Beyond fat-trees without antennae, mirrors, and disco-balls

Simon Kassing, Asaf Valadarsky, Gal Shahaf, Michael Schapira, Ankit Singla
Skewed traffic within data centers
Skewed traffic within data centers
All-to-all non-blocking connectivity is expensive
Oversubscribed fat-trees
Oversubscribed fat-trees
Oversubscribed fat-trees

Capacity

75%
Oversubscribed fat-trees

Bottleneck

Capacity: 75%
Demand: 50%
Oversubscribed fat-trees: \textbf{A tragedy ...}

$k = 96$

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>75%</td>
<td>2%</td>
</tr>
</tbody>
</table>
Dynamically set up network connections!

- OFC ’09  Glick et al.
- SIGCOMM ’10  Wang et al.
- SIGCOMM ’10  Farrington et al.
- SIGCOMM ’11  Halperin et al.
- NSDI ’12  Chen et al.
- SIGCOMM ’12  Zhou et al.
- SIGCOMM ’13  Porter et al.
- SIGCOMM ’14  Liu et al.
- SIGCOMM ’14  Hamedzimi et al.
- SIGCOMM ’16  Ghabadi et al.
- NSDI ’17  Chen et al.
Set up network connections on the fly!

**Advantage:**
Gained the ability to move links around
Set up network connections on the fly!

Advantage:
Gained the ability to move links around

Set up network connections **on the fly!**

**Advantage:**
Gained the ability to move links around

Engineering challenges facing dynamic topologies

- Spatial planning and organisation?
- Environmental factors?
- Lack of operational experience?
- Device packaging?
- Monitoring and debugging?
- Reliability and lifetime of devices?
- Unknown unknowns?
Foundational questions
Foundational questions

1. Rigorous benchmarks?

   Fat-trees are the easiest baseline — ideally inflexible!
Foundational questions

1. Rigorous benchmarks?
   Fat-trees are the easiest baseline — ideally inflexible!

2. What is the utility of dynamic links?
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand

\[ (1, \alpha) \]
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand
Ideally flexible network

\[ \alpha(x, \alpha/x) \]

Throughput per server

Fraction of servers with traffic demand
Ideally flexible network

Throughput per server

Fraction of servers with traffic demand

Throughput proportional

\[ \text{Throughput proportional} \]

\[ (\alpha, 1) \]

\[ (x, \alpha/x) \]

\[ (1, \alpha) \]
Fat-trees: ideally *in*flexible

Throughput per server

Throughput proportional

Fraction of servers with traffic demand

\[ \frac{\alpha}{x} \]
Near-optimal expander networks
Static but flexible
Instead of rigid, layered connectivity...
Expander-based data centers

- Jellyfish NSDI '12
- Slimfly SC '14
- Xpander CoNEXT '16
Xpander: deterministic wiring-friendly expander-based data center

A fundamental question…

Setup network connections on the fly!

How **valuable** is the **ability to move links** around?
Optimal flow comparison

Throughput per server

Fraction of servers with traffic demand
Baseline: oversubscribed fat-tree

![Graph showing throughput per server vs. fraction of servers with traffic demand. The line for the fat-tree is horizontal at a throughput of 0.2.]
Indeed, dynamic networks can be better
... but so can static ones

![Graph showing throughput per server vs. fraction of servers with traffic demand. Dynamic network (\(\delta=1.5\)), Expander, and Fat-tree are compared.]

Throughput per server

Dynamic network (\(\delta=1.5\))

Expander

Fat-tree

Fraction of servers with traffic demand

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

0 0.2 0.4 0.6 0.8 1
… especially in the regime of interest

“46-99% of the rack pairs exchange no traffic at all”
— Ghobadi et al., 2016
... especially in the regime of interest

“46-99% of the rack pairs exchange no traffic at all”
— Ghobadi et al., 2016
Not too far from proportionality!

Throughput per server

Fraction of servers with traffic demand

Throughput proportional

Dynamic network ($\delta=1.5$)

Expander

Fat-tree
Workloads

pFabric Web search (2.4MB mean)
Modelled after a real workload
Maximum flow size of 30MB

Pareto-HULL (100KB mean)
Pareto distributed
Highly skewed
Many short flows (<100 KB)
Few very large flows (max. 1GB)

... at a fixed arrival rate per second ($\lambda$)

Traffic scenarios

A2A(x): fractional all-to-all
Only the servers under x% of the ToRs communicate all-to-all

Permute(x): fractional random permutation
A random pairing of x% of the ToRs, of which in each pair all servers only communicate with the servers of the counterpart

ProjecToR
Empirical skewed traffic from a Microsoft cluster

Skew(x, y)
x fraction of ToRs has y probability of participating in a flow (rack-pair)
E.g. θ=4% of ToRs have φ=77% chance of participating in a flow
Topologies & Routing

Two topologies (k=16):
• Full fat-tree with n=320
• Xpander at with n=216 (67.5%)
  … with 10 Gbps links
  … both supporting ~1K servers

At servers:
• DCTCP
• Flowlets (change path upon exceeding gap)

Fat-tree:
• ECMP

Xpander:
• HYBRID
Introducing HYBRID routing

HYBRID routing:
• ECMP until # sent bytes > threshold Q
• After threshold Q, use valiant load balancing (VLB)

Advantages:
• Oblivious to the network congestion state
• Introduces little to no overhead in current switches
Experimental take-aways

• Xpander achieves *comparable performance* to non-blocking fabrics…
• At **lower cost**: 2/3rds or less
• Matching the performance of dynamic topologies
A2A(x): fractional all-to-all (pFabric)

99th %-tile FCT for small flows
(lower is better)

Average throughput for large flows
(higher is better)
A2A(x): fractional all-to-all (pFabric)

99th %-tile FCT for small flows (lower is better)

Average throughput for large flows (higher is better)
A2A(x): fractional all-to-all (pFabric)

99th %-tile FCT for small flows (lower is better)

Average throughput for large flows (higher is better)
**Permuted(x): fractional random permutation (pFabric)**

**99th %-tile FCT**  
for small flows  
(lower is better)

**Average throughput**  
for large flows  
(higher is better)
Permute(x): fractional random permutation (pFabric)

99th %-tile FCT for small flows
(lower is better)

Average throughput for large flows
(higher is better)
Permute(x): fractional random permutation (pFabric)

99th %-tile FCT for small flows (lower is better)

Average throughput for large flows (higher is better)
A2A(0.31) with many short flows (Pareto-HULL)

99th %-tile FCT for small flows (lower is better)

Average throughput for large flows (higher is better)
Comparing against ProjecToR

- Creating the same experiment as ProjecToR
- Same workload (pFabric)
- Same traffic scenario
- Same network sizes:
  
  **k=16 fat-tree**: 320 switches
  
  **d=16, r=8 Xpander**: 128 switches (40%)
ProjecToR: same # of network ports but static

Average FCT for all flows (lower is better)

Empirical

![Graph showing Average FCT vs Load \( \lambda \) for Fat-tree and Xpander HYB](image-url)
ProjecToR: same # of network ports but static

Average FCT for all flows (lower is better)

Empirical

Skew (4%, 77%)
Skew(4%, 77%) using same equipment at larger scale

k=24 fat-tree (720 switches)

d=13, r=11 Xpander (322 switches = 45%)

... both supporting ~3.5k servers

Average FCT for all flows (lower is better)
cheaper expander + simple, practical routing

= performance of full-bandwidth fat-tree
Expanders: the static topology benchmark

Demonstrating an advantage of dynamic topologies over static topologies requires...

• … comparing to expander-based static networks
• … at equal cost
• … using more expressive routing than ECMP
• … accounting for reconfiguration/buffering latency

All proposals to date don’t hit this benchmark
Future work

A. Better (oblivious) routing schemes?

B. Adaptive routing?

C. Deployment?
Get in touch

My e-mail: simon.kassing [at] inf.ethz.ch

Code available: https://github.com/ndal-eth/netbench