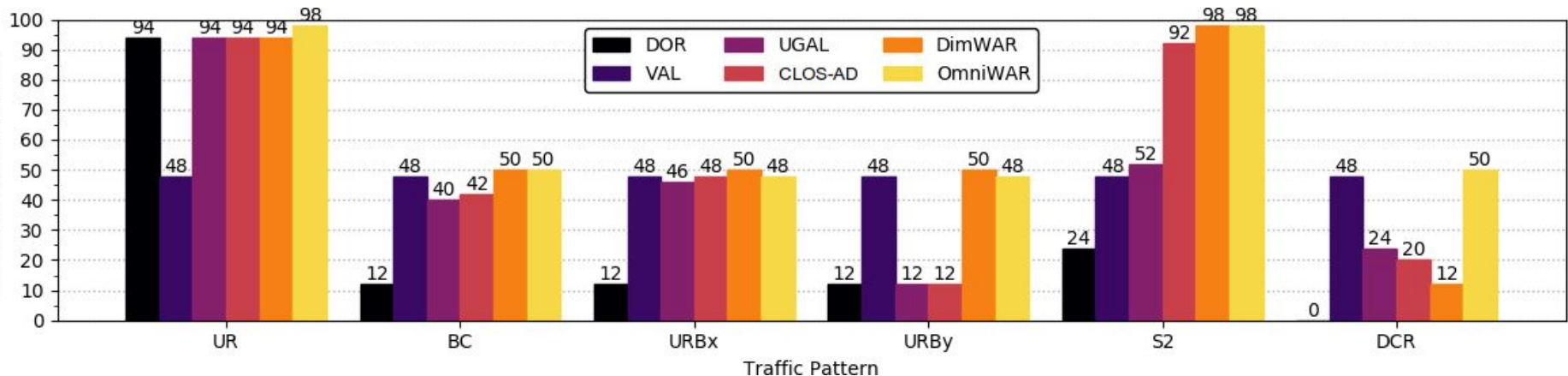
Synthetic Workload Analysis - S2



Synthetic Workload Analysis - DCR



Synthetic Workload Analysis



Already load
balanced
traffic

All
dimensions
have
bottlenecks


Dimension X
has
bottleneck,
dimensions Y
and Z are
balanced

Dimension Y
has
bottleneck,
dimensions X
and Z are
balanced

Half of links
are unused,
the other half
of the links
have
bottlenecks

Pathological
worse case
traffic pattern
for dimension
ordering



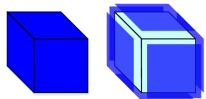


27-Point Stencil Workload Analysis

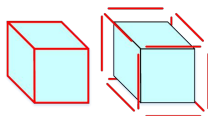


27-Point Stencil Workload Analysis

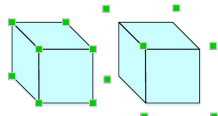
```
for (int i = 0; i < iters; i++) {
    compute();
    exchange();
    collective();
}
```



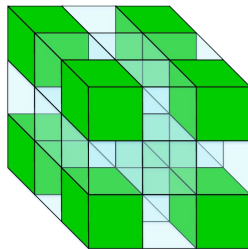
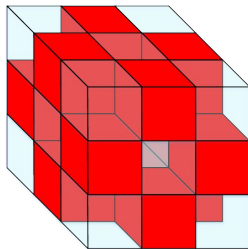
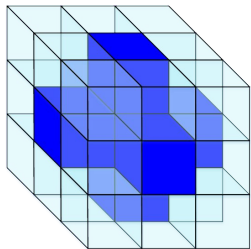
Neighbors with touching faces:
Message size = "face area" * 8 bytes



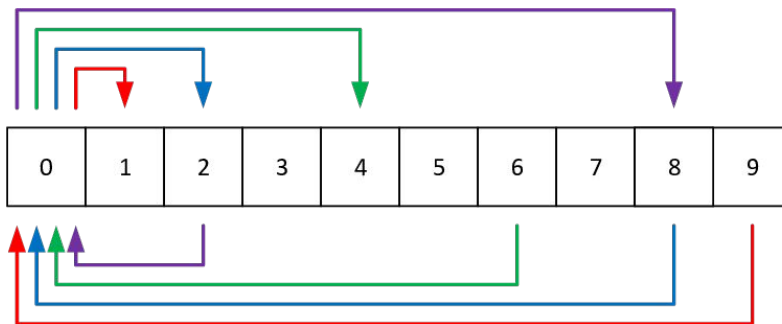
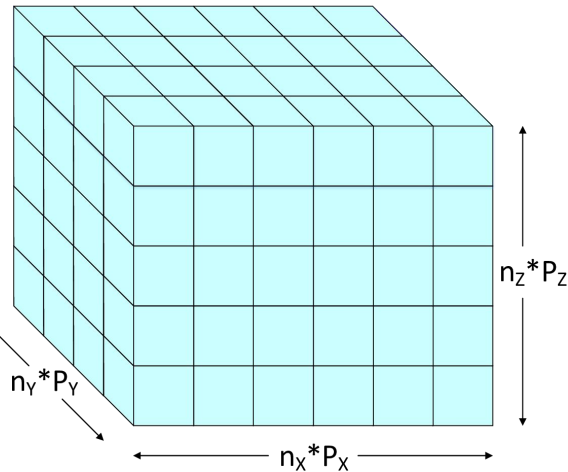
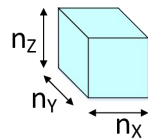
Neighbors with touching edges:
Message size = "edge-length" * 8 bytes



Neighbors with touching corners:
Message size = 8 bytes

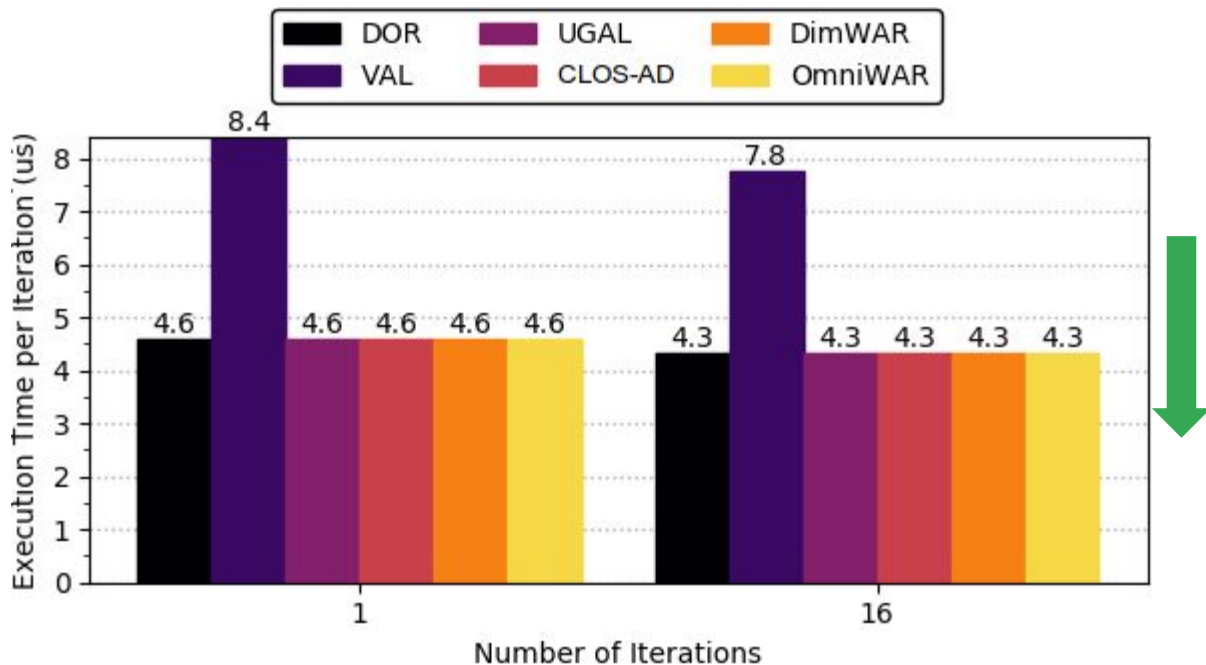


Per-process size is replicated to build up a global domain



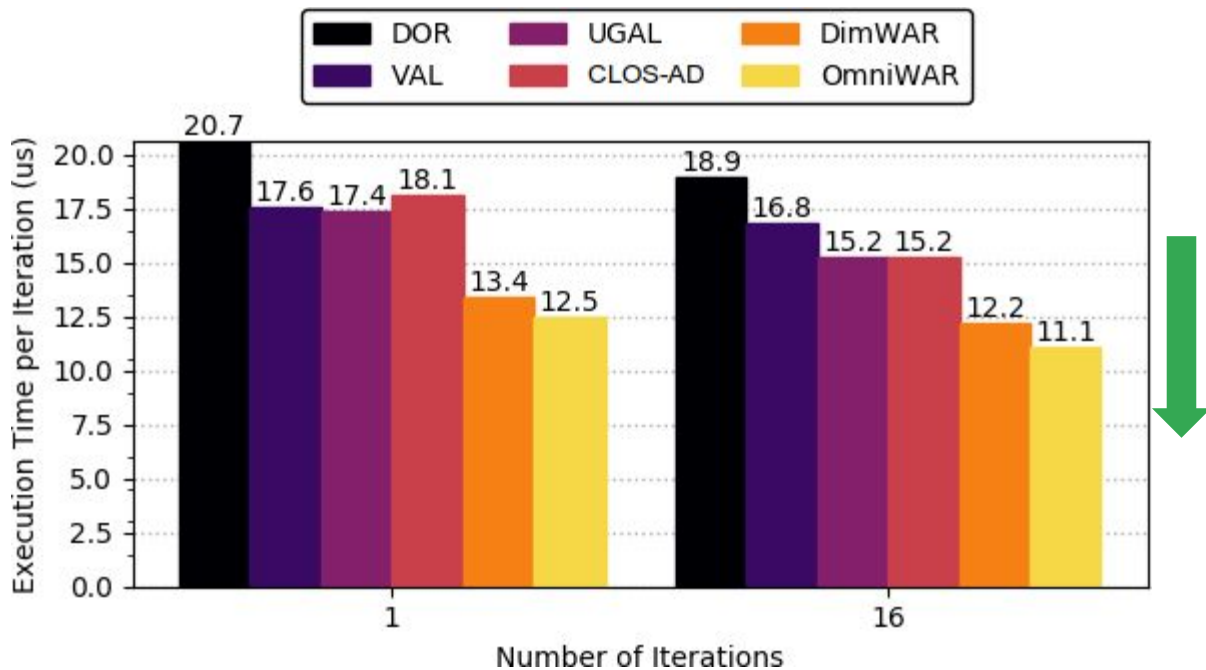
- Iteration 0
- Iteration 1
- Iteration 2
- Iteration 3

27-Point Stencil Workload Analysis - Collective



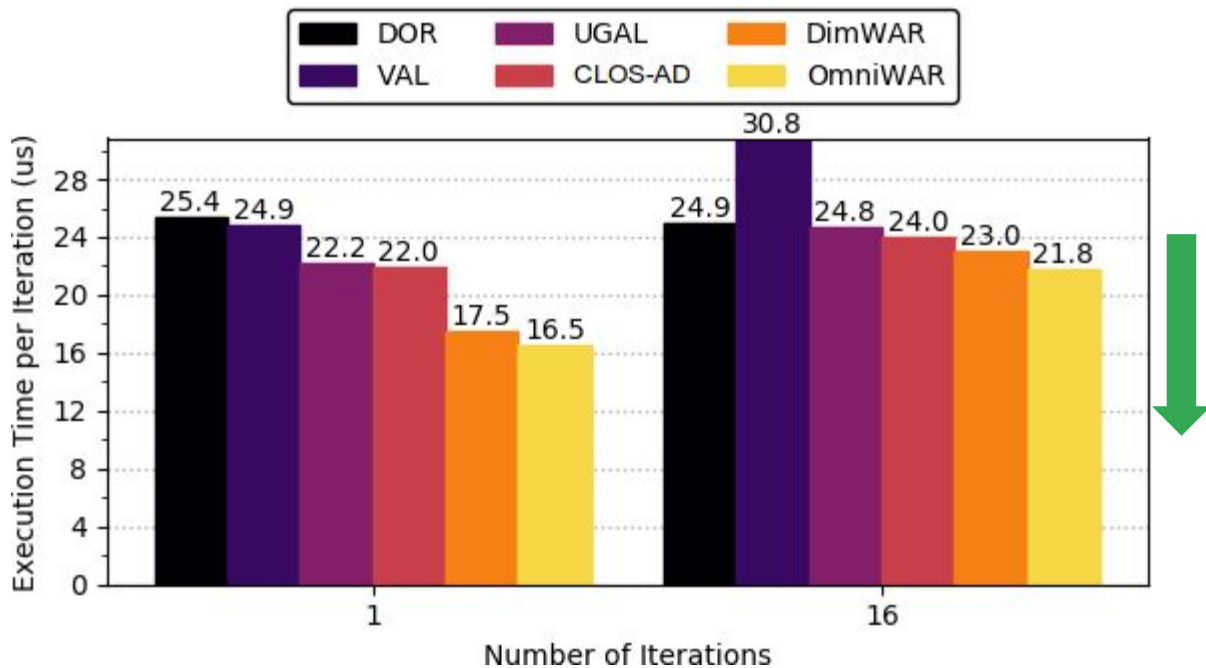
* random process placement used

27-Point Stencil Workload Analysis - 100kB Halo Exchange



* random process placement used

27-Point Stencil Workload Analysis - Collective & Halo Exchange



* random process placement used

Conclusion



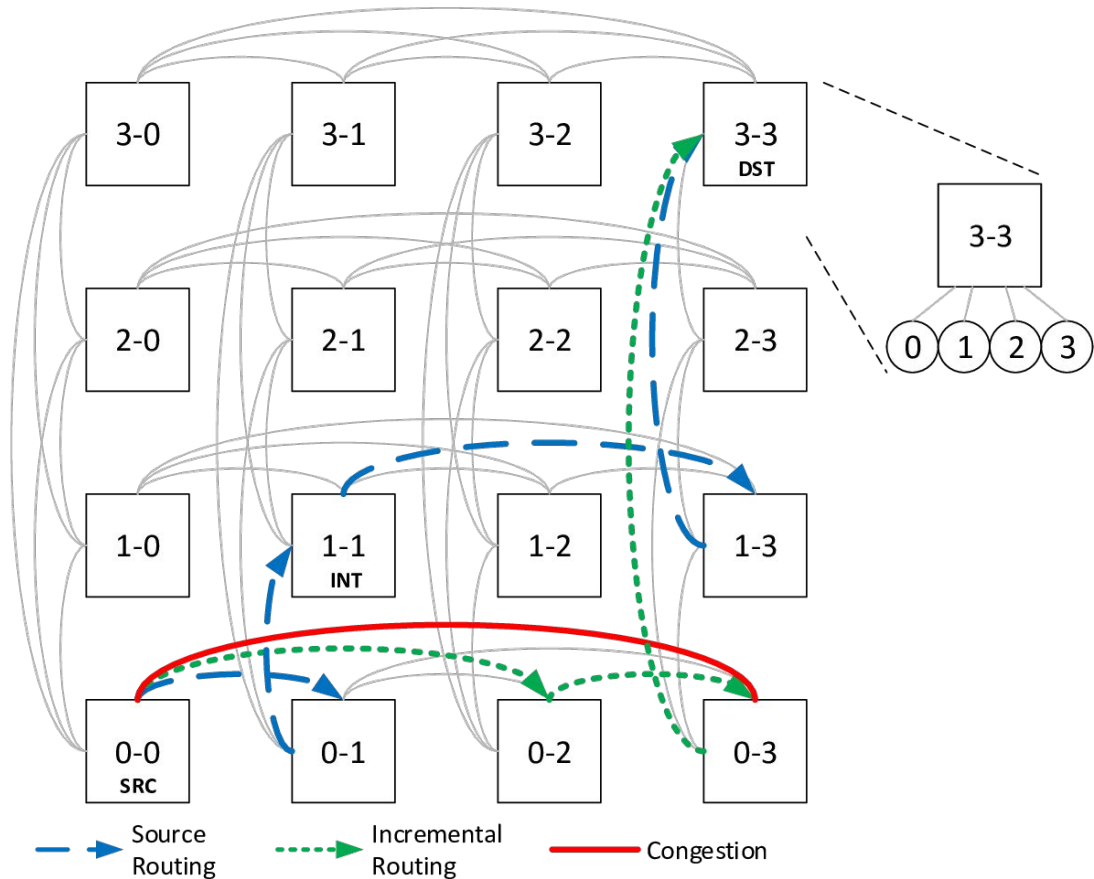
Conclusion

DimWAR & OmniWAR

Implementable in modern high radix routers and networks with VC flow control.

Up to 4x throughput increase for synthetic traffic.

Reduce communication time by 25% for 27-point stencil workloads.



THANK YOU

Nic McDonald	<u>nicm@google.com</u>
Mikhail Isaev	<u>michael.v.isaev@gatech.edu</u>
Adriana Flores	<u>adrianaf@nvidia.com</u>
Al Davis	<u>ald@hpe.com</u>
John Kim	<u>ijk12@kaist.edu</u>

