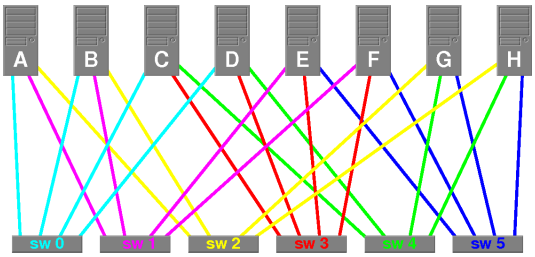
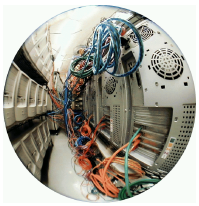
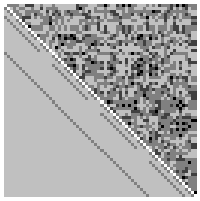
Flat Outperformance

INTERCONNECTION NETWORKS play a critical role on the performance of parallel computers – be they clusters spanning many racks or massively parallel supercomputers with multiple cores. Although commercial hardware and topologies with straight-forward mathematical properties are sometimes effective, communications within parallel programs tend to have properties specific to applications that allow a well-engineered network to dramatically outperform the obvious alternatives.

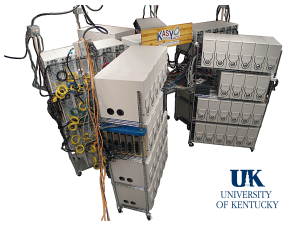


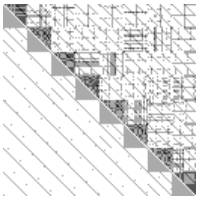
FLAT NEIGHBORHOOD NETWORKS (FNNs). *Bisection bandwidth* and *latency* are the critical metrics that affect performance on parallel systems. Using multiple network interfaces per node, FNNs provide *single-switch* latency and better bisection bandwidth than a FAT TREE with comparable hardware complexity. As shown above for 4-port switches connecting 8 nodes, an FNN connects nodes to switches such that each node-pair has at least one switch in common. The best wiring pattern usually is *asymmetric*; the design is *evolved* from random wiring patterns using a genetic algorithm.



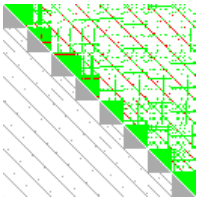


FNN design problems and solution quality are summarized by square maps in which the darkness of each point indicates how many single-switch paths exist between the corresponding pairs of nodes; the lower left triangle is the minimum design requirement, the upper right is what the FNN design actually provides. The map above shows the world's first FNN, which in April 2000 connected the 66 nodes of **KLAT2** (**K**ENTUCKY **L**INUX **A**THLON **T**ESTBED **2**) using 31-port 100Mb/s ETHERNET switches and standard IP to deliver close to 25Gb/s bisection bandwidth, and 30μs latency, at a network cost of about $8,100.

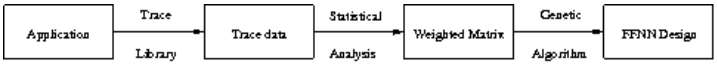




SPARSE FLAT NEIGHBORHOOD NETWORKS (SFNNs). Surprisingly, very few parallel programs depend on every node talking to every other; usually, each node talks to at most $O(log(N))$ other nodes. This is even true using personalized all-to-all communication as MPI typically implements it. By ensuring single-switch latency *only for node pairs that are expected to communicate*, SFNNs can provide single-switch latency for all critical communications in a typical application suite using cheap, narrow, switches for systems having many thousands of nodes. The first SFNN was demonstrated in **KASY0** (**K**ENTUCKY **ASY**METRIC ZER**0**), built in 2003, covering 128 nodes using 24-port switches – at about half the network cost of KLAT2's FNN.



FRACTIONAL FLAT NEIGHBORHOOD NETWORKS (FFNNs). Although SFNNs have great price/performance, the search is driven by performance. FFNNs flip priorities, finding the best coverage possible with a fixed network cost. For example, using far less network hardware than KASY0, the above map shows coverage of an FFNN in **green** – only the **red** spots deliver poorer latency. The design technique depends on empirically identifying the relative importance of each pair of communicating nodes in an application and covering only those pairs that *maximize* the sum of the importance values. The design process for an FFNN can be summarized as follows:



A 96-node cluster, **HAK**, some of whose nodes are shown above, is the testbed for FFNNs. Note that FNN, SFNN, and FFNN topologies all are fully compatible with ETHERNET, IP, and most other commonly available network technologies and protocols.

This document should be cited as:

```
@techreport{sc13flat,
author={Henry Dietz and Krishna Prabhala},
title={Flat Outperformance},
institution={University of Kentucky},
address={http://aggregate.org/WHITE/sc13flat.pdf},
month={Nov}, year={2013}}
```

Aggregate.Org
UNBRIDLED COMPUTING

SC13
Denver, CO | 2013