Swing: Short-cutting Rings for Higher Bandwidth Allreduce

Daniele De Sensi[1], Tommaso Bonato[2], David Saam[3], Torsten Hoefler[2]

2 ETH zürich

1 

3 RWTH AACHEN UNIVERSITY

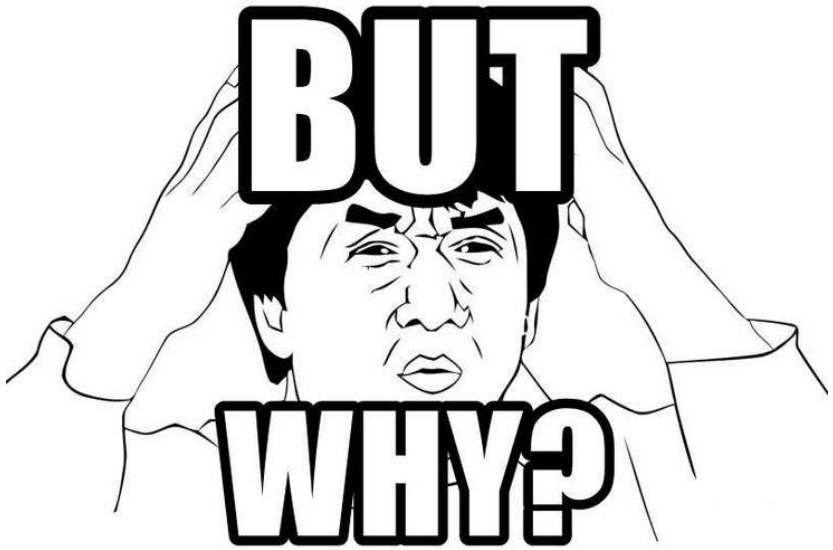
NSDI
Santa Clara – April 16-18, 2024


SAPIENZA
UNIVERSITÀ DI ROMA

Swing

A novel **allreduce/allgather/reduce-scatter** algorithm

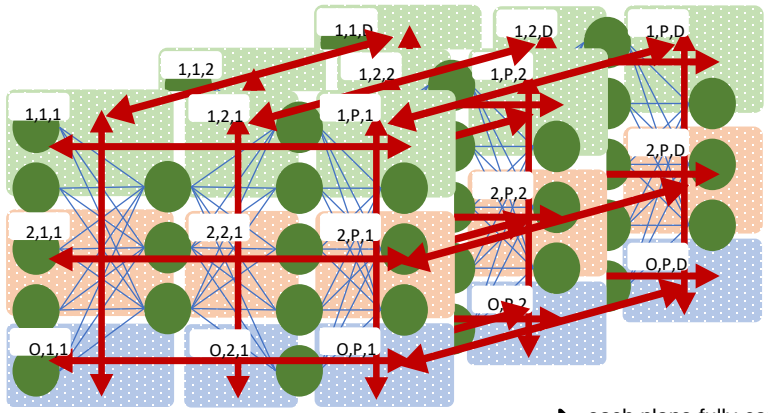
optimized for multi-dimensional **torus networks**

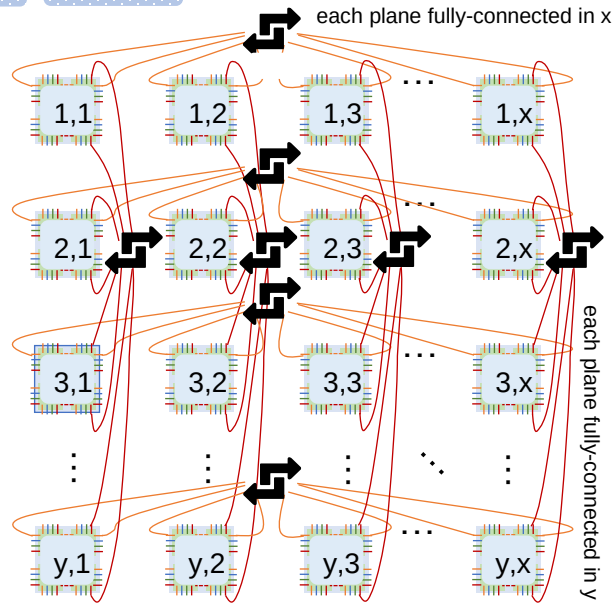


(expected advantages on any blocking network)

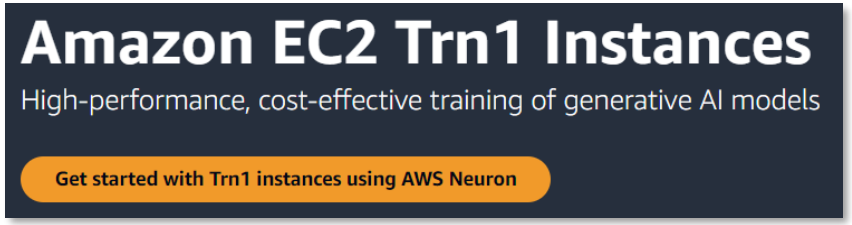
Why torus and why allreduce?

3D - Data, Pipeline, and Operator Parallelism

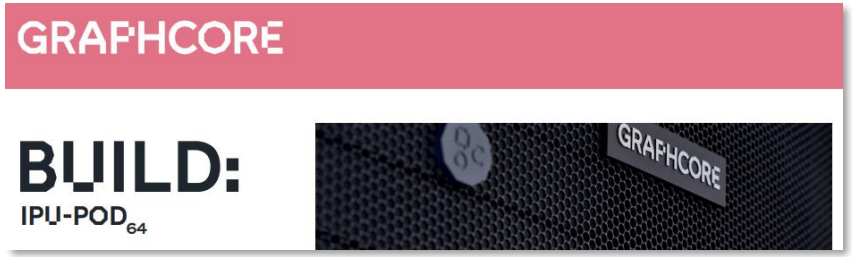




Google's TPU v5p Pod
(> 9,000 chips on a 3D torus)



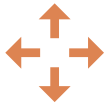
AWS Trainium Instances
(16 chips on a 2D torus)



Graphcore IPU-POD
(64 chips on a 2D torus)

HammingMesh: A Network Topology for Large-Scale Deep Learning (2022)
Torsten Hoefler, Tommaso Bonato, Daniele De Sensi, Salvatore Di Girolamo, Shigang Li, Marco Heddes, Jon Belk, Deepak Goel, Miguel Castro, Steve Scott

Allreduce Algorithms



	Bucket [1]	Recursive Doubling [2]	Swing
	Linear in the number of nodes per dimension	**Logarithmic** in the number of nodes	**Logarithmic** in the number of nodes
	Uses **all** the network ports simultaneously	Uses only **one** network port at a time	Uses **all** the network ports simultaneously
	No flow collisions	**Many** flow collisions	**Few** flow collisions

State-of-the-art

[1] N. Jain and Y. Sabharwal. *Optimal bucket algorithms for large MPI collectives on torus interconnects* (2010)

[2] P. Sack and W. Gropp. *Collective algorithms for multiported torus networks* (2015)

Swing Allreduce

Details, generalization to D dimensions, and correctness proof in the paper

$-1 \bmod p = p-1$, whereas node 1 communicates with node 2. Thus, we can say that at step 1 the data sent from 0 reached nodes $\{1, 2, p-1\}$ (2 has been reached *indirectly* through node 1).

Because the number of reached nodes doubles at each step, and because we perform $\log_2 p$ steps, the data sent from any given node would eventually reach $p-1$ nodes. We need, however, to prove that those $p-1$ nodes are distinct (i.e., that the data sent by each node reach every other node exactly once and is thus never aggregated twice). To do so, we need first to prove a few lemmas.

Lemma A.1. *$\rho(s)$ and $\delta(s)$ are odd $\forall s \in \mathbb{N}$.*

Proof. $(-2)^i$ is odd for $i = 0$, and even for $i > 0$. The sum of even numbers with an odd number is odd. □

Lemma A.2. *If r is even, $\pi(r,s)$ is odd, and vice versa.*

Proof. An even node r communicates at step s with node $\pi(r,s) = r + \rho(s) \bmod p$. Because p is a power of two (thus even), and $\rho(s)$ is odd (Lemma A.1), $\pi(r,s)$ is odd. Vice versa, odd nodes communicate with even nodes. □

If a node r communicates at step s with a node $q = \pi(r,s)$, and q communicates with a node $z = \pi(q,h)$ at step $h > s$, we say that s indirectly reached node z. Because even nodes only communicate with odd nodes (and vice versa), if r is even, we can rewrite:

$$z = \overbrace{\underbrace{(r + \rho(s) \bmod p)}_{q=\pi(r,s)} - \rho(h) \bmod p}^{\pi(q,h)} = r + \rho(s) - \rho(h) \bmod p$$

I.e., the sign behind $\rho(s)$ alternates between positive and negative, starting from positive. In general, an even node r can reach through a sequence of k steps $\{s_0 < s_1 < s_2 < \ldots < s_{k-1}\}$ a node q, with:

$$q = r + \rho(s_0) - \rho(s_1) + \rho(s_2) - \ldots \bmod p =$$
$$= (r + \sum_{i=0}^{k-1} -1^i \rho(s_i)) \bmod p$$

The same applies if r is odd, by replacing -1^i with -1^{i+1}.

Lemma A.3. *Even nodes reach (directly or indirectly) odd nodes through an odd number of steps k. Odd nodes reach (directly or indirectly) even nodes through an odd number of steps k.*

Proof. This stems from Lemma A.2. If r is even and k is odd, then $q = (r + \sum_{i=0}^{k-1} -1^i \rho(s_i)) \bmod p$ is odd because $\rho(s)$ is always odd. Similarly, if r is odd and k is odd, q is even. □

Lemma A.4. *Given k integers $\{e_0 < e_1 < \ldots < e_{k-1}\}$, with $e_{k-1} \leq \log_2(p) - 1$, then $-p < \sum_{i=0}^{k-1} (-2)^{e_i} < p$.*

Proof. We have $\sum_{i=0}^{k-1} (-2)^{e_i} \leq \sum_{i=0}^{k-1} 2^{e_i} < 2^{e_{k-1}+1} \leq p$. Similarly, $\sum_{i=0}^{k-1} (-2)^{e_i} \geq -\sum_{i=0}^{k-1} 2^{e_i} > -(2^{e_{k-1}+1}) \geq -p$. □

Theorem A.5. *On a 1D torus, if a node r at step s communicates with node $\pi(r,s)$ (defined in Eq. 2), it will reach (directly or indirectly) all the other $p-1$ nodes in $\log_2(p)$ steps (with p power of two).*

Proof. We need to prove that, a unique sequence of k steps $\{s_0 < s_1 < \ldots s_{k-1}\}$ exists by which a given node r reaches a node q. We prove this by contradiction, and we will prove it by assuming that r is even and q is odd (the proof for the other cases is analogous and only requires changing the signs before the ρ terms). Assume that there are two different sequences of steps $\{s_0 < s_1 < \ldots s_{k-1} \leq \log_2(p) - 1\}$ and $\{t_0 < t_1 < \ldots t_{h-1} \leq \log_2(p) - 1\}$ of k and h steps respectively (both k and h are odd from Lemma A.3), so that:

$$q = r + \rho(s_0) - \rho(s_1) + \rho(s_2) - \ldots + \rho(s_{k-1}) \bmod p =$$
$$= r + \rho(t_0) - \rho(t_1) + \rho(t_2) - \ldots + \rho(t_{h-1}) \bmod p \quad (4)$$

By expanding the first of the two sequences we have:

$$q = r + \sum_{i=0}^{s_0} (-2)^i - \sum_{i=0}^{s_1} (-2)^i + \ldots + \sum_{i=0}^{s_{k-1}} (-2)^i \bmod p$$
$$= r + \sum_{i=0}^{s_0} (-2)^i + \sum_{i=s_1+1}^{s_2} (-2)^i + \ldots + \sum_{i=s_{k-2}+1}^{s_{k-1}} (-2)^i \bmod p$$

By expanding similarly the second assignment in Eq. 4, we have that the two sequences exist if:

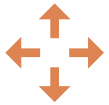
$$\sum_{i=0}^{s_0} (-2)^i + \ldots + \sum_{i=s_{k-2}+1}^{s_{k-1}} (-2)^i \equiv$$
$$\equiv \sum_{i=0}^{t_0} (-2)^i + \ldots + \sum_{i=t_{k-2}+1}^{t_{h-1}} (-2)^i \pmod p \quad (5)$$

From Lemma A.4, we know that both sides are in the range $(-p, p)$. Thus, the two sides are congruent only if: i) they have the same sign and are equal, or; ii) they have different signs, and by summing p on the negative side, we get the positive side. Since each side is the sum of distinct powers of -2, case i) is only possible if the two sequences of steps are equal. To prove that case ii) is impossible, let us consider the case where the left side is negative (the other case is analogous). Because $p = 2^a$ for some $a \in \mathbb{N}$, and because $2^a = (-2)^a$ (if a is even[2]), Eq. 5 becomes:

$$\sum_{i=0}^{s_0} (-2)^i + \ldots + \sum_{i=s_{k-2}+1}^{s_{k-1}} (-2)^i + (-2)^a =$$
$$= \sum_{i=0}^{t_0} (-2)^i + \ldots + \sum_{i=t_{k-2}+1}^{t_{h-1}} (-2)^i$$

[2] If a is odd, $p = 2^a = (-2)^{a+1} + (-2)^a$, and the same considerations still hold.

Allreduce Algorithms



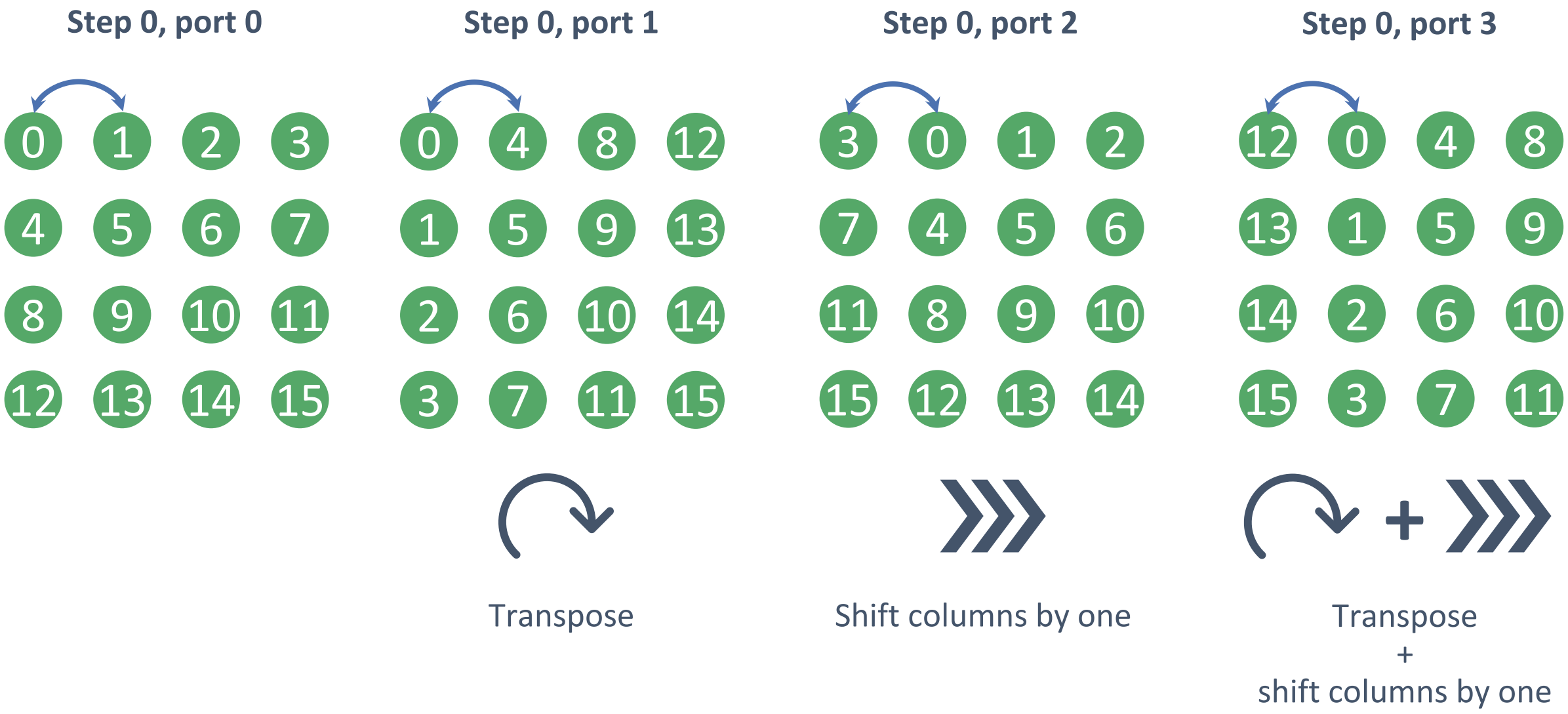
	Bucket [1]	Recursive Doubling [2]	Swing
	Linear in the number of nodes per dimension	**Logarithmic** in the number of nodes	**Logarithmic** in the number of nodes
	Uses **all** the network ports simultaneously	Uses only **one** network port at a time	Uses **all** the network ports simultaneously
	No flow collisions	**Many** flow collisions	**Few** flow collisions

State-of-the-art

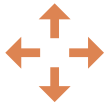
[1] N. Jain and Y. Sabharwal. *Optimal bucket algorithms for large MPI collectives on torus interconnects* (2010)

[2] P. Sack and W. Gropp. *Collective algorithms for multiported torus networks* (2015)

Multiport Swing



Allreduce Algorithms

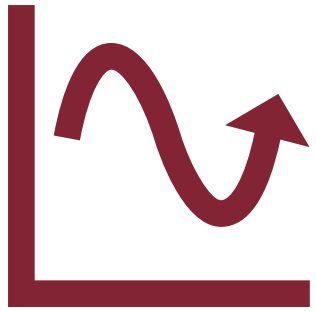


	Bucket [1]	Recursive Doubling [2]	Swing
	Linear in the number of nodes per dimension	**Logarithmic** in the number of nodes	**Logarithmic** in the number of nodes
	Uses **all** the network ports simultaneously	Uses only **one** network port at a time	Uses **all** the network ports simultaneously
	No flow collisions	**Many** flow collisions	**Few** flow collisions

State-of-the-art

[1] N. Jain and Y. Sabharwal. *Optimal bucket algorithms for large MPI collectives on torus interconnects* (2010)

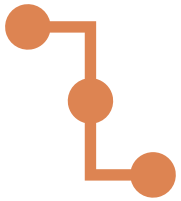
[2] P. Sack and W. Gropp. *Collective algorithms for multiported torus networks* (2015)



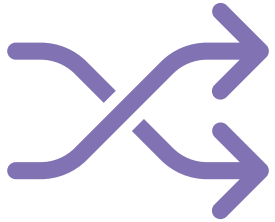
Evaluation

Setup

SST packet-level
network simulator

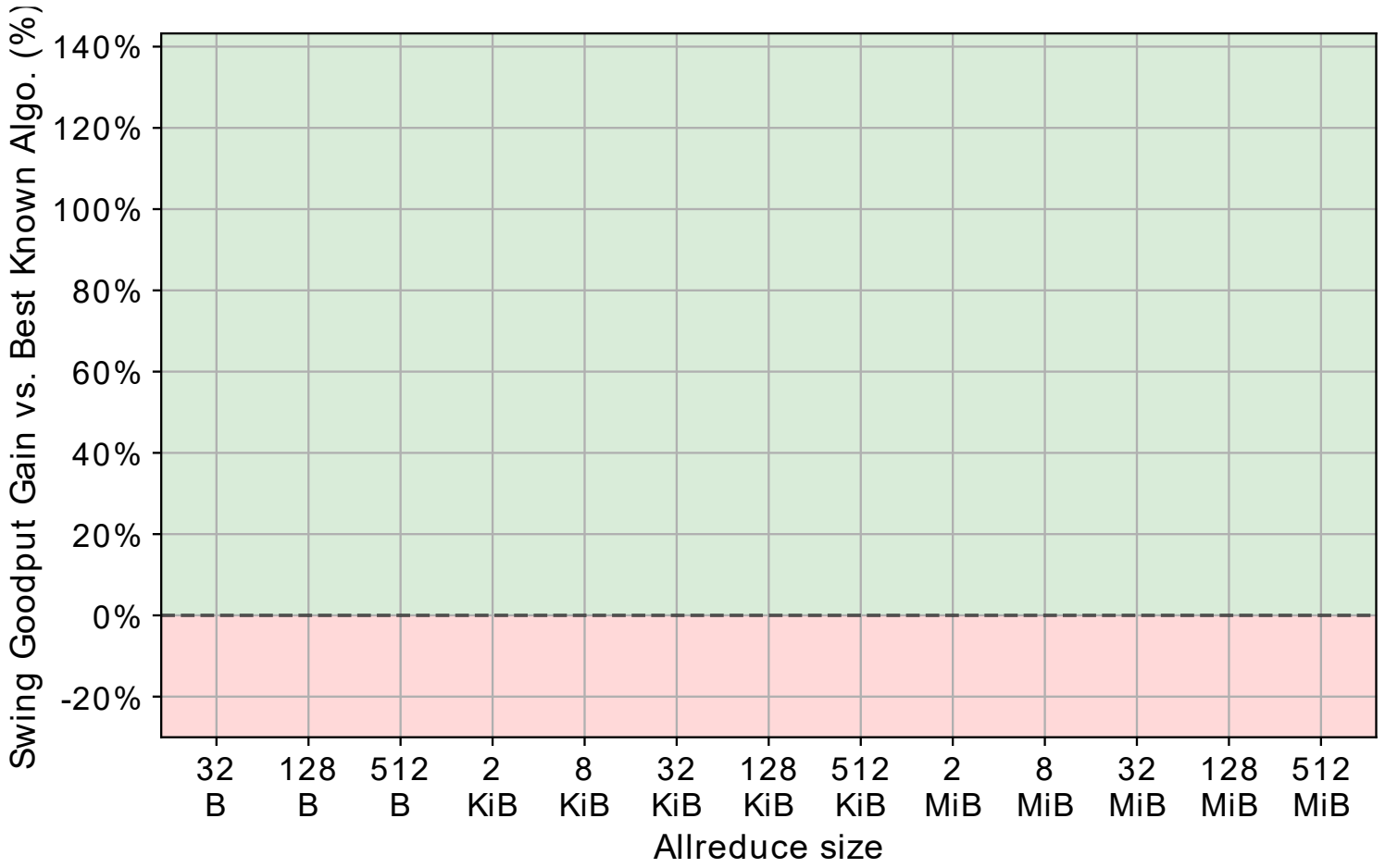


400 Gb/s links
100 ns latency

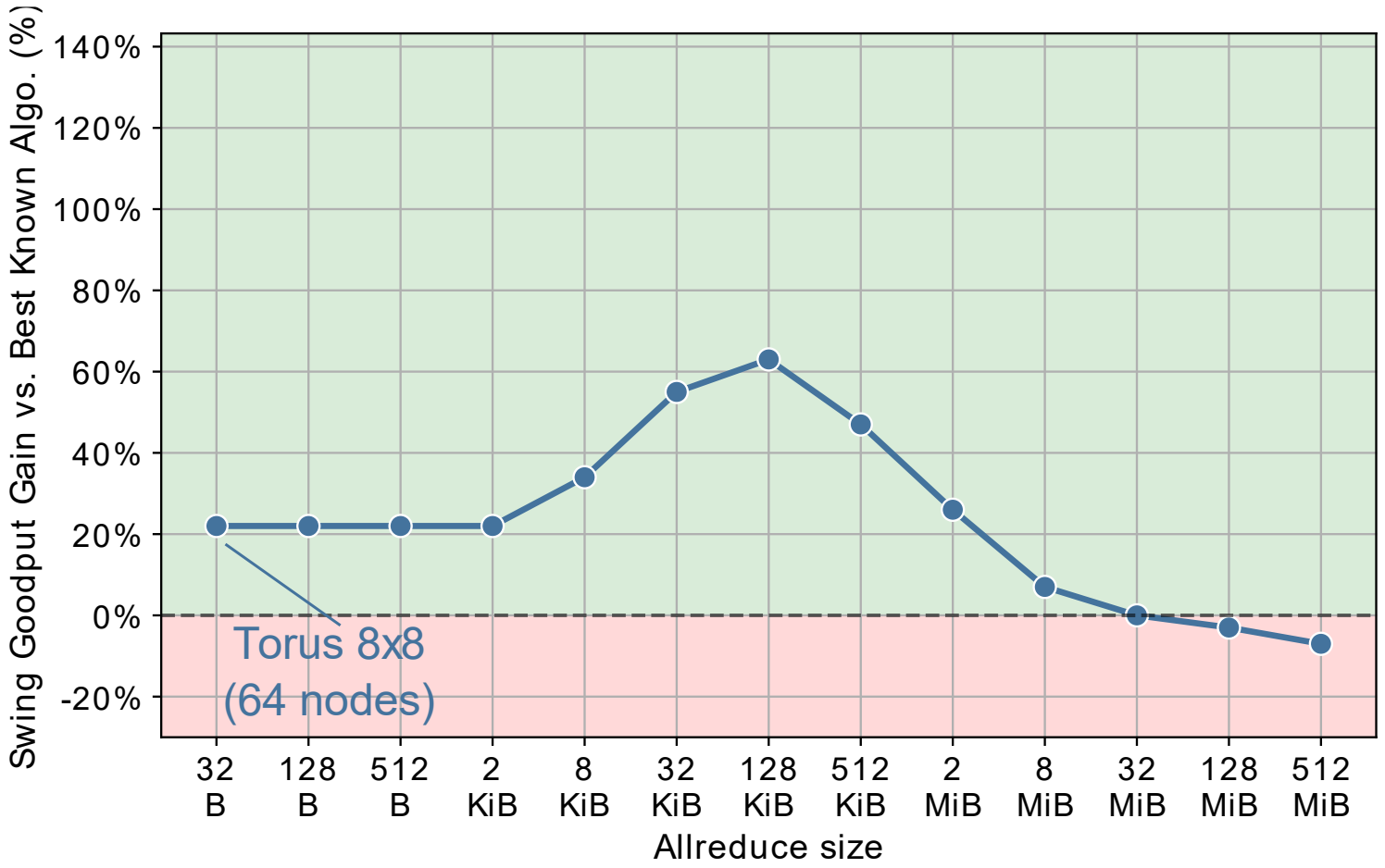


300 ns
per-hop latency

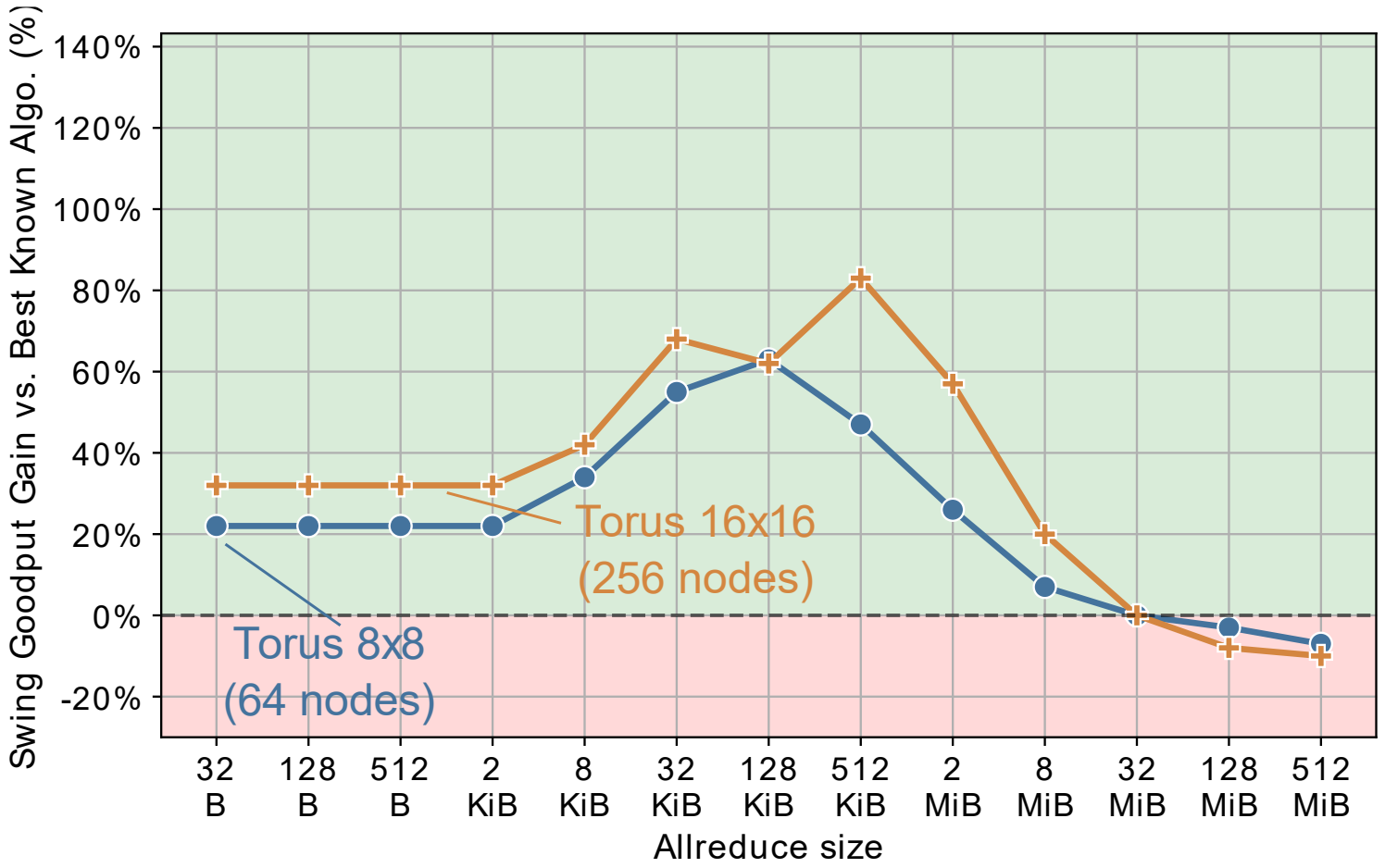
Results - Performance Gain



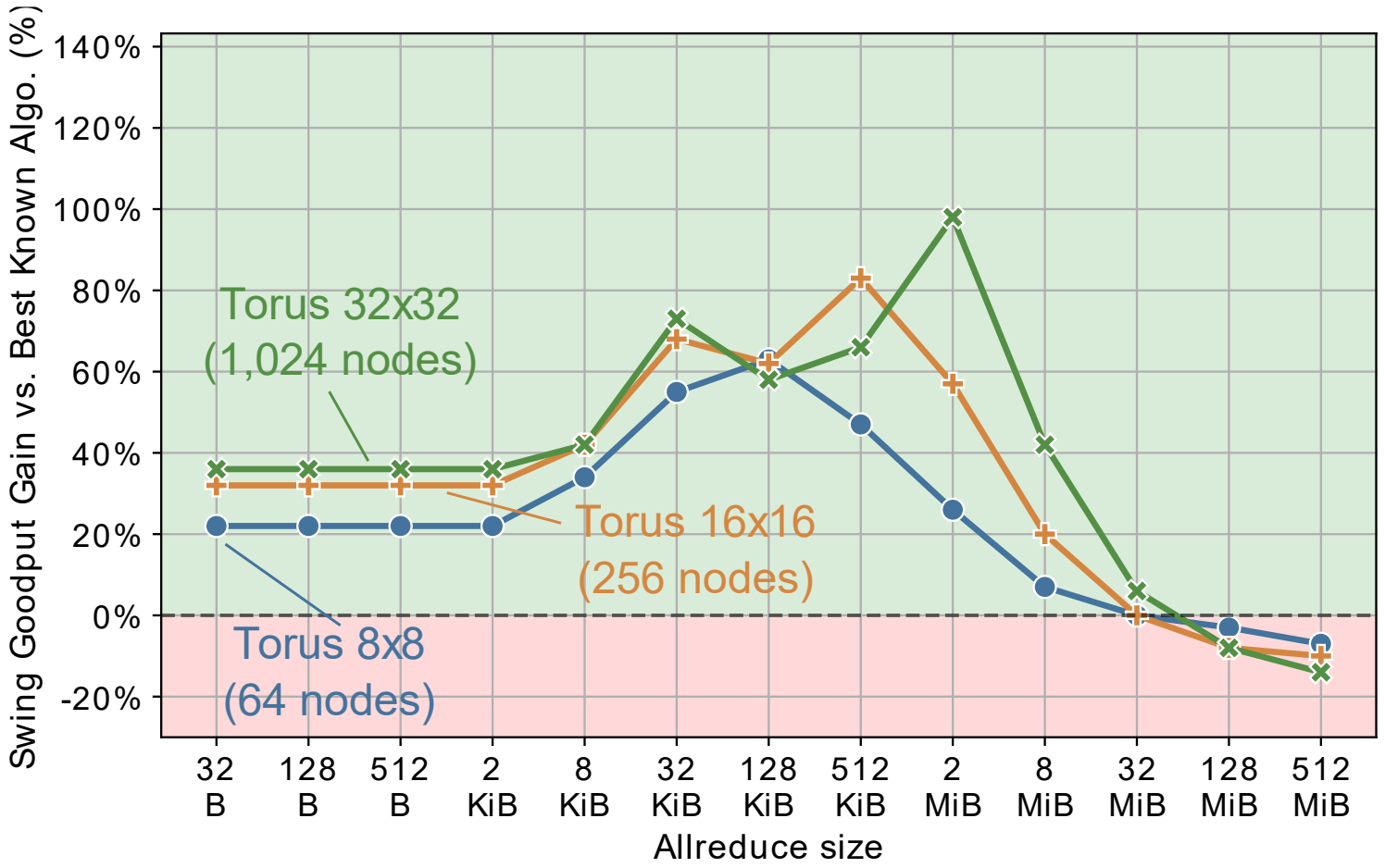
Results - Performance Gain



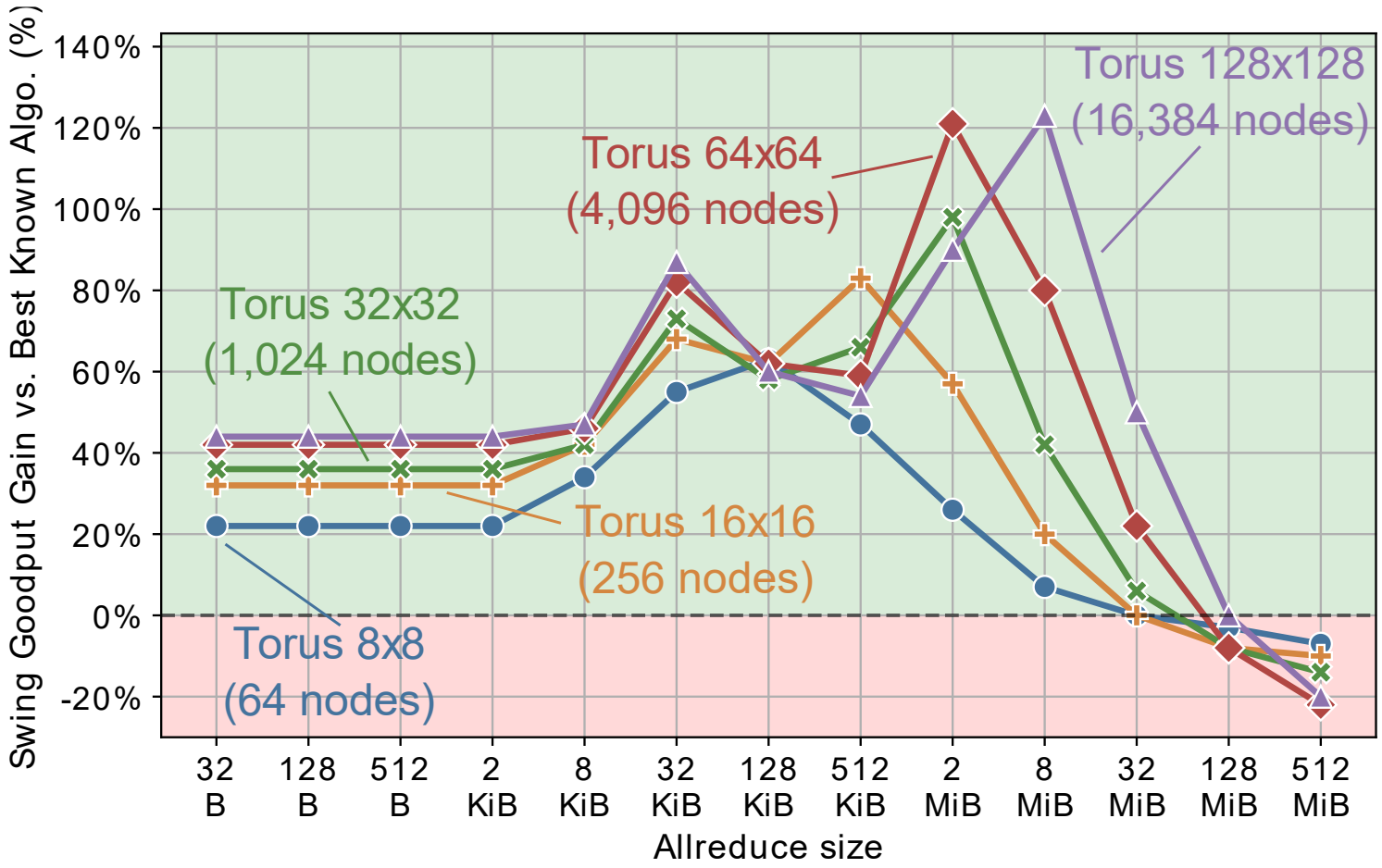
Results - Performance Gain



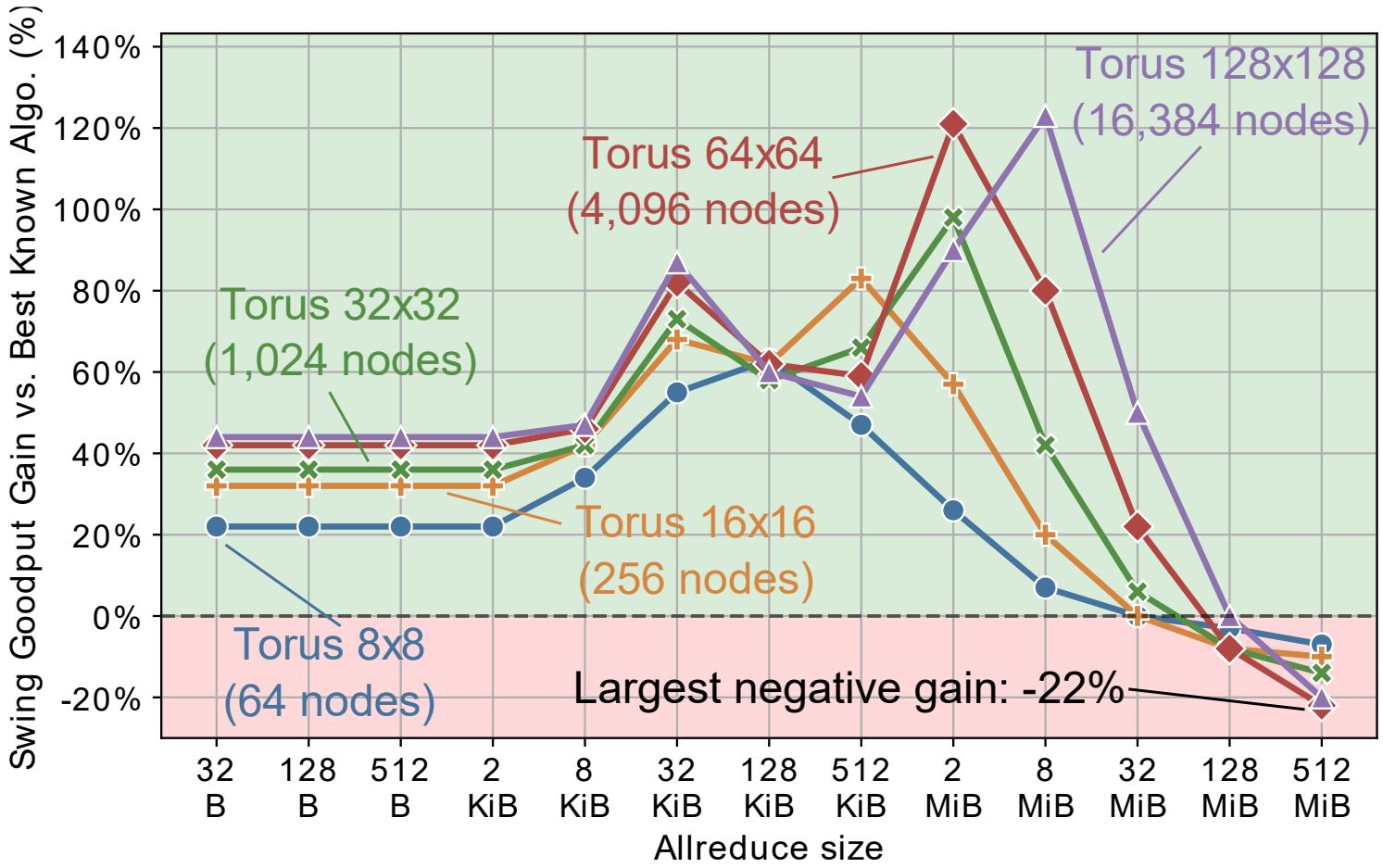
Results - Performance Gain



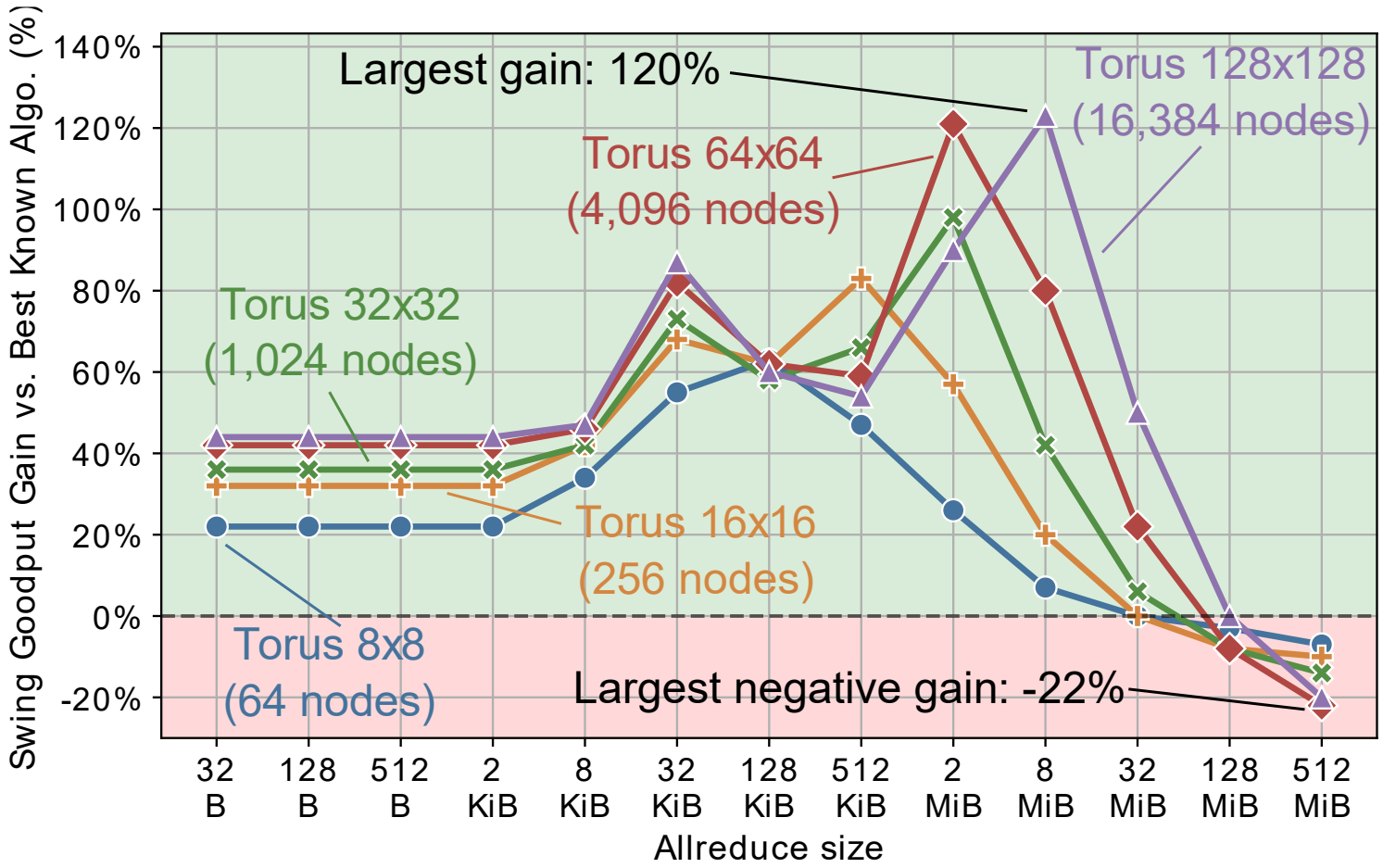
Results - Performance Gain



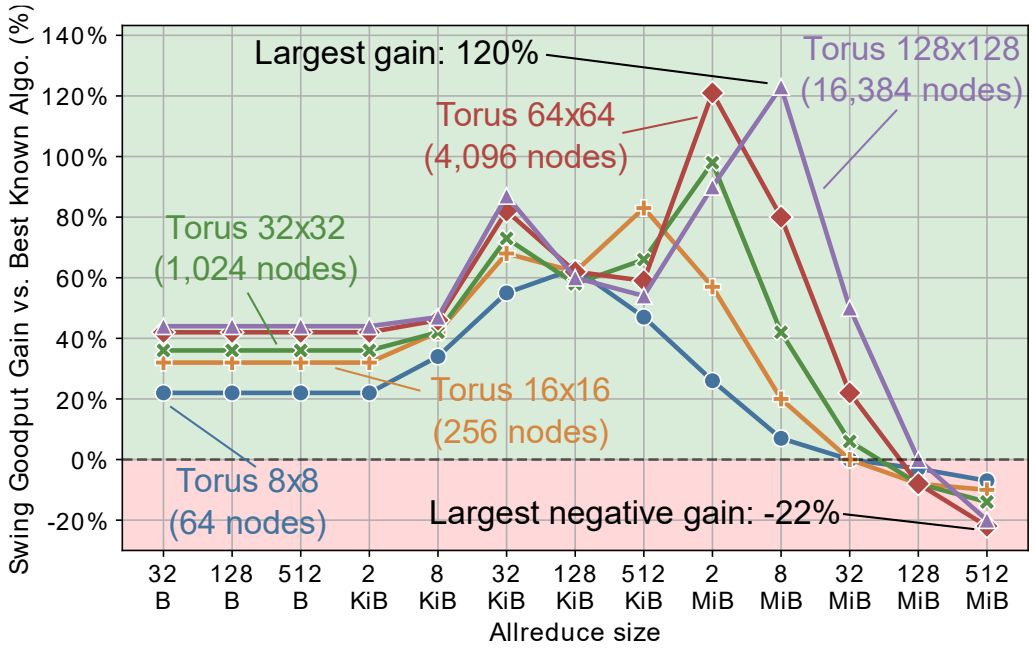
Results - Performance Gain

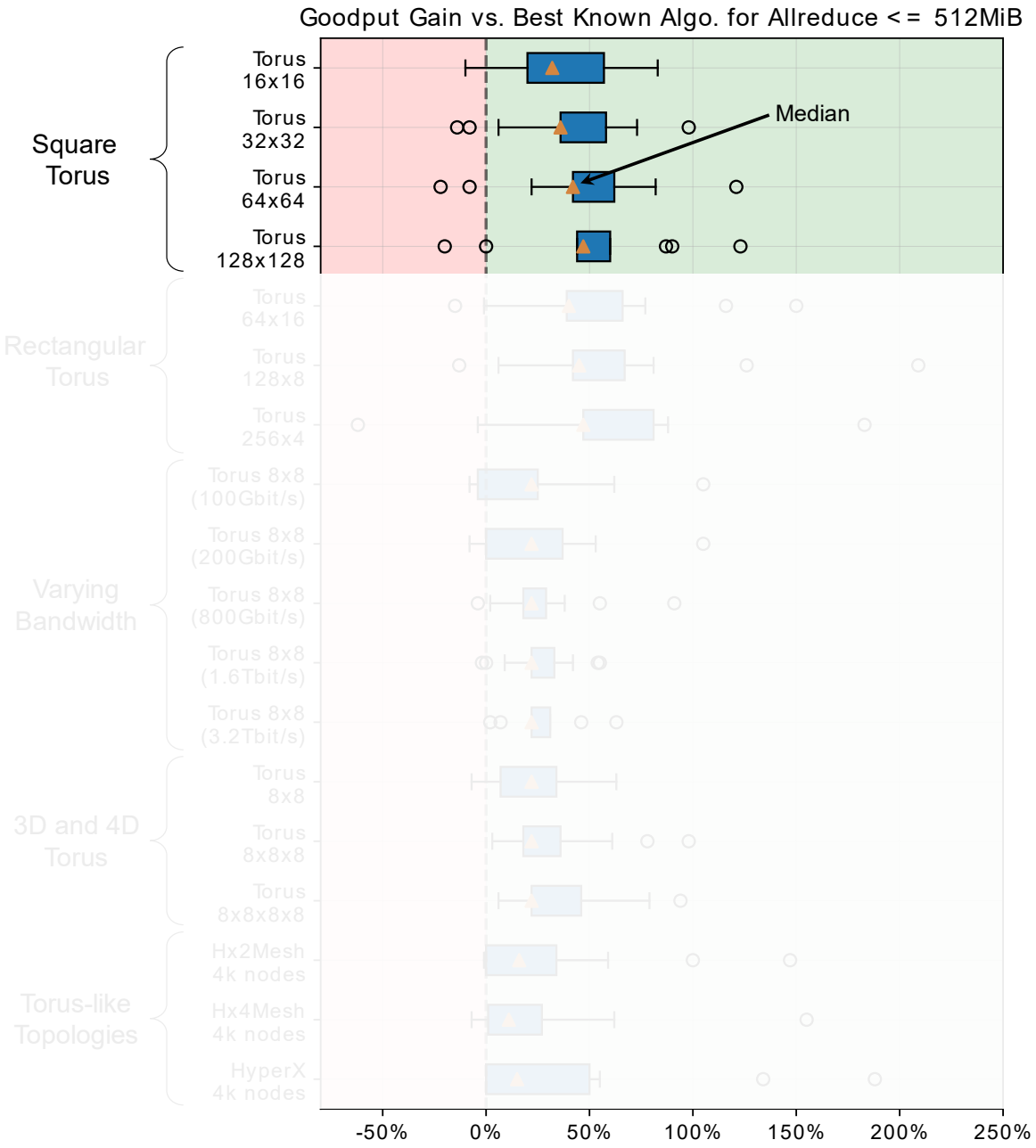


Results - Performance Gain



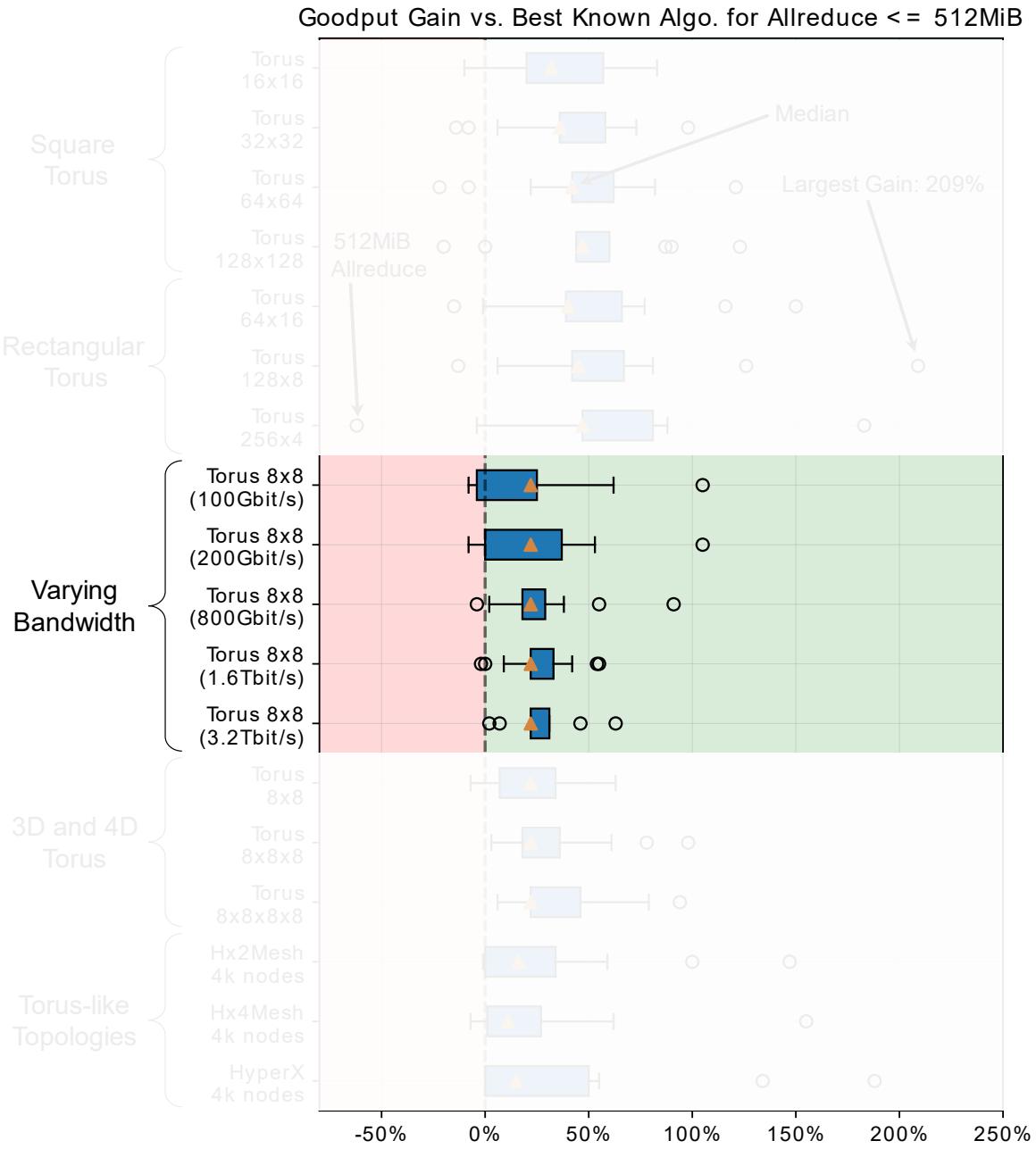
Results Summary



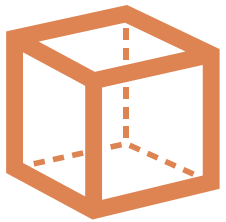


Results Summary

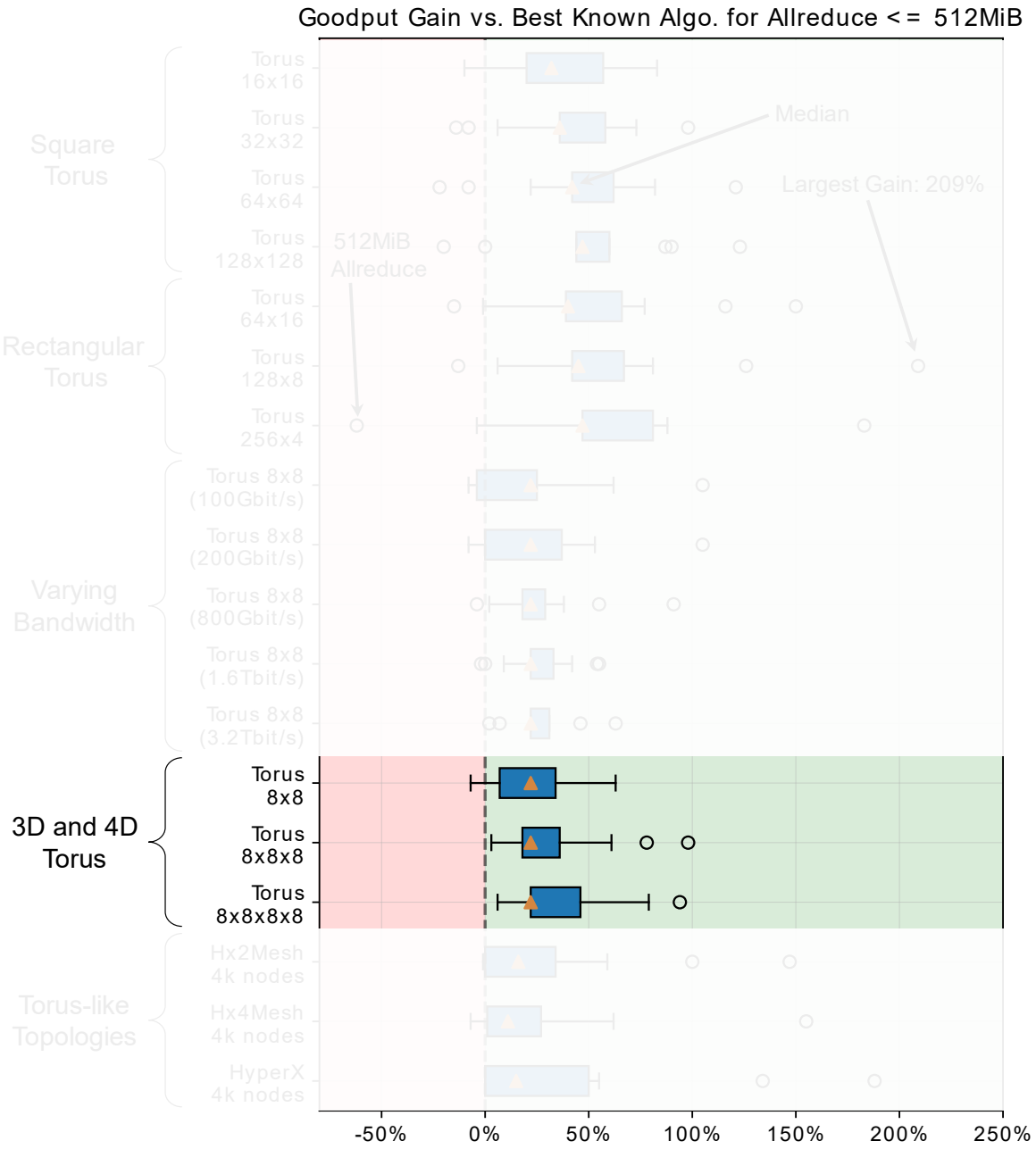
At higher bandwidth, the number of steps has a higher relative impact on performance



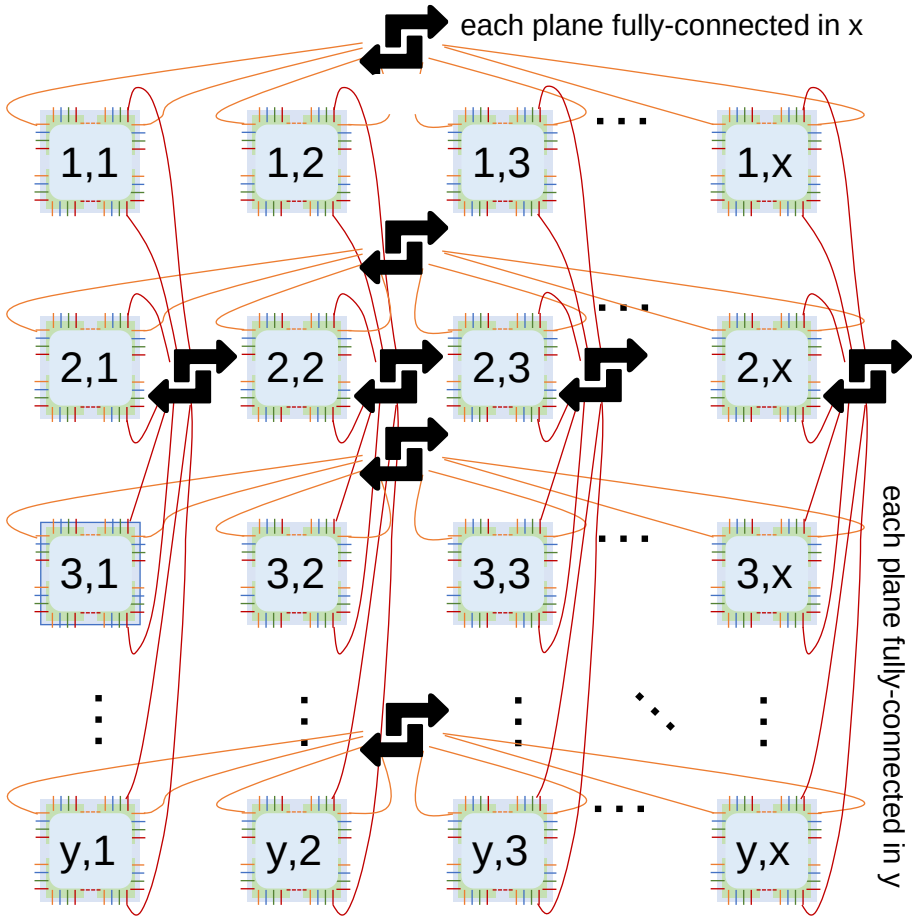
Results Summary

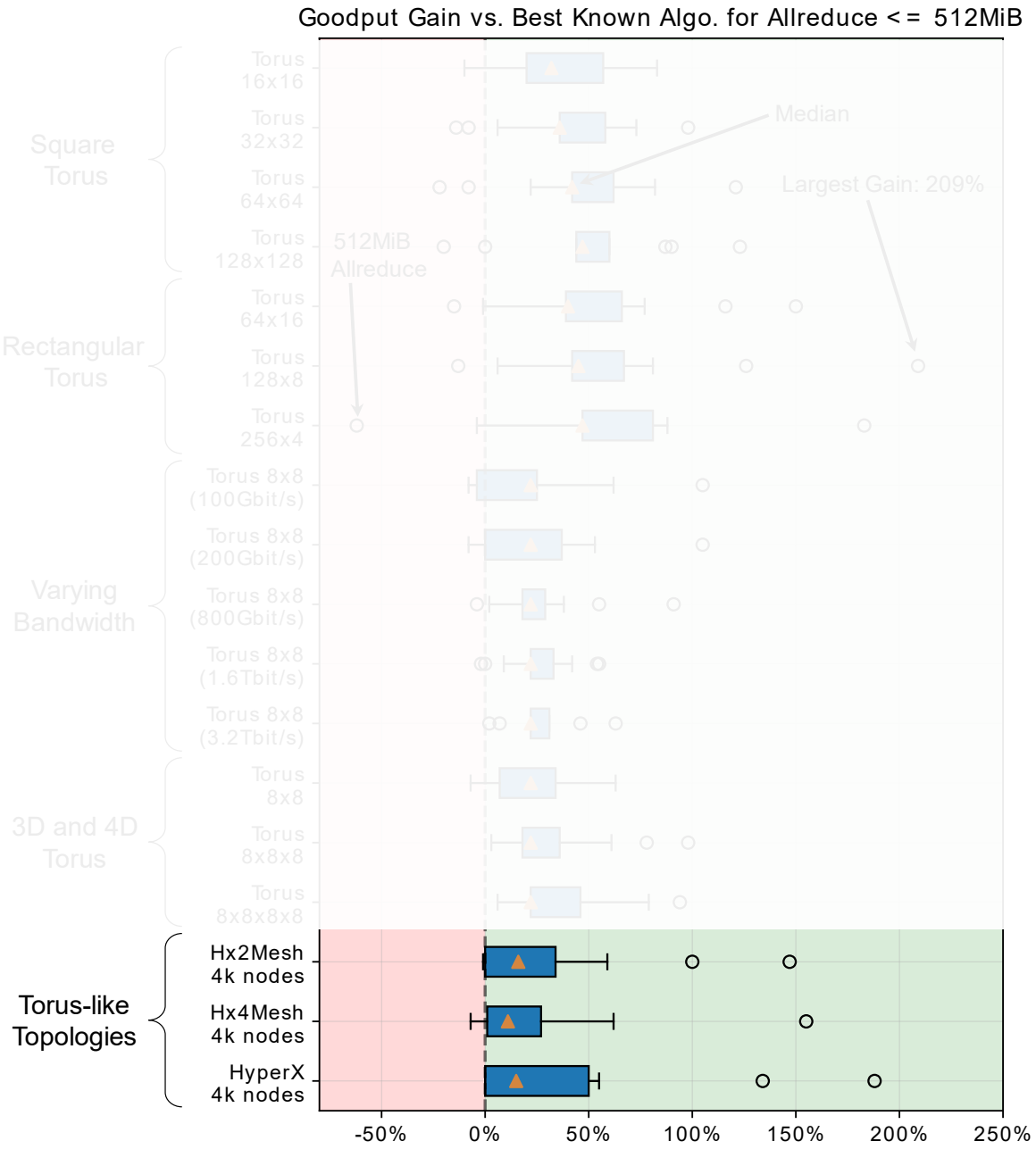


More dimensions imply more communications with close nodes



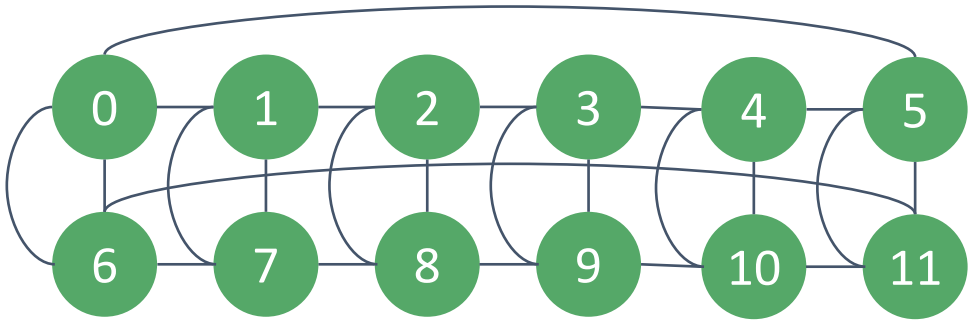
Results Summary

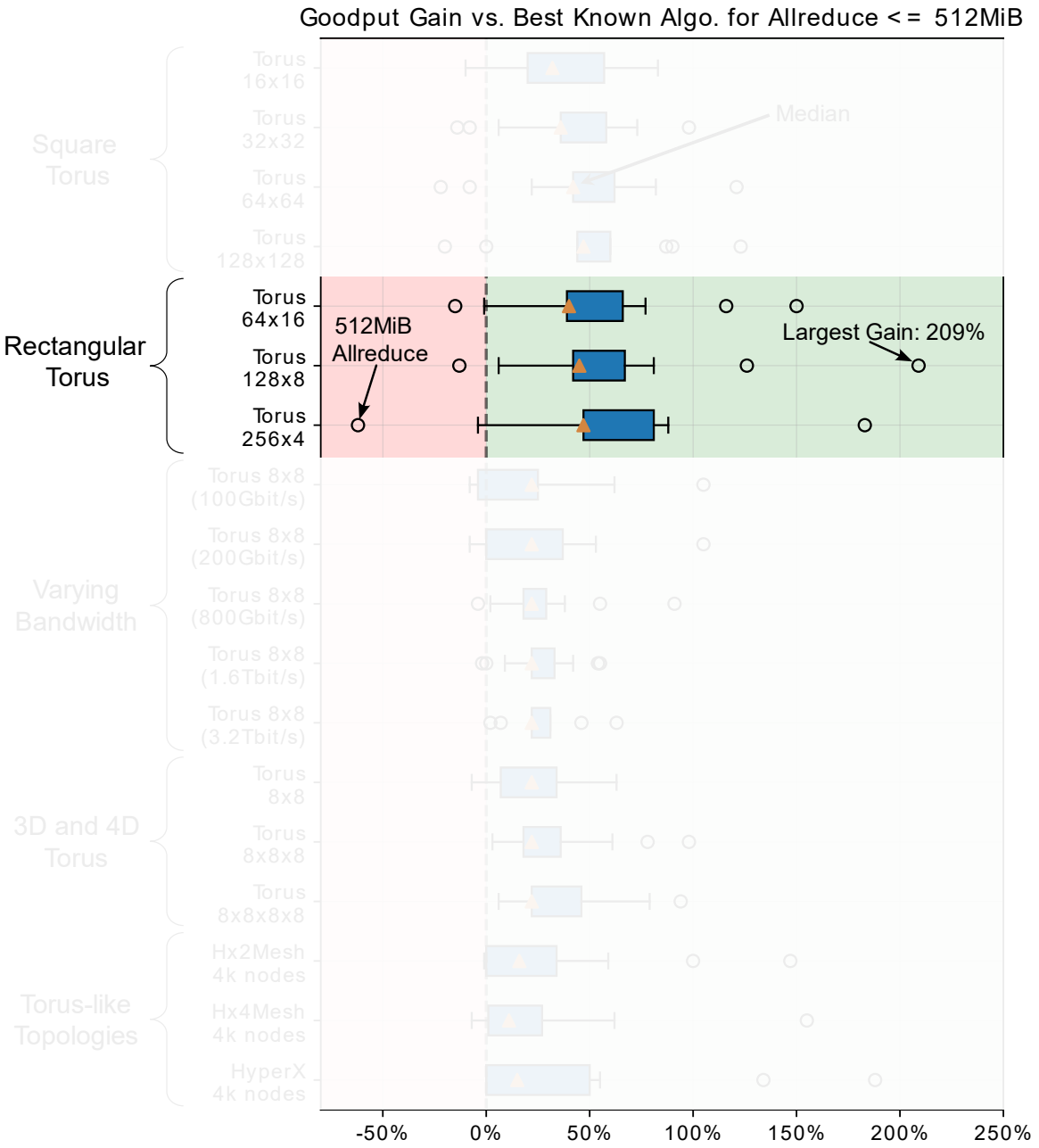




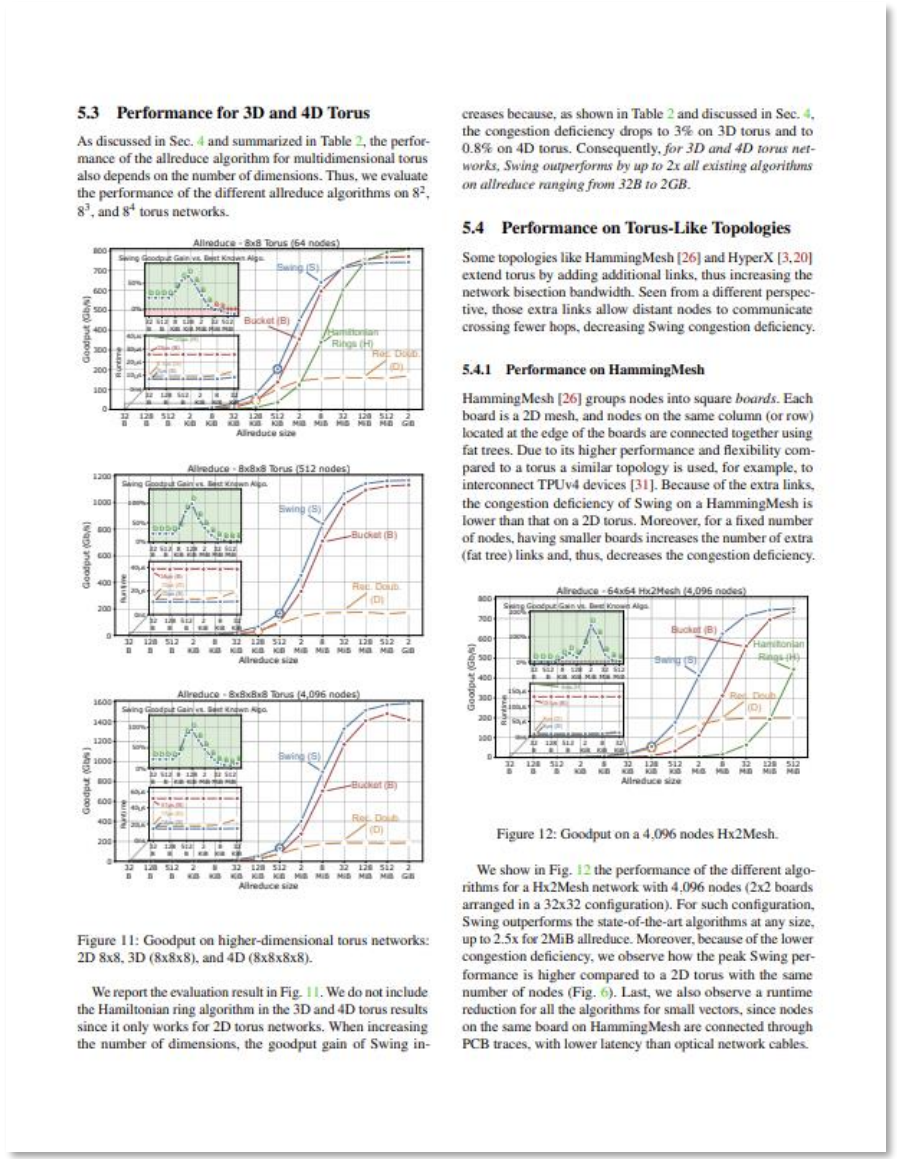
HammingMesh: A Network Topology for Large-Scale Deep Learning (2022)
Torsten Hoefler, Tommaso Bonato, Daniele De Sensi, Salvatore Di Girolamo, Shigang Li, Marco Heddes, Jon Belk, Deepak Goel, Miguel Castro, Steve Scott

Results Summary





Results Summary



5.3 Performance for 3D and 4D Torus

As discussed in Sec. 4 and summarized in Table 2, the performance of the allreduce algorithm for multidimensional torus also depends on the number of dimensions. Thus, we evaluate the performance of the different allreduce algorithms on 8^2, 8^3, and 8^4 torus networks.

Allreduce - 8x8 Torus (64 nodes)

Allreduce - 8x8x8 Torus (512 nodes)

Allreduce - 8x8x8x8 Torus (4,096 nodes)

Figure 11: Goodput on higher-dimensional torus networks: 2D 8x8, 3D (8x8x8), and 4D (8x8x8x8).

We report the evaluation result in Fig. 11. We do not include the Hamiltonian ring algorithm in the 3D and 4D torus results since it only works for 2D torus networks. When increasing the number of dimensions, the goodput gain of Swing increases because, as shown in Table 2 and discussed in Sec. 4, the congestion deficiency drops to 3% on 3D torus and to 0.8% on 4D torus. Consequently, *for 3D and 4D torus networks, Swing outperforms by up to 2x all existing algorithms on allreduce ranging from 32B to 2GB.*

5.4 Performance on Torus-Like Topologies

Some topologies like HammingMesh [26] and HyperX [3,20] extend torus by adding additional links, thus increasing the network bisection bandwidth. Seen from a different perspective, those extra links allow distant nodes to communicate crossing fewer hops, decreasing Swing congestion deficiency.

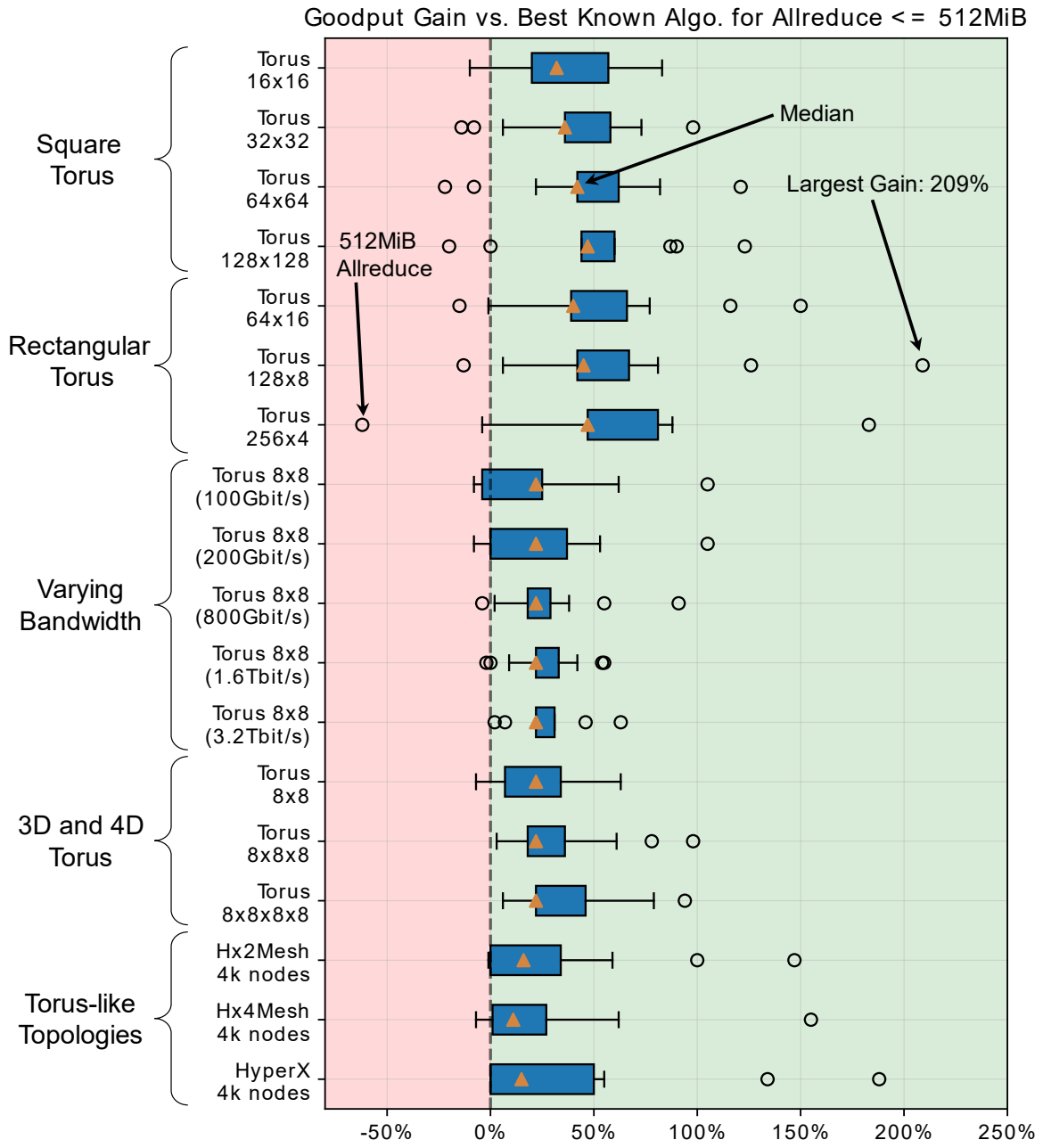
5.4.1 Performance on HammingMesh

HammingMesh [26] groups nodes into square *boards*. Each board is a 2D mesh, and nodes on the same column (or row) located at the edge of the boards are connected together using fat trees. Due to its higher performance and flexibility compared to a torus a similar topology is used, for example, to interconnect TPUv4 devices [31]. Because of the extra links, the congestion deficiency of Swing on a HammingMesh is lower than that on a 2D torus. Moreover, for a fixed number of nodes, having smaller boards increases the number of extra (fat tree) links and, thus, decreases the congestion deficiency.

Allreduce - 64x64 Hx2Mesh (4,096 nodes)

Figure 12: Goodput on a 4,096 nodes Hx2Mesh.

We show in Fig. 12 the performance of the different algorithms for a Hx2Mesh network with 4,096 nodes (2x2 boards arranged in a 32x32 configuration). For such configuration, Swing outperforms the state-of-the-art algorithms at any size, up to 2.5x for 2MiB allreduce. Moreover, because of the lower congestion deficiency, we observe how the peak Swing performance is higher compared to a 2D torus with the same number of nodes (Fig. 6). Last, we also observe a runtime reduction for all the algorithms for small vectors, since nodes on the same board on HammingMesh are connected through PCB traces, with lower latency than optical network cables.



Conclusions