Design and Evaluation of Topology-aware Scatter and AllGather Algorithms for Dragonfly Networks

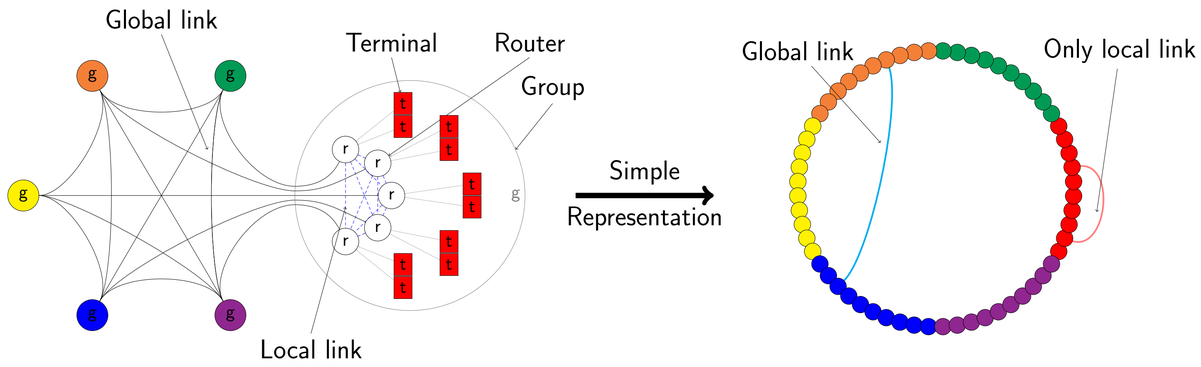
Nathanaël Cheriere, Matthieu Dorier

Design and Evaluation of Topology-aware Scatter and AllGather Algorithms for Dragonfly Networks

Nathanaël Cheriere[1], Matthieu Dorier[2].
[1] ENS Rennes, [2] Argonne National Laboratory

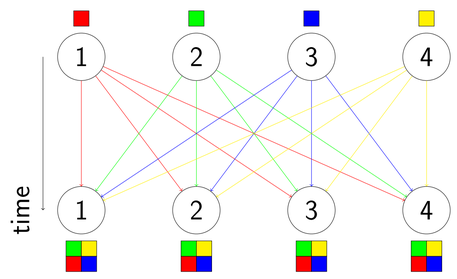
New efficient communication algorithms for Dragonfly

- Cost of traditional topologies too high at scale
- Dragonfly: a new topology
- Opportunity for new efficient communication algorithms



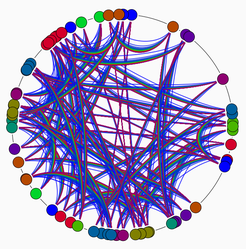
Challenge: Global links are potential bottlenecks

AllGather



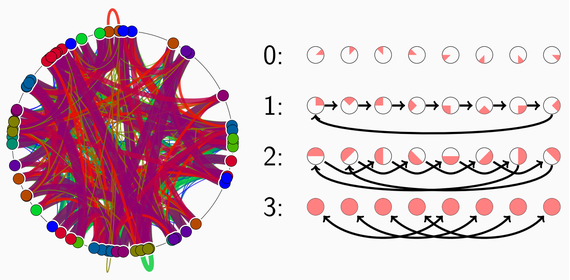
Existing algorithms

RING



Default in MPICH for 80 KiB and more

BRUCK - Bruck's algorithm



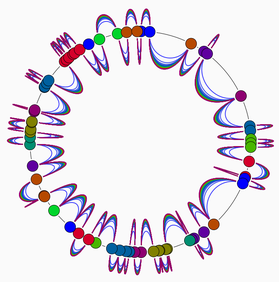
Default in MPICH for less than 80 KiB

Our contribution

TAR - Topology Aware Ring

Idea: mimimize global link utilization

1. Build a smart ring
2. Send data

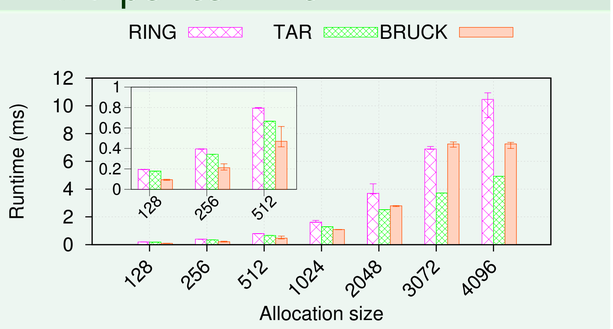


Minimize data transfers on global links

Evaluation

- CODES based on ROSS, an event driven simulator
- 5256 terminals: 73 groups, 12 routers per group, and 6 terminals per router
- Bandwidth:
  * Global links at 4.7 GiB/s,
  * Local links at 5.25 GiB/s,
  * Terminal links at 5.25 GiB/s

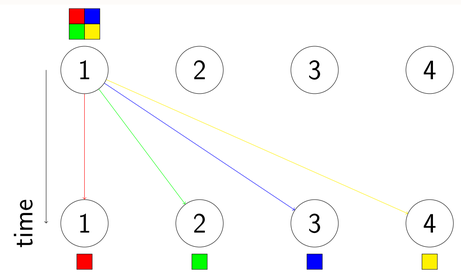
1 KiB per terminal



Conclusion

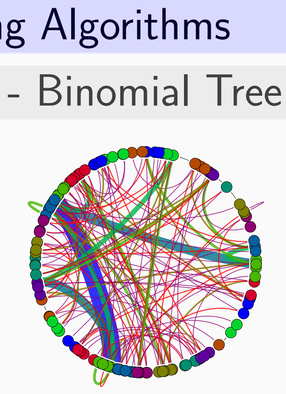
- Topology awareness matters: TAR wins!
- BRUCK still better for small transfers
- Background traffic does not change the behavior

Scatter



Existing Algorithms

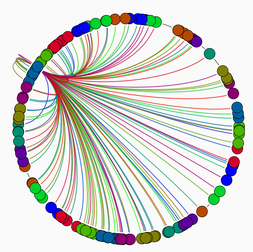
TREE - Binomial Tree



Default in MPICH

Simple approach

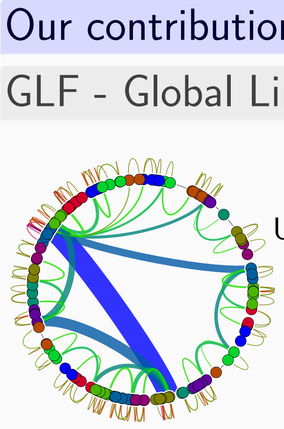
LIN



Send data to terminals one by one
Minimize link utilization

Our contribution

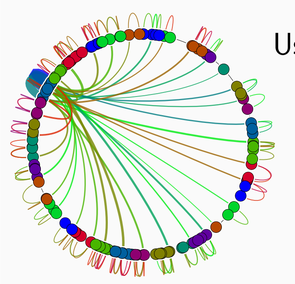
GLF - Global Link First



Using TREE:
1. Send data to groups
2. Send data to routers
3. Send data to terminals

Use global links only during the first phase

LLF - Local Link First

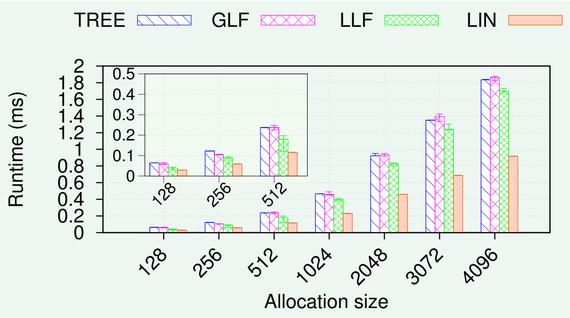


Using TREE:
1. Send data to routers in root group
2. Send data to groups
3. Send data to routers
4. Send data to terminals

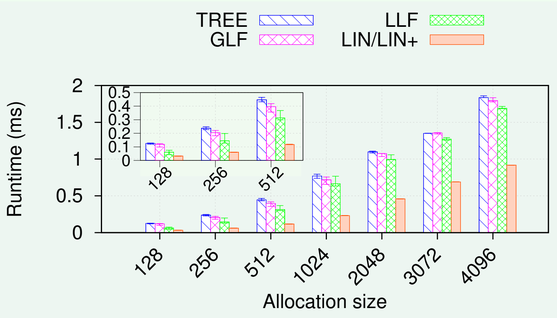
Minimize data transfers on global links

LIN twice faster than the others!

Sending 1 KiB per terminal

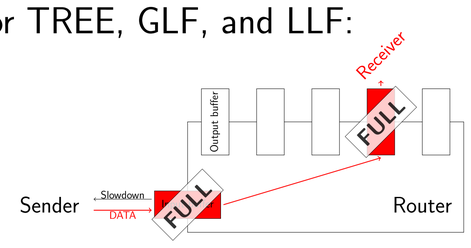


With uniform background traffic



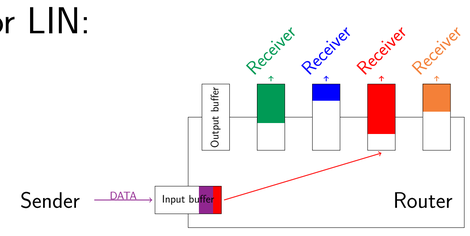
Major impact factor: the buffer size!

For TREE, GLF, and LLF:



- Large amounts of data transferred
- Fills buffers, slows down transfer

For LIN:



- 1 KiB transferred each time
- No buffer saturated, no slowdown

Conclusion

- Topology and hardware characteristics impact the performance of algorithms
- LIN is twice faster than LLF, GLF, and TREE
- No significant differences between LLF, GLF, and TREE

Discussion

Why is it complex to make AllGather hardware-aware?

- A lot more data sent over the network creating more contention
- Overall run time determined by the slowest link due to the ring structure
- Terminals transfer on the same path (TAR, RING), no balance across buffers

Ongoing work

- *Broadcast* as *Scatter* followed by *AllGather*
- Computation and communication algorithms such as *Reduce*
- Energy efficiency of communication algorithms