# Efficient Direct-Connect Topologies for Collective Communications

**Liangyu Zhao**[1] Siddharth Pal[2] Tapan Chugh[1] Weiyang Wang[3] Jason Fantl[2]
Prithwish Basu[2] Joud Khoury[2] Arvind Krishnamurthy[1]

[1]*University of Washington* [2]*RTX BBN Technologies* [3]*MIT CSAIL*

22nd USENIX Symposium on Networked Systems Design and Implementation

# Collective Communication

- **Collective Communication**: a set of communication operations among parallel computing nodes.
  - e.g., allgather, reduce-scatter, allreduce, all-to-all, etc.

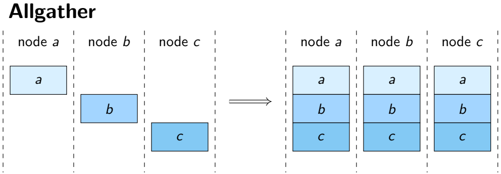



Figure: Allgather Operation

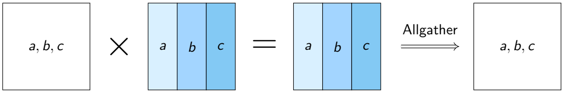



Figure: Distributed Matrix Multiplication

- **Collective Communication**: a set of communication operations among parallel computing nodes.
    - e.g., allgather, reduce-scatter, allreduce, all-to-all, etc.
- **AI/ML Workloads:** Originating in HPC, collective communication is now performance-critical for distributed ML training and inferencing.

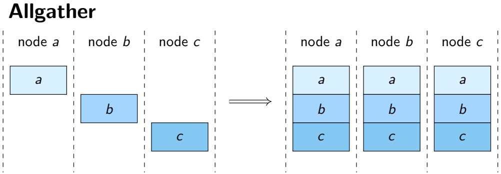



Figure: Allgather Operation

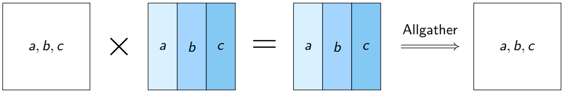



Figure: Distributed Matrix Multiplication

# Collective Communication

- **Collective Communication**: a set of communication operations among parallel computing nodes.
  - e.g., allgather, reduce-scatter, allreduce, all-to-all, etc.
- **AI/ML Workloads:** Originating in HPC, collective communication is now performance-critical for distributed ML training and inferencing.
- **Problem:** As ML models grow larger, scaling AI infra networks in both size and speed is technically challenging and expensive.

**Allgather**

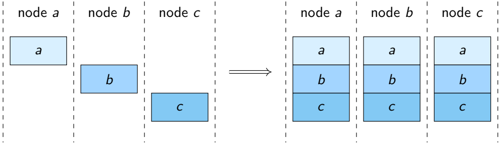



Figure: Allgather Operation

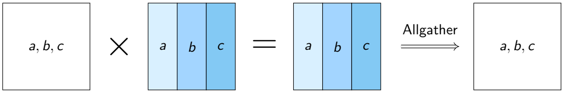
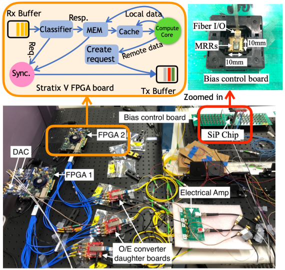



Figure: Distributed Matrix Multiplication

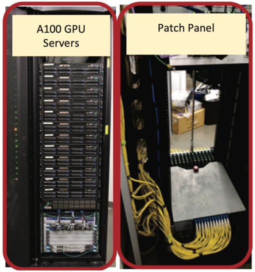
# Optical Circuit Network

An emerging approach is to use **optical circuit networks**:

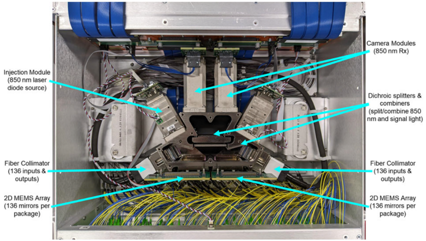
- **Advantages:** Higher ⇑ bandwidth at lower ⇓ capital expenditure and energy cost.
- **Reconfigurability:** The network can be configured into any node-to-node direct-connect topology.
- **Disadvantages:** High reconfiguration latency, requiring relatively fixed topologies in tasks.



(a) SiP-ML (SIGCOMM '21)



(b) TopoOpt (NSDI '23)



(c) TPU (Google)

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce** Latency-Sensitive | **Large-Data Allreduce** Throughput-Sensitive | **All-to-All** Throughput |
|---|---|---|---|
| **Traditional Topologies** (e.g., ring, multi-ring, torus) | | | |
| | | | |

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce** Latency-Sensitive | **Large-Data Allreduce** Throughput-Sensitive | **All-to-All** Throughput |
|---|---|---|---|
| **Traditional Topologies** (e.g., ring, multi-ring, torus) | ✗ | ✓ | ✗ |
| | | | |

- Traditional topologies rely on variants of **ring allreduce**. They offer high allreduce throughput, but their **high diameter** makes low-latency allreduce and efficient all-to-all theoretically impossible.

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce** Latency-Sensitive | **Large-Data Allreduce** Throughput-Sensitive | **All-to-All** Throughput |
|---|---|---|---|
| **Traditional Topologies** (e.g., ring, multi-ring, torus) | ✗ | ✓ | ✗ |
| | | | |

- Traditional topologies rely on variants of **ring allreduce**. They offer high allreduce throughput, but their **high diameter** makes low-latency allreduce and efficient all-to-all theoretically impossible.
- ✘: A single task may involve multiple types of workloads.
  - e.g., MoE training requires both large-data allreduce and all-to-all.

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce**<br>Latency-Sensitive | **Large-Data Allreduce**<br>Throughput-Sensitive | **All-to-All**<br>Throughput |
|---|---|---|---|
| **Traditional Topologies**<br>(e.g., ring, multi-ring, torus) | ✗ | ✓ | ✗ |
| **Low-Diameter Graphs**<br>(e.g., expander graphs) | | | |

# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce** Latency-Sensitive | **Large-Data Allreduce** Throughput-Sensitive | **All-to-All** Throughput |
|---|---|---|---|
| **Traditional Topologies** (e.g., ring, multi-ring, torus) | ✗ | ✓ | ✗ |
| **Low-Diameter Graphs** (e.g., expander graphs) | ✓ | **???** | ✓ |

- Low-diameter graphs enable high all-to-all throughput and low-latency allreduce, but **high-throughput allreduce scheduling** for them remains unknown.

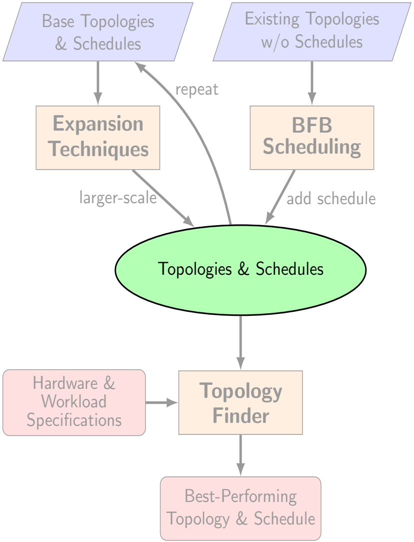
# Topology Dilemma

**Problem:** What direct-connect topology should we choose for a given task?

| | **Small-Data Allreduce** Latency-Sensitive | **Large-Data Allreduce** Throughput-Sensitive | **All-to-All** Throughput |
|---|---|---|---|
| **Traditional Topologies** (e.g., ring, multi-ring, torus) | × | ✓ | × |
| **Low-Diameter Graphs** (e.g., expander graphs) | ✓ | **???** | ✓ |

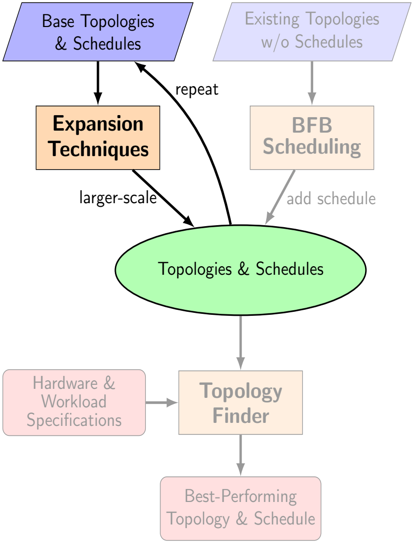
- **Conclusion:** Traditional topologies are theoretically limited. Low-diameter graphs are promising but lack high-throughput allreduce.

# Overview




**Contribution:** a suite of low-diameter topologies with high-throughput allreduce schedules.

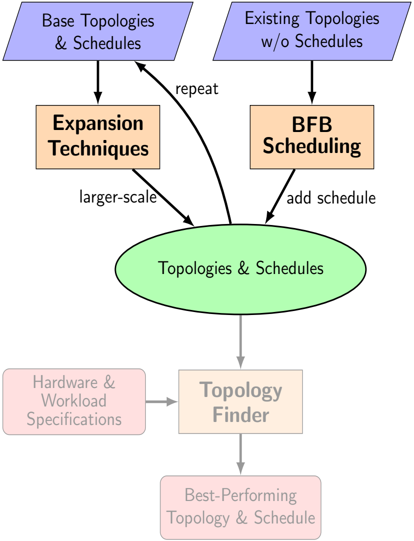
# Overview




**Contribution:** a suite of low-diameter topologies with high-throughput allreduce schedules.

- **Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

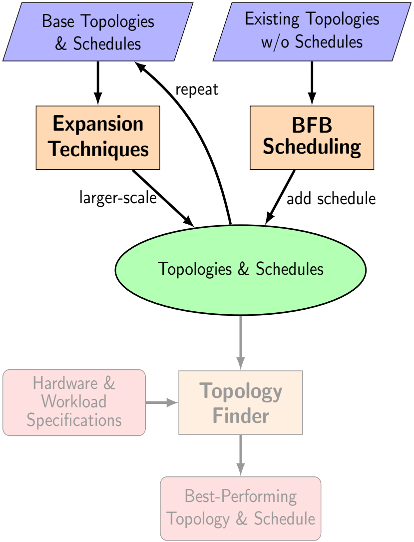
# Overview




**Contribution:** a suite of low-diameter topologies with high-throughput allreduce schedules.

- **Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.
- **BFB Scheduling:** Generate high-throughput allreduce schedules for existing topologies in polynomial time.

# Overview

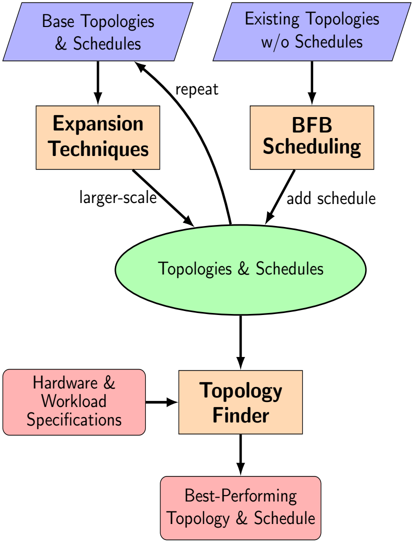



**Contribution:** a suite of low-diameter topologies with high-throughput allreduce schedules.

- **Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.
- **BFB Scheduling:** Generate high-throughput allreduce schedules for existing topologies in polynomial time.

The generated topologies & schedules form a **Pareto-frontier** of low-diameter vs high-throughput allreduce.

# Overview



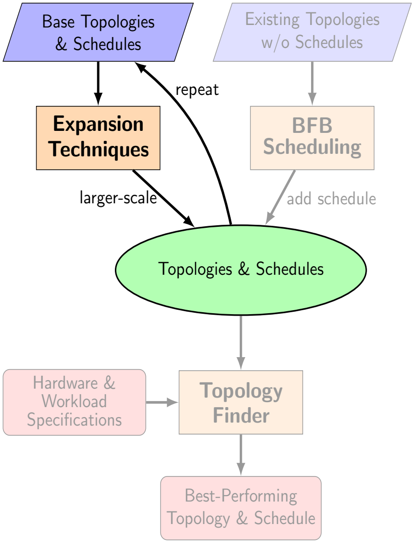

**Contribution:** a suite of low-diameter topologies with high-throughput allreduce schedules.

- **Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.
- **BFB Scheduling:** Generate high-throughput allreduce schedules for existing topologies in polynomial time.

The generated topologies & schedules form a **Pareto-frontier** of low-diameter vs high-throughput allreduce.

- **Topology Finder:** Select the best-suited topology and schedule for given hardware and workload specifications.

# Expansion Techniques



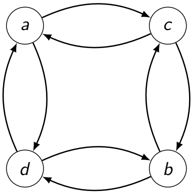
# Expansion Techniques

**Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

# Expansion Techniques

**Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

Given a small-scale base topology,



# Expansion Techniques

**Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

Given a small-scale base topology,

- We apply **graph transformations** to map nodes and links into a larger topology.
  - e.g., line graph expansion: $N$-node degree-$d$ graph $\Longrightarrow$ $dN$-node degree-$d$ graph.

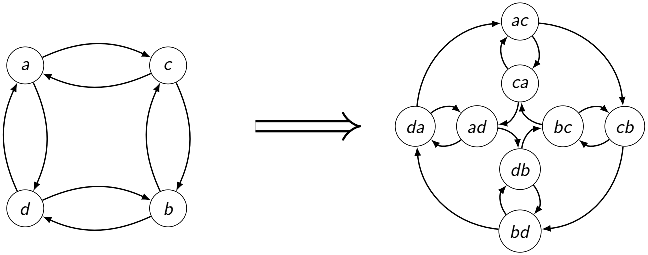



Figure: Line Graph Expansion

# Expansion Techniques

**Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

Given a small-scale base topology,

- We apply **graph transformations** to map nodes and links into a larger topology.
  - e.g., line graph expansion: $N$-node degree-$d$ graph $\Longrightarrow$ $dN$-node degree-$d$ graph.
- The **communication schedule** on the base topology can also be mapped to the larger topology.

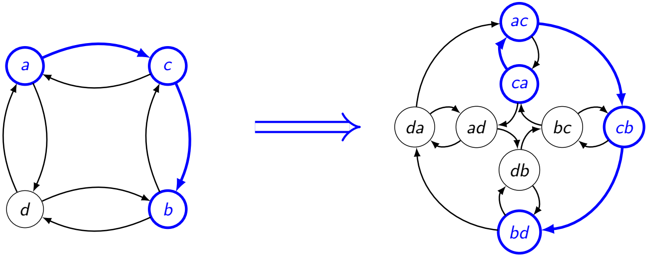



Figure: Line Graph Expansion

**Expansion Techniques:** Generate larger-scale topologies and schedules from small-scale ones.

Given a small-scale base topology,

- We apply **graph transformations** to map nodes and links into a larger topology.
  - e.g., line graph expansion: $N$-node degree-$d$ graph $\Longrightarrow$ $dN$-node degree-$d$ graph.
- The **communication schedule** on the base topology can also be mapped to the larger topology.
- The expansion can be applied **repeatedly** to scale topologies and schedules indefinitely.

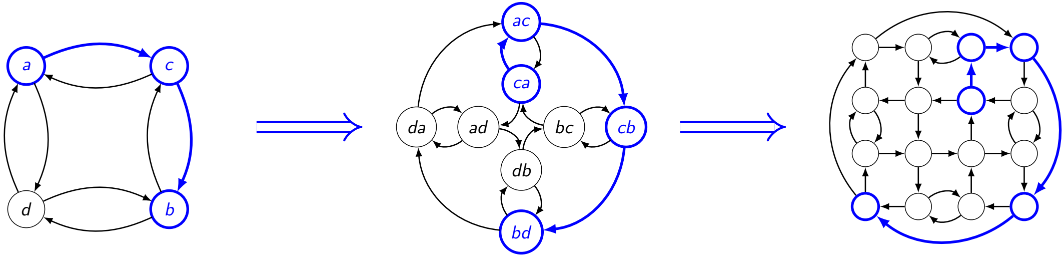



Figure: Line Graph Expansion

# Expansion Techniques

- **Line Graph Expansion:** expanding topology size while maintaining the same degree.

# Expansion Techniques

We have a variety of expansion techniques offering different characteristics:

- **Line Graph Expansion:** expanding topology size while maintaining the same degree.
- **Degree Expansion:** expanding both topology size and degree.
- **Cartesian Product Expansion:** creating a new topology by combining existing ones.

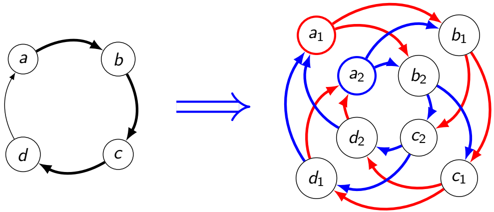



Figure: Degree Expansion

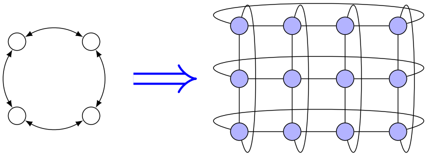


Figure: Cartesian Product Expansion

# Expansion Techniques

We have a variety of expansion techniques offering different characteristics:

- **Line Graph Expansion:** expanding topology size while maintaining the same degree.
- **Degree Expansion:** expanding both topology size and degree.
- **Cartesian Product Expansion:** creating a new topology by combining existing ones.

**Result:** These techniques enrich the pool of available topologies and schedules.

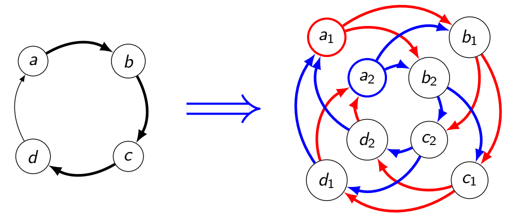



Figure: Degree Expansion

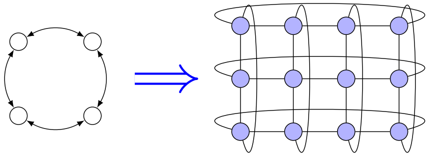


Figure: Cartesian Product Expansion

# Expansion Techniques

Expansions offer **performance guarantees** for the expanded topologies and schedules.

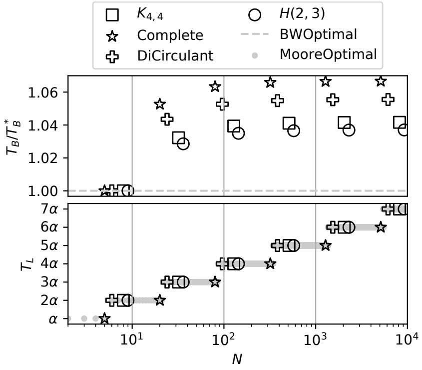



Figure: Line Graph Exp Performance Scaling

# Expansion Techniques

Expansions offer **performance guarantees** for the expanded topologies and schedules.

For example, in line graph expansion:

- If the base is throughput-optimal, then the expanded is $\leq \frac{1}{(d-1)N}$ away from optimality asymptotically.

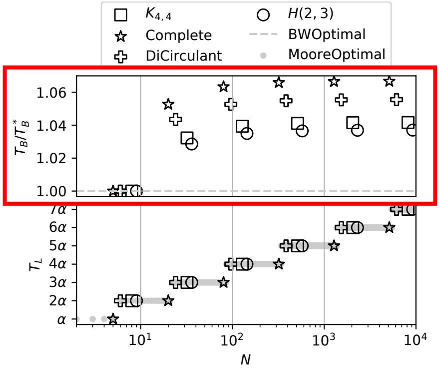



Figure: Line Graph Exp Performance Scaling

# Expansion Techniques

Expansions offer **performance guarantees** for the expanded topologies and schedules.

For example, in line graph expansion:

- If the base is throughput-optimal, then the expanded is $\leq \frac{1}{(d-1)N}$ away from optimality asymptotically.
- The topology maintains a low diameter, with diameter growth following $\mathcal{O}(\log_d N)$ as $N\uparrow$.

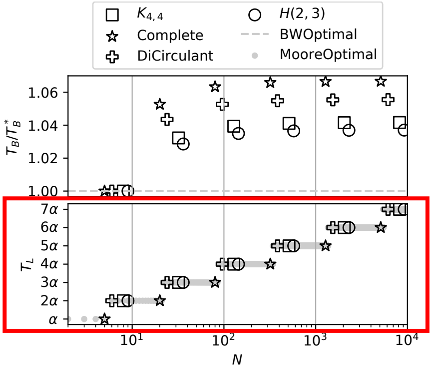
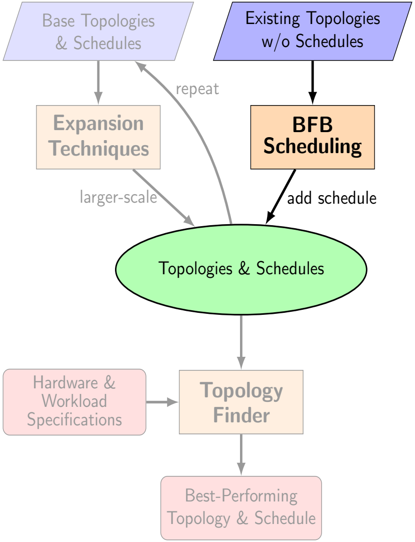



Figure: Line Graph Exp Performance Scaling

# Breadth-First-Broadcast Schedule



# Motivation

- **Observations:**
  - Expansion techniques may produce limited options for certain topology sizes.
  - There exist plenty of off-the-shelf low-diameter expander graphs from graph theory.
    - **Problem:** lack of efficient allreduce schedules.

# Motivation

- **Observations:**
  - Expansion techniques may produce limited options for certain topology sizes.
  - There exist plenty of off-the-shelf low-diameter expander graphs from graph theory.
    - **Problem:** lack of efficient allreduce schedules.
- **Question:** Can we utilize these off-the-shelf expander graphs by generating allreduce communication schedules for them?

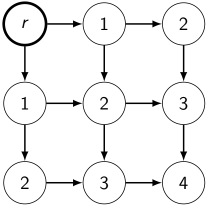
# Motivation

- **Observations:**
  - Expansion techniques may produce limited options for certain topology sizes.
  - There exist plenty of off-the-shelf low-diameter expander graphs from graph theory.
    - **Problem:** lack of efficient allreduce schedules.
- **Question:** Can we utilize these off-the-shelf expander graphs by generating allreduce communication schedules for them?
- **Challenge:** Generating collective communication schedules can easily be an NP-hard problem.
  - SCCL [PPoPP '21]: *satisfiability modulo theories* (SMT).
  - TACCL [NSDI '23], TE-CCL [SIGCOMM '24]: *mixed integer linear program* (MILP).
  - Existing approaches are unable to scale to large topologies.

# Breadth-First-Broadcast Schedule

For any given topology, we propose **Breadth-First-Broadcast** (BFB) allgather schedule.

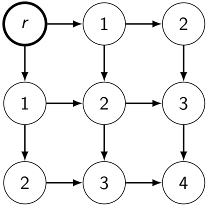
- Each node's data is broadcast to other nodes in a *breadth-first* order along the shortest paths.




$$
\begin{aligned}
\text{minimize} \quad & U_{u,t} \\
\text{subject to} \quad & \sum_{v} x_{v,(w,u),t} \leq U_{u,t}, && \forall w \in N^{-}(u) \\
& \sum_{w} x_{v,(w,u),t} = 1, && \forall v \in N_t^{-}(u) \\
& 0 \leq x_{v,(w,u),t} \leq 1. && \forall w, v
\end{aligned}
$$

For any given topology, we propose **Breadth-First-Broadcast** (BFB) allgather schedule.

- Each node's data is broadcast to other nodes in a *breadth-first* order along the shortest paths.
- **Optimization:** Choosing among multiple shortest paths from src to dst to minimize congestion.

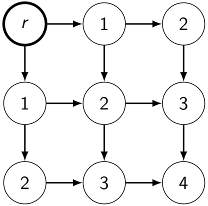



$$
\begin{array}{lll}
\text{minimize} & U_{u,t} & \\
\text{subject to} & \sum_{v} x_{v,(w,u),t} \leq U_{u,t}, & \forall w \in N^{-}(u) \\
& \sum_{w} x_{v,(w,u),t} = 1, & \forall v \in N_t^{-}(u) \\
& 0 \leq x_{v,(w,u),t} \leq 1. & \forall w, v
\end{array}
$$

# Breadth-First-Broadcast Schedule

For any given topology, we propose **Breadth-First-Broadcast** (BFB) allgather schedule.

- Each node's data is broadcast to other nodes in a *breadth-first* order along the shortest paths.
- **Optimization:** Choosing among multiple shortest paths from src to dst to minimize congestion.
- **Scalability:** Breadth-first ordering allows optimization via polynomial-time **linear programs**.

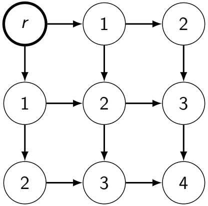



$$
\begin{aligned}
\text{minimize} \quad & U_{u,t} \\
\text{subject to} \quad & \sum_{v} x_{v,(w,u),t} \le U_{u,t}, && \forall w \in N^{-}(u) \\
& \sum_{w} x_{v,(w,u),t} = 1, && \forall v \in N_t^{-}(u) \\
& 0 \le x_{v,(w,u),t} \le 1. && \forall w, v
\end{aligned}
$$

# Breadth-First-Broadcast Schedule

For any given topology, we propose **Breadth-First-Broadcast** (BFB) allgather schedule.

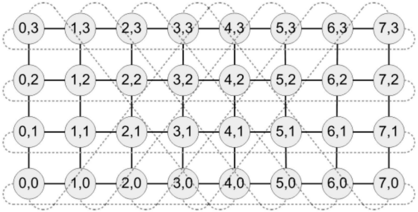
- Each node's data is broadcast to other nodes in a *breadth-first* order along the shortest paths.
- **Optimization:** Choosing among multiple shortest paths from src to dst to minimize congestion.
- **Scalability:** Breadth-first ordering allows optimization via polynomial-time **linear programs**.
- The resulting allgather schedule can be easily transformed into reduce-scatter and allreduce as well.



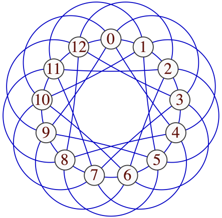

$$
\begin{array}{lll}
\text{minimize} & U_{u,t} & \\
\text{subject to} & \sum_{v} x_{v,(w,u),t} \leq U_{u,t}, & \forall w \in N^{-}(u) \\
 & \sum_{w} x_{v,(w,u),t} = 1, & \forall v \in N_t^{-}(u) \\
 & 0 \leq x_{v,(w,u),t} \leq 1. & \forall w, v
\end{array}
$$

# BFB Efficient Topologies

- **Significance:** BFB enables efficient collective operations on complex topologies.
  - Previously, collective operations are limited to simple variants of rings (e.g., multiring, torus).



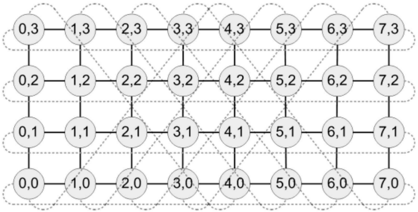

(a) Twisted Torus



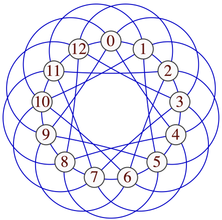

(b) Circulant Graph

# BFB Efficient Topologies

- **Significance:** BFB enables efficient collective operations on complex topologies.
  - Previously, collective operations are limited to simple variants of rings (e.g., multiring, torus).
- **Performance:** BFB offers mathematically provable performance guarantees on many topologies.



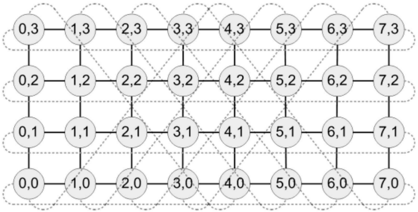

(a) Twisted Torus



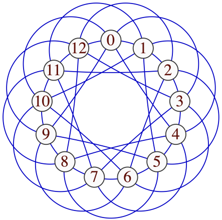

(b) Circulant Graph

# BFB Efficient Topologies

- **Significance:** BFB enables efficient collective operations on complex topologies.
  - Previously, collective operations are limited to simple variants of rings (e.g., multiring, torus).
- **Performance:** BFB offers mathematically provable performance guarantees on many topologies.
  - Throughput-optimal on **Asymmetric Torus** and TPU v4's **Twisted Torus**.



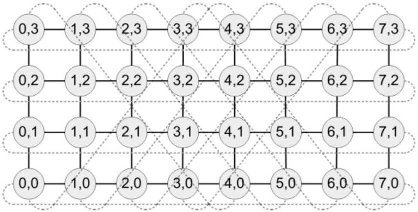

(a) Twisted Torus



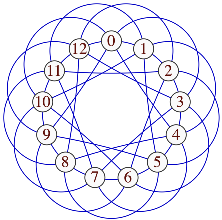

(b) Circulant Graph

# BFB Efficient Topologies

- **Significance:** BFB enables efficient collective operations on complex topologies.
  - Previously, collective operations are limited to simple variants of rings (e.g., multiring, torus).
- **Performance:** BFB offers mathematically provable performance guarantees on many topologies.
  - Throughput-optimal on **Asymmetric Torus** and TPU v4's **Twisted Torus**.
  - Throughput-optimal with $\mathcal{O}(\sqrt{N})$ diameter on **Circulant Graph** for any $N$ and even-value $d$.



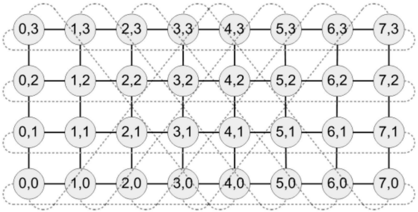

(a) Twisted Torus



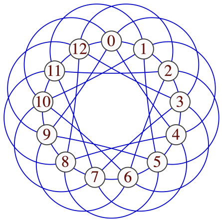

(b) Circulant Graph

# BFB Efficient Topologies

- **Significance:** BFB enables efficient collective operations on complex topologies.
  - Previously, collective operations are limited to simple variants of rings (e.g., multiring, torus).
- **Performance:** BFB offers mathematically provable performance guarantees on many topologies.
  - Throughput-optimal on **Asymmetric Torus** and TPU v4's **Twisted Torus**.
  - Throughput-optimal with $\mathcal{O}(\sqrt{N})$ diameter on **Circulant Graph** for any $N$ and even-value $d$.
  - Also others like distance-regular graphs, generalized-Kautz graphs, etc.




(a) Twisted Torus




(b) Circulant Graph

# BFB vs Existing Schedule Generations

BFB schedule generation excels in both speed and quality.

- **Scalability:** BFB generation is orders of magnitude faster than previous methods.
- **Schedule Performance:** BFB schedules are theoretically optimal on hypercube and torus.

| # of nodes | 4 | 8 | 16 | 32 | 64 | **1024** |
|---|---|---|---|---|---|---|
| SCCL | 0.59s | 0.86s | 21.4s | $>10^4$s | $>10^4$s | $>10^4$s |
| TACCL | 0.50s | 7.39s | 1801s | 1802s | n/a | n/a |
| **BFB** | **<0.01s** | **<0.01s** | **<0.01s** | **0.03s** | **0.17s** | **52.7s** |

Table: Generation Time on Hypercube

| # of nodes | 4 | 9 | 16 | 25 | 36 | **2500** |
|---|---|---|---|---|---|---|
| SCCL | 0.61s | 1.00s | 60s | 3286s | $>10^4$s | $>10^4$s |
| TACCL | 0.45s | 67.8s | 1801s | 1802s | n/a | n/a |
| **BFB** | **<0.01s** | **<0.01s** | **<0.01s** | **0.01s** | **0.03s** | **61.1s** |

Table: Generation Time on 2D Torus ($n \times n$)

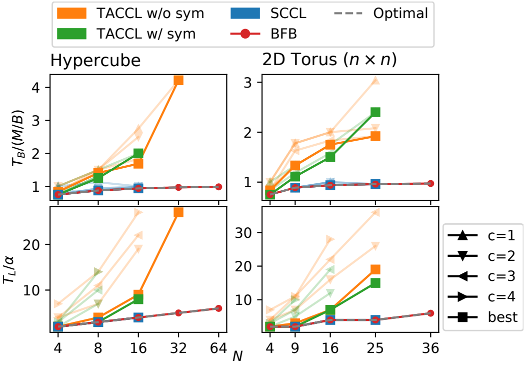
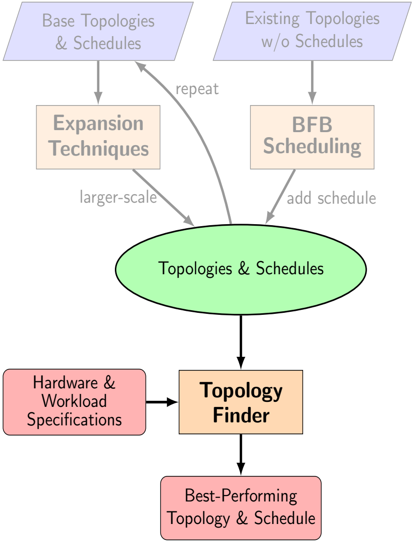



Figure: Theoretical Performance of Schedules

# Topology Finder



# Topology Finder

Given a target topology size ($N$ and $d$),

# Topology Finder

Given a target topology size ($N$ and $d$),

- The topology finder explores **combinations of base topologies and expansion techniques** to generate topologies of the target size.

| Topology | | | | | | |
|---|---|---|---|---|---|---|
| $\Pi_{4,1024}$<br>$L^3(C(16,\{3,4\}))$<br>$L^2(\text{Diamond}^{\square 2})$<br>$L(\text{DBJMod}(2,4)^{\square 2})$<br>$(\text{UniRing}(1,4)\square\text{UniRing}(1,8))^{\square 2}$ | | | | | | |
| | | | | | | |
| | | | | | | |

Table: Pareto-frontier for $N=1024$, $d=4$ with $\alpha=10\mu s$ and $M/B=1\text{MB}/100\text{Gbps}$.

# Topology Finder

Given a target topology size ($N$ and $d$),

- The topology finder explores **combinations of base topologies and expansion techniques** to generate topologies of the target size.
- The resulting topologies and schedules form a **Pareto-frontier**.

low $\Longleftrightarrow$ high

Diameter

| Topology | Diameter | | | | | |
|---|---|---|---|---|---|---|
| $\Pi_{4,1024}$ | 5 | | | | | |
| $L^3(C(16,\{3,4\}))$ | 6 | | | | | |
| $L^2(\mathrm{Diamond}^{\square 2})$ | 8 | | | | | |
| $L(\mathrm{DBJMod}(2,4)^{\square 2})$ | 9 | | | | | |
| $(\mathrm{UniRing}(1,4)\square\mathrm{UniRing}(1,8))^{\square 2}$ | 20 | | | | | |
| | | | | | | |
| | | | | | | |

Table: Pareto-frontier for $N=1024$, $d=4$ with $\alpha=10\mu s$ and $M/B=1\text{MB}/100\text{Gbps}$.

# Topology Finder

Given a target topology size ($N$ and $d$),

- The topology finder explores **combinations of base topologies and expansion techniques** to generate topologies of the target size.
- The resulting topologies and schedules form a **Pareto-frontier**.

high-throughput all-to-all, low-latency allreduce

low $\Longleftrightarrow$ high

Diameter

| Topology | Diameter | All-to-All MCF | Latency | | | |
|---|---|---|---|---|---|---|
| $\Pi_{4,1024}$ | 5 | 8.01e−4 | $5\alpha$ | | | |
| $L^3(C(16, \{3,4\}))$ | 6 | 8.12e−4 | $6\alpha$ | | | |
| $L^2(\mathrm{Diamond}^{\square 2})$ | 8 | 7.34e−4 | $8\alpha$ | | | |
| $L(\mathrm{DBJMod}(2,4)^{\square 2})$ | 9 | 6.18e−4 | $11\alpha$ | | | |
| $(\mathrm{UniRing}(1,4)\square\mathrm{UniRing}(1,8))^{\square 2}$ | 20 | 2.79e−4 | $20\alpha$ | | | |
| | | | | | | |
| | | | | | | |

Table: Pareto-frontier for $N=1024$, $d=4$ with $\alpha=10\mu s$ and $M/B=1\text{MB}/100\text{Gbps}$.

# Topology Finder

Given a target topology size ($N$ and $d$),

- The topology finder explores **combinations of base topologies and expansion techniques** to generate topologies of the target size.
- The resulting topologies and schedules form a **Pareto-frontier**.

high-throughput all-to-all, low-latency allreduce $\xLeftarrow{\text{low}\qquad\qquad\qquad\text{high}}\!\!\!\!\Rightarrow$ high-throughput allreduce

Diameter

| Topology | Diameter | All-to-All MCF | Latency | Throughput | | |
|---|---|---|---|---|---|---|
| $\Pi_{4,1024}$ | 5 | 8.01e$-$4 | $5\alpha$ | $0.751B$ | | |
| $L^3(C(16,\{3,4\}))$ | 6 | 8.12e$-$4 | $6\alpha$ | $0.981B$ | | |
| $L^2(\text{Diamond}^{\square 2})$ | 8 | 7.34e$-$4 | $8\alpha$ | $0.996B$ | | |
| $L(\text{DBJMod}(2,4)^{\square 2})$ | 9 | 6.18e$-$4 | $11\alpha$ | $1.000B$ | | |
| $(\text{UniRing}(1,4)\square\text{UniRing}(1,8))^{\square 2}$ | 20 | 2.79e$-$4 | $20\alpha$ | $1.001B$ | | |
| | | | | | | |
| | | | | | | |

Table: Pareto-frontier for $N=1024$, $d=4$ with $\alpha=10\mu s$ and $M/B=1\text{MB}/100\text{Gbps}$.

# Topology Finder

Given a target topology size ($N$ and $d$),

- The topology finder explores **combinations of base topologies and expansion techniques** to generate topologies of the target size.
- The resulting topologies and schedules form a **Pareto-frontier**.

$$\begin{array}{c}\text{high-throughput all-to-all,}\\\text{low-latency allreduce}\end{array} \xleftarrow[\text{Diameter}]{\text{low} \qquad\qquad\qquad\qquad \text{high}} \text{high-throughput allreduce}$$

- The best-suited topology and schedule are selected based on the hardware (e.g., latency, bandwidth) and workload (e.g., allreduce and all-to-all sizes) specifications.

| Topology | Diameter | All-to-All MCF | Latency | Throughput | All-to-All | Allreduce |
|---|---|---|---|---|---|---|
| $\Pi_{4,1024}$ | 5 | 8.01e−4 | $5\alpha$ | $0.751B$ | 409.1us | 323.5us |
| $L^3(C(16,\{3,4\}))$ | 6 | 8.12e−4 | $6\alpha$ | $0.981B$ | **403.5us** | **291.0us** |
| $L^2(\text{Diamond}^{\square 2})$ | 8 | 7.34e−4 | $8\alpha$ | $0.996B$ | 446.6us | 328.4us |
| $L(\text{DBJMod}(2,4)^{\square 2})$ | 9 | 6.18e−4 | $11\alpha$ | $1.000B$ | 529.9us | 387.8us |
| $(\text{UniRing}(1,4)\square\text{UniRing}(1,8))^{\square 2}$ | 20 | 2.79e−4 | $20\alpha$ | $1.001B$ | 1174.4us | 567.6us |
| **Baseline:** 32x32 Torus | 32 | 2.44e−4 | $62\alpha$ | $1.001B$ | 1342.2us | 1407.6us |
| **Theoretical Bound** | **5** | **8.57e−4** | $\mathbf{5\alpha}$ | $\mathbf{1.001B}$ | **382.3us** | **267.6us** |

Table: Pareto-frontier for $N=1024$, $d=4$ with $\alpha=10\mu s$ and $M/B=1\text{MB}/100\text{Gbps}$.

# Evaluation

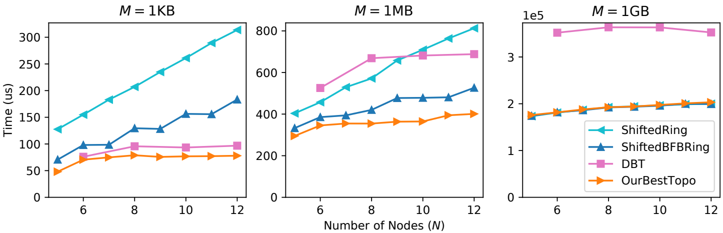
- Experiments on small-scale optical network testbed
- Experiments on Frontera Supercomputer
- Simulated large-scale MoE training

# Optical Testbed Experiments

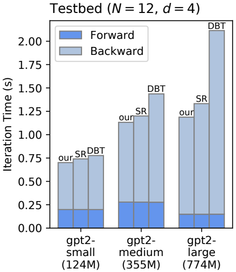
**Optical Testbed:** 12x A100 nodes with reconfigurable optical interconnects ($d = 4$).

- **Allreduce Experiments:** our generated topologies outperform shifted ring and double binary tree across topology sizes and allreduce data sizes.
- **GPT-2 Training:** our generated topologies consistently surpass baselines in data-parallel training across varying model sizes.

(a) Optical Testbed




(b) Allreduce Experiments

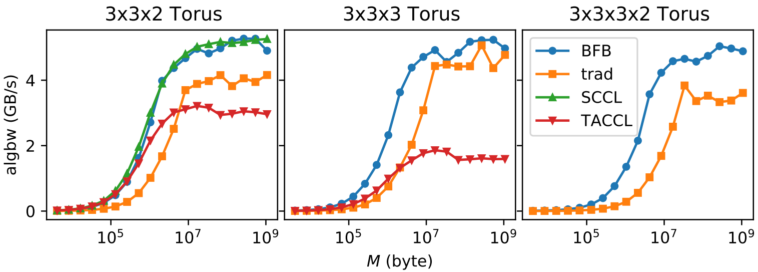



(c) GPT-2 Training

# Supercomputing Evaluation

**Frontera Supercomputer** at the Texas Advanced Computing Center (TACC)

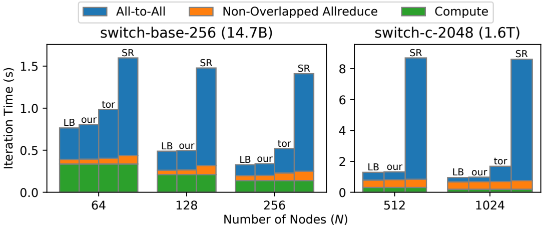
- **Topology:** various configurations of multi-dimensional torus.
- **Asymmetric Torus:** BFB torus schedules significantly outperform traditional torus scheduling.
- **Scalability:** BFB scales to topology sizes beyond the reach of other schedule generation methods.



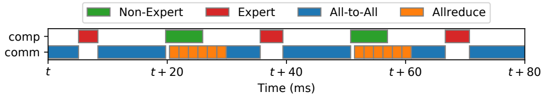
# Simulated Expert-Parallel Training

Expert-parallel training involves both **allreduce** and **all-to-all** communications.

- **Performance:** Efficient in both allreduce and all-to-all, our topology outperforms torus and shifted ring by 40%+ in MoE model training.
- **Theoretical Bound:** Our topologies remain within 5% of the theoretical lower bound at all times.




(a) Simulated Training of Switch Transformers.



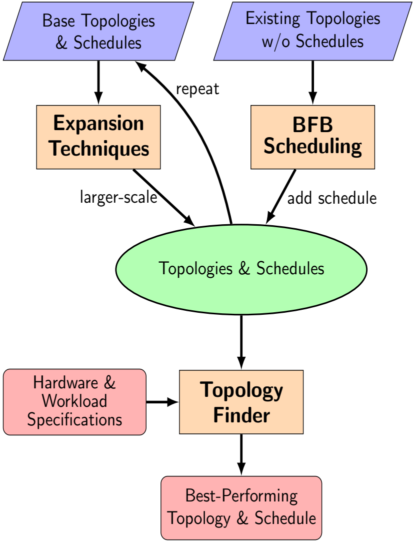

(b) Training Timeline.

# Conclusion

In this work, we introduce

- **Expansion techniques** to expand small-scale optimized topologies and schedules into large-scale ones.
- **Breadth-First-Broadcast** method to generate efficient communication schedules for large-scale topologies in polynomial time.
- **Topology Finder** to explore and identify the best-suited topology for the given hardware and workload.

Together, we enable efficient collective communications with direct-connect topologies.




Efficient Direct-Connect Topologies for Collective Communications
Contact: liangyu@cs.washington.edu
NSDI '25