Using Multi-rail Networks in High-Performance Clusters

Eitan Frachtenberg

with

Salvador Coll, Fabrizio Petrini, Adolfy Hoisie and Leonid Gurvits

CCS-3 Modeling, Algorithms, and Informatics Group

Los Alamos National Laboratory

eitanf,scoll,fabrizio,hoisie,gurvits@lanl.gov

Los Alamos NATIONAL LABORATORY

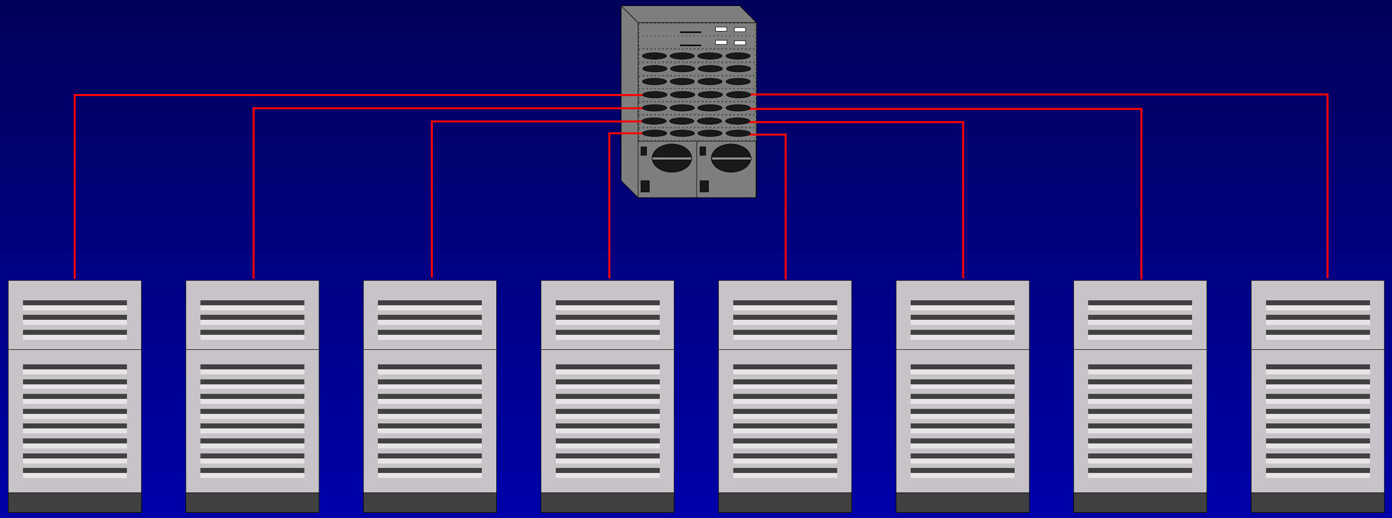
Multiple Independent Network Rails

Using multiple independent networks is an emerging technique to (1) overcome bandwidth limitations and (2) enhance fault-tolerance.

Los Alamos NATIONAL LABORATORY

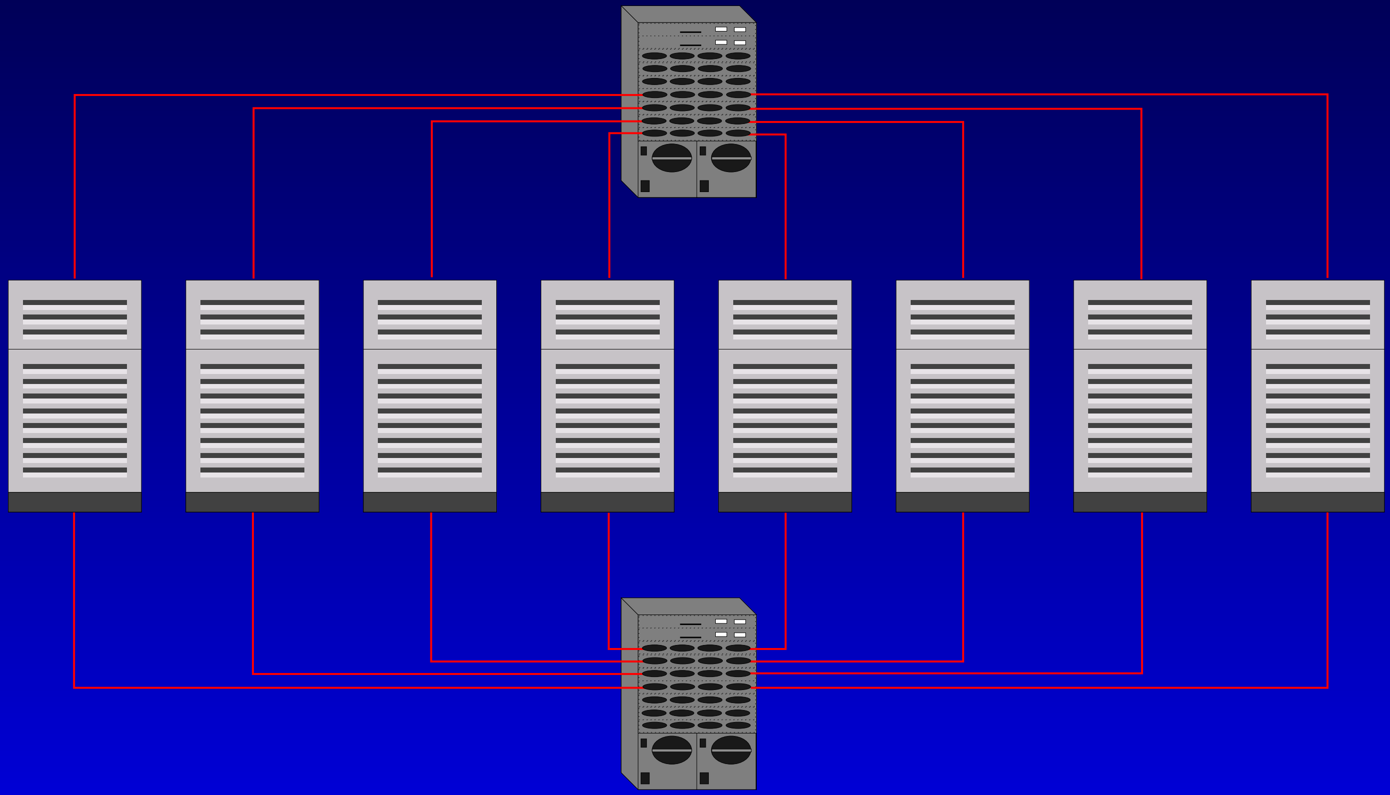
Multiple Independent Network Rails

Using multiple independent networks is an emerging technique to (1) overcome bandwidth limitations and (2) enhance fault-tolerance.



Multiple Independent Network Rails

Using multiple independent networks is an emerging technique to (1) overcome bandwidth limitations and (2) enhance fault-tolerance.



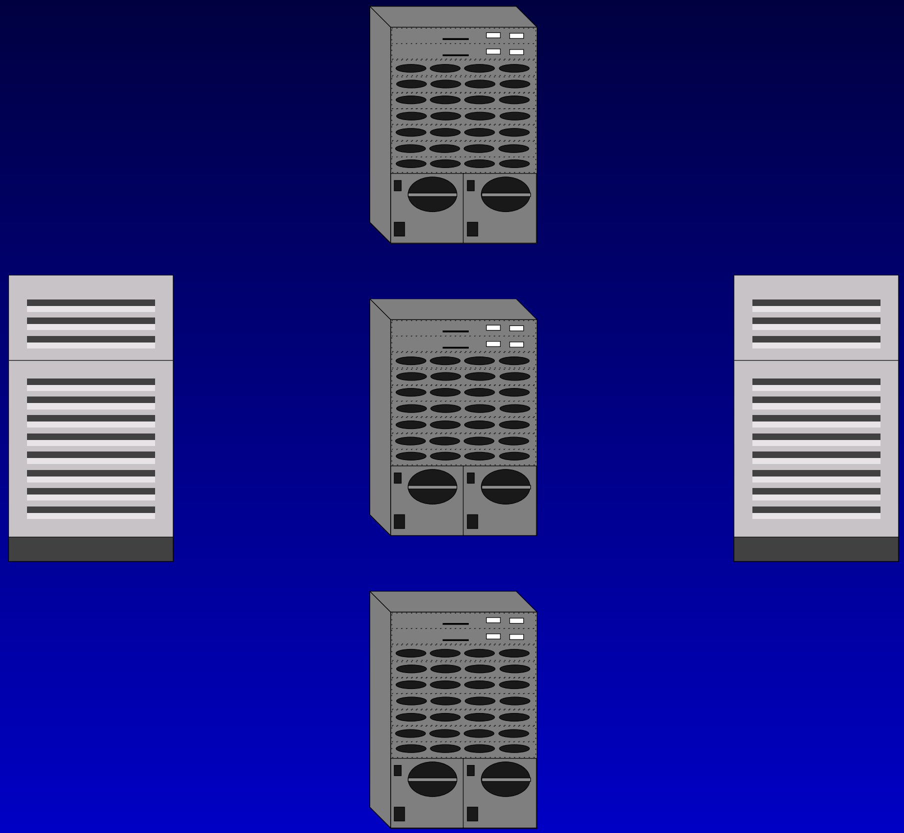
Examples of Multirailed Machines

- ASCI White at Lawrence Livermore National Laboratory (IBM SP)
- The Terascale Computing System (TCS) at the Pittsburgh Supercomputing Center (Quadrics)
- ASCI Q at Los Alamos National Laboratory (Quadrics)
- Experimental Linux clusters, with Infiniband, Quadrics and Myrinet

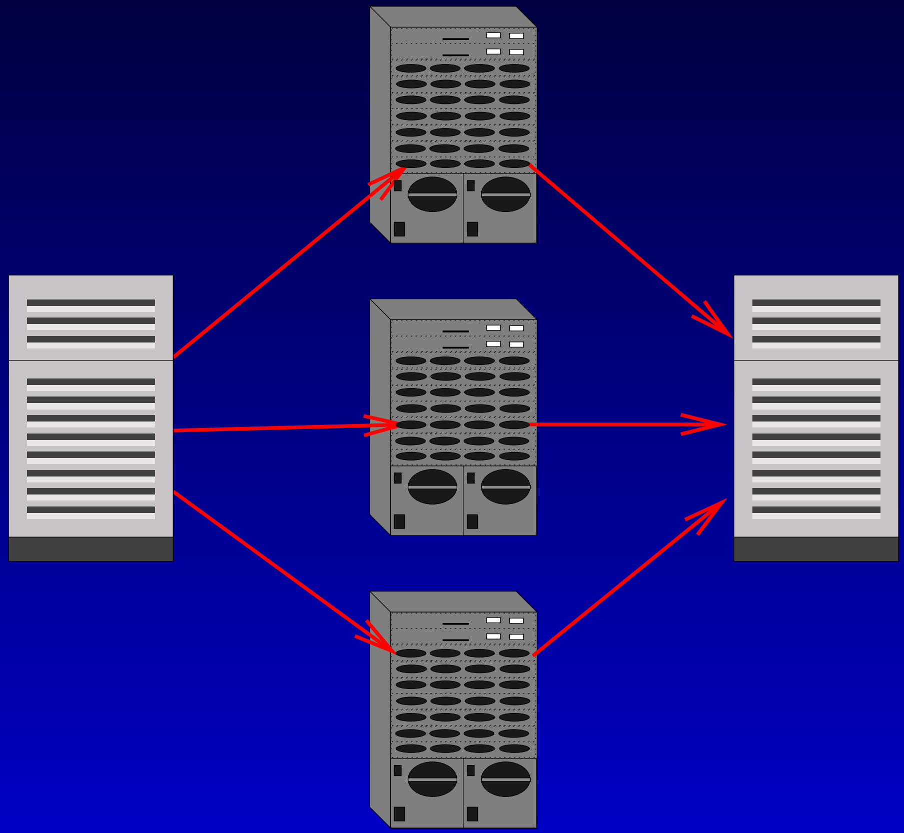
Challenges with Multirailed networks

- Rail assignment
- Striping over multiple rails
- Implementation of communication libraries (e.g., MPI, Cray Shmem)
- Interaction between NICs and I/O interfaces
- Congestion control: using multiple rails to decrease conflicts

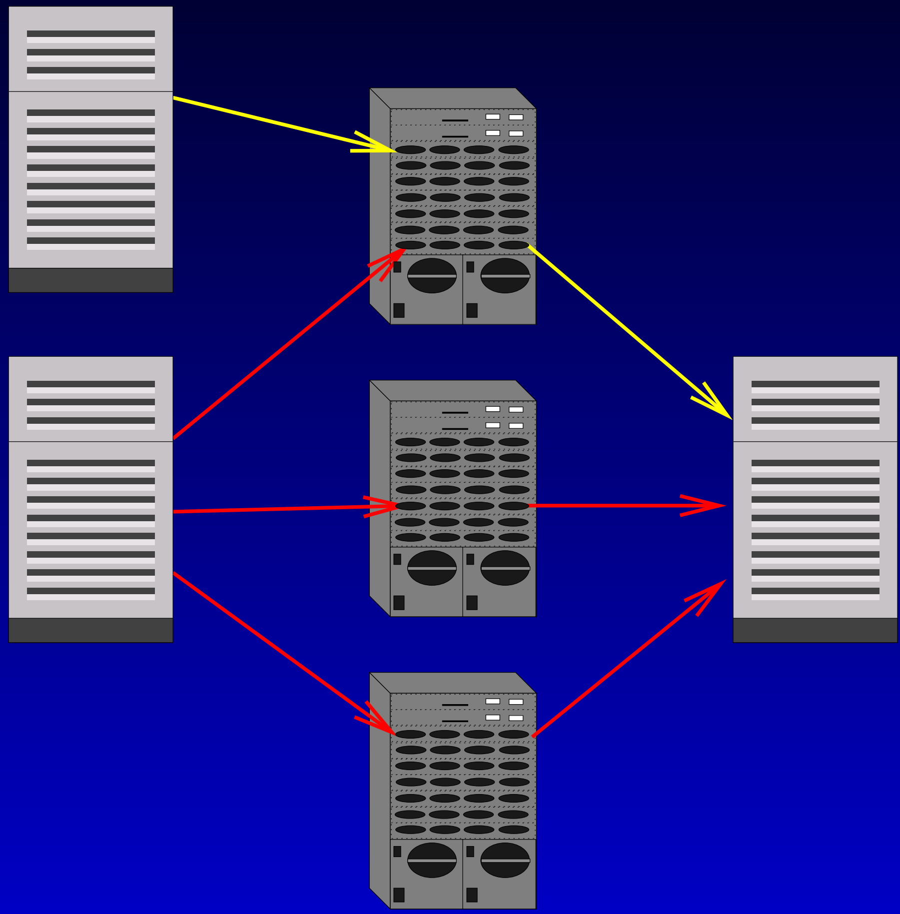
Rail Allocation



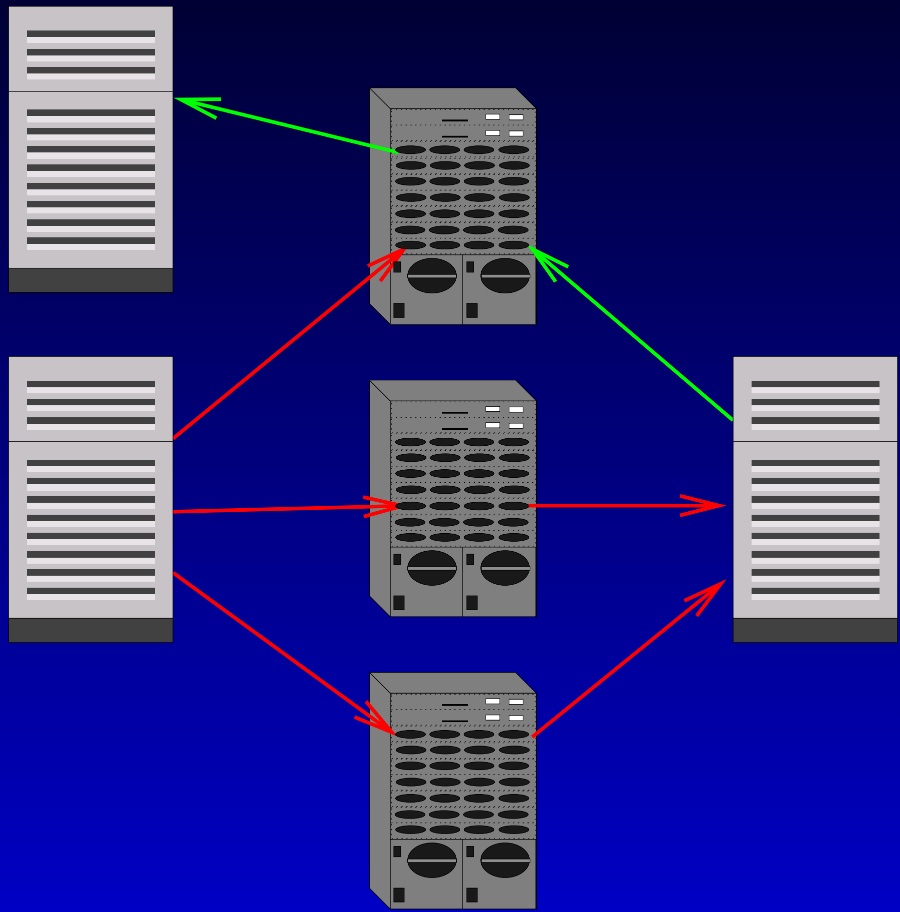
Rail Allocation



Rail Allocation



Rail Allocation



Bidirectional Traffic on the I/O bus

- Most PCI busses cannot efficiently handle bidirectional traffic with high performance networks
- Typically, aggregate bidirectional bandwidth is only 80% of the unidirectional one (Intel 840, Serverworks HE, Compaq Wildfire)
- PCI-X implementations (e.g., those based on the Intel 870) also suffer from performance degradation in bidirectional traffic
- Bidirectional traffic is very common in ASCI applications

Simple Rail Allocation

- A common algorithm to allocate messages to rails is to choose the rail based on the process id of the destination process (rail = destination_id $\bmod$ RAILS)
- Multiple processes can compete for the same rail even if other rails are available
- No message striping
- No attempt to minimize bidirectional traffic

Los Alamos NATIONAL LABORATORY

Outline

- Basic Algorithm
- Static rail allocation
- Dynamic rail allocation with local information
- Dynamic rail allocation with global information
- Experimental evaluation
- Hybrid algorithm
- Conclusions

Los Alamos NATIONAL LABORATORY

Basic (Round-Robin) Algorithm

- The basic algorithm doesn't use any communication protocol
- Whenever a node needs to send a message, it sends it on one rail, choosing it in round-robin fashion, blocking while it's busy
- Negligible overhead, but doesn't account for bidirectional traffic and congestion

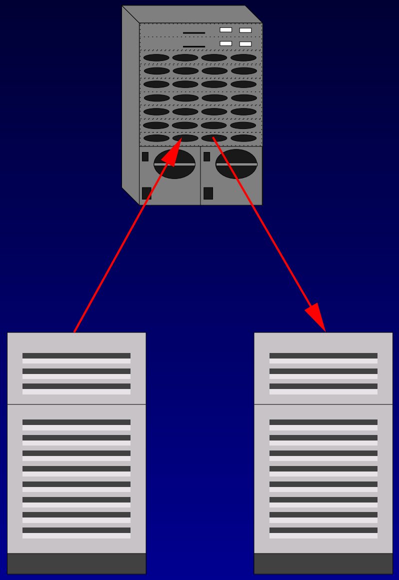
Los Alamos NATIONAL LABORATORY

Static Rail Allocation

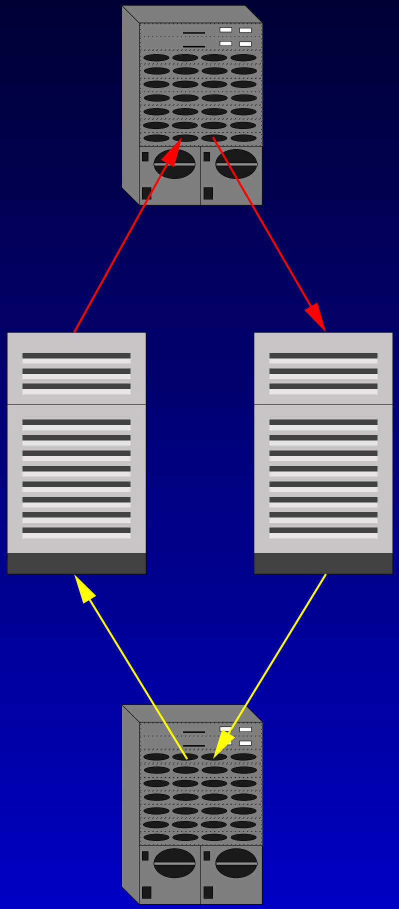
- With static rail allocation each network interface can either send or receive messages, and the direction is defined at initialization time.
- Initially proposed for ASCI Q, with 384 SMPs and 8 rails (ASCI Q is now 2048 SMPs and 2 rails)
- How many rails do we need for static allocation of n SMPs?

Los Alamos NATIONAL LABORATORY

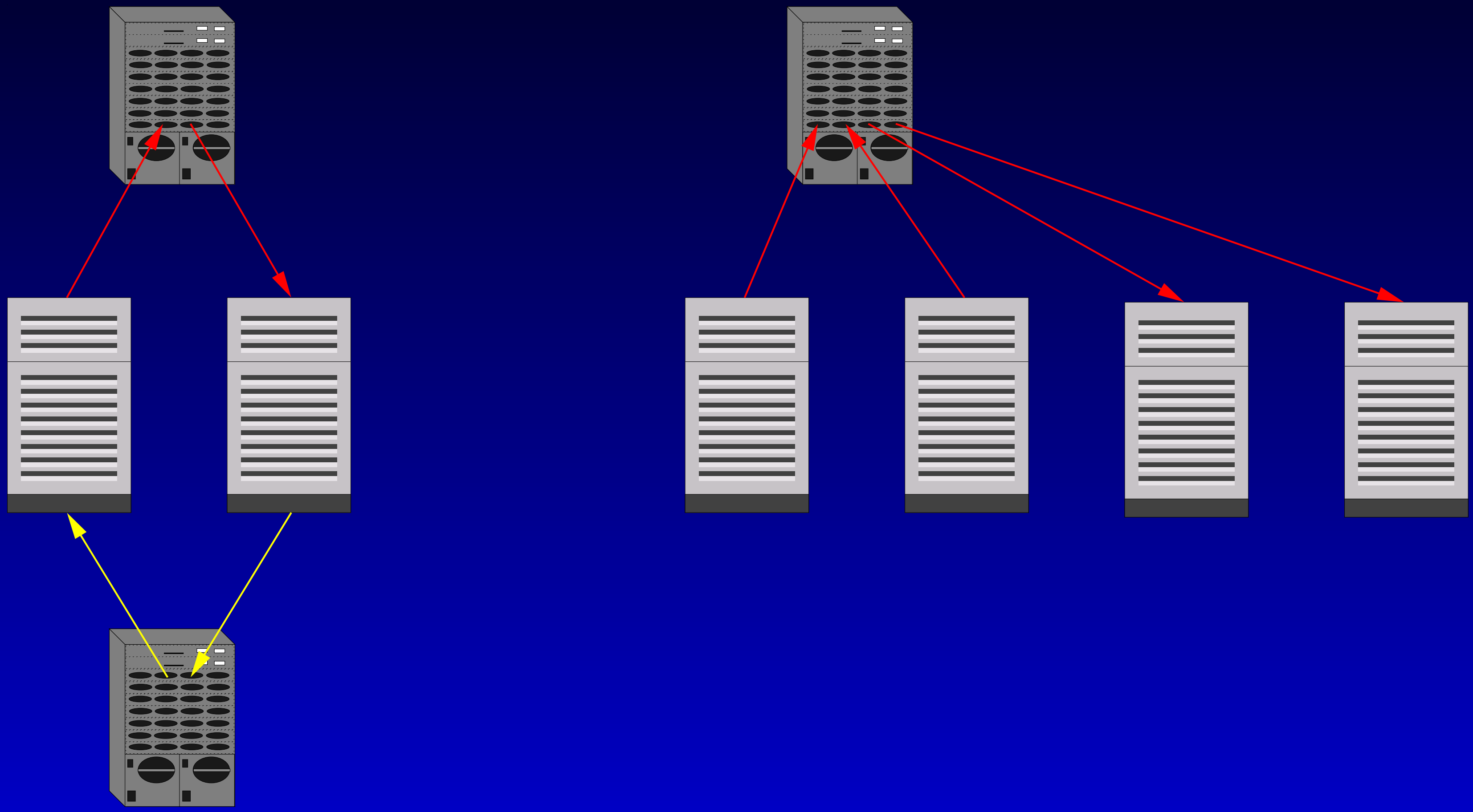
Static Rail Allocation Examples



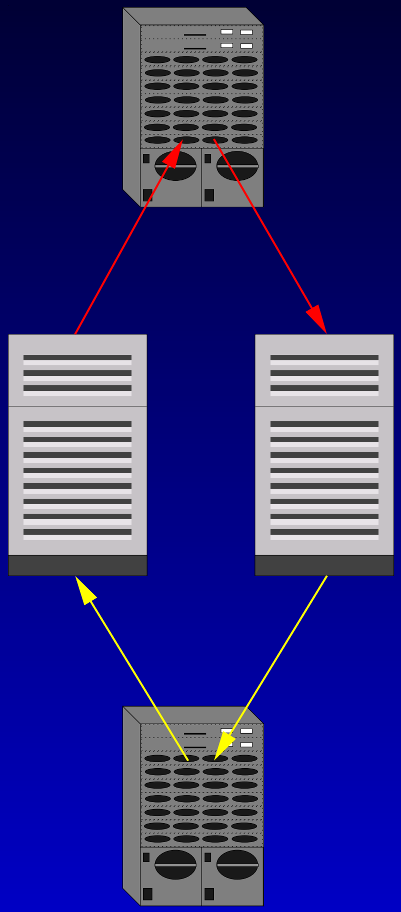
Static Rail Allocation Examples

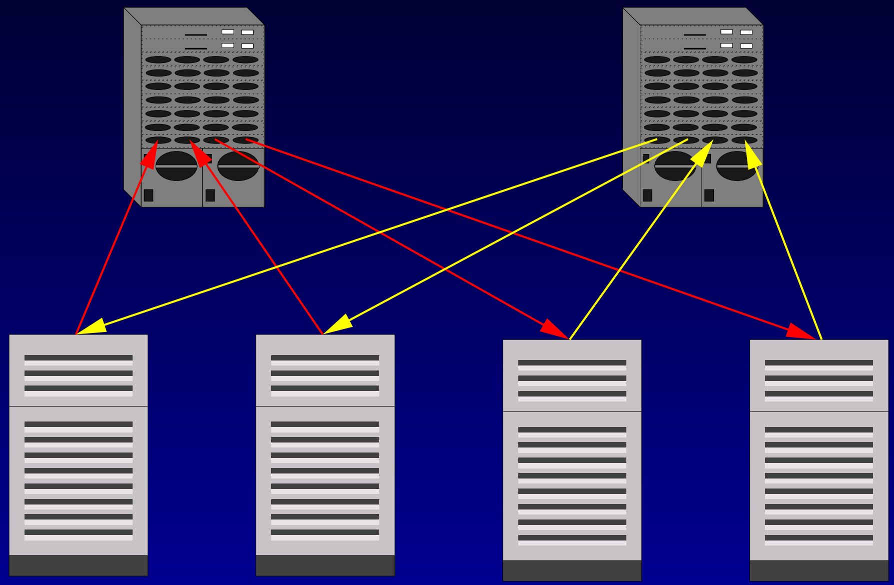


Static Rail Allocation Examples

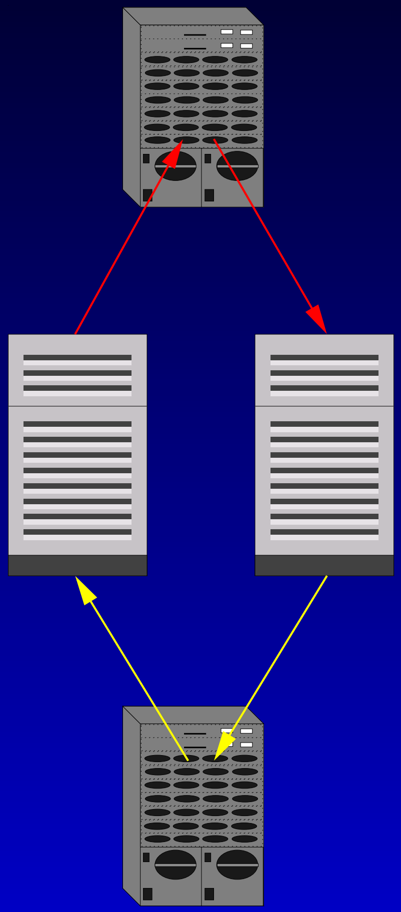


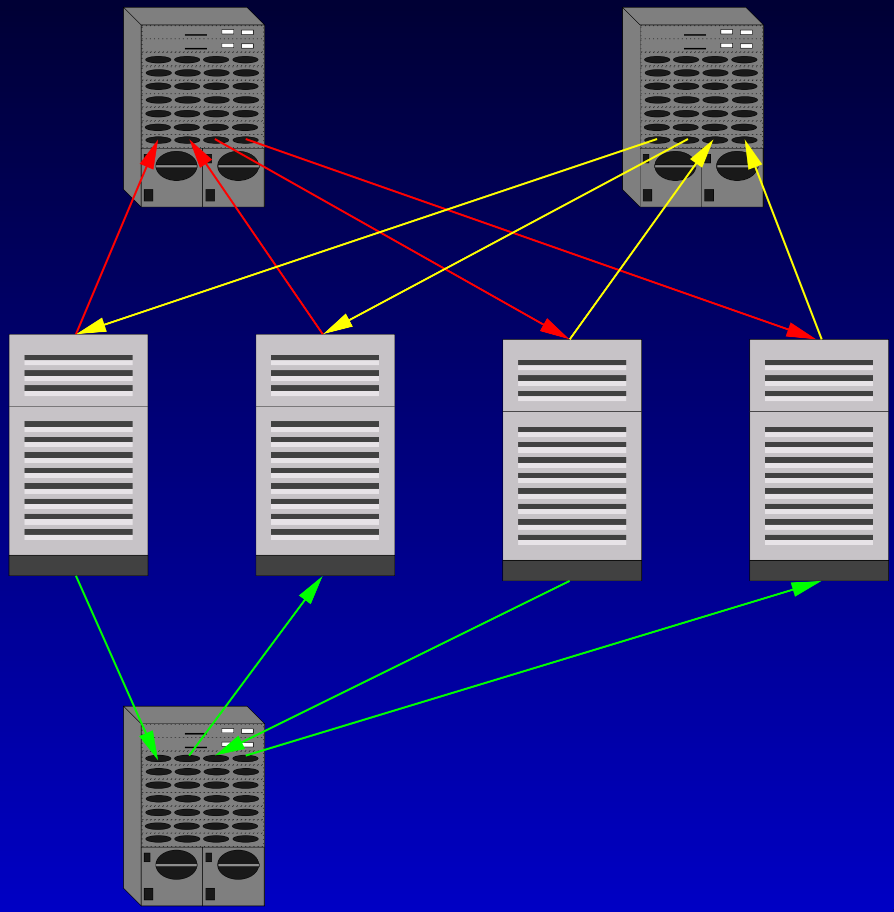
Static Rail Allocation Examples



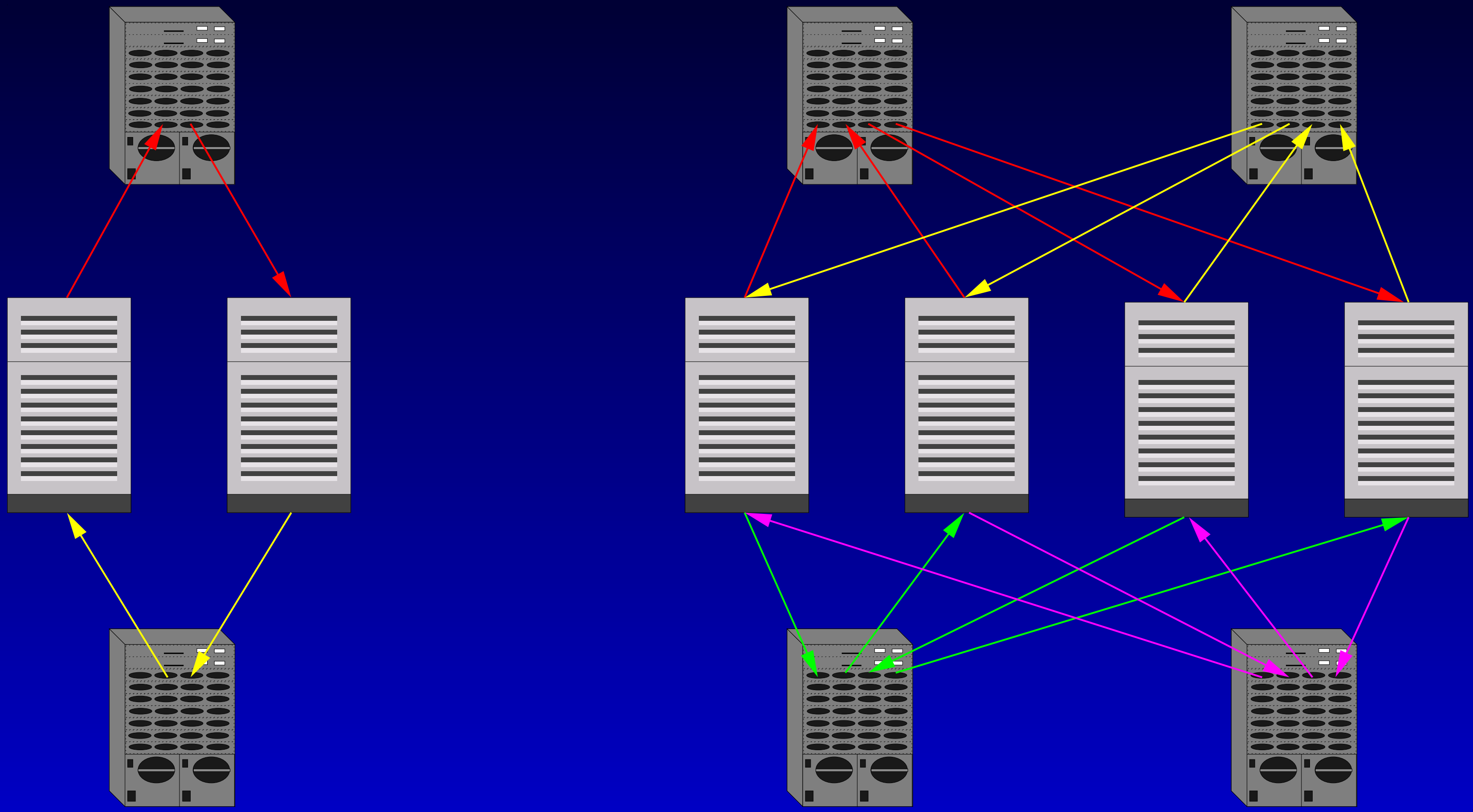


Static Rail Allocation Examples





Static Rail Allocation Examples



Los Alamos NATIONAL LABORATORY

Lower Bound with Static Rail Allocation

What is the minimal number of rails to connect a cluster if:

- Each node can only transmit or receive on a given rail
- Each node can transmit to any other node directly
- Rails are independent (messages stay on rail)

Lower Bound with Static Rail Allocation

What is the minimal number of rails to connect a cluster if:

- Each node can only transmit or receive on a given rail
- Each node can transmit to any other node directly
- Rails are independent (messages stay on rail)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

Los Alamos NATIONAL LABORATORY

Lower Bound with Static Rail Allocation

What is the minimal number of rails to connect a cluster if:

- Each node can only transmit or receive on a given rail
- Each node can transmit to any other node directly
- Rails are independent (messages stay on rail)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

This generalizes easily to an allocation with $n \leq 2^{\frac{r}{2}}$ rails

Los Alamos NATIONAL LABORATORY

Lower Bound with Static Rail Allocation

What is the minimal number of rails to connect a cluster if:

- Each node can only transmit or receive on a given rail
- Each node can transmit to any other node directly
- Rails are independent (messages stay on rail)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

This generalizes easily to an allocation with $n \leq 2^{\frac{r}{2}}$ rails
Is this optimal?

Lower Bound with Static Rail Allocation

Using binary matrix notation, what is the maximum number of columns n for r rows, so that for every two columns, a row exists with a '0' and another with a '1'?

Lower Bound with Static Rail Allocation

Using binary matrix notation, what is the maximum number of columns n for r rows, so that for every two columns, a row exists with a '0' and another with a '1'?

We can reduce this to Sperner's Lemma, to obtain:

Los Alamos NATIONAL LABORATORY

Lower Bound with Static Rail Allocation

Using binary matrix notation, what is the maximum number of columns n for r rows, so that for every two columns, a row exists with a ’0’ and another with a ’1’?

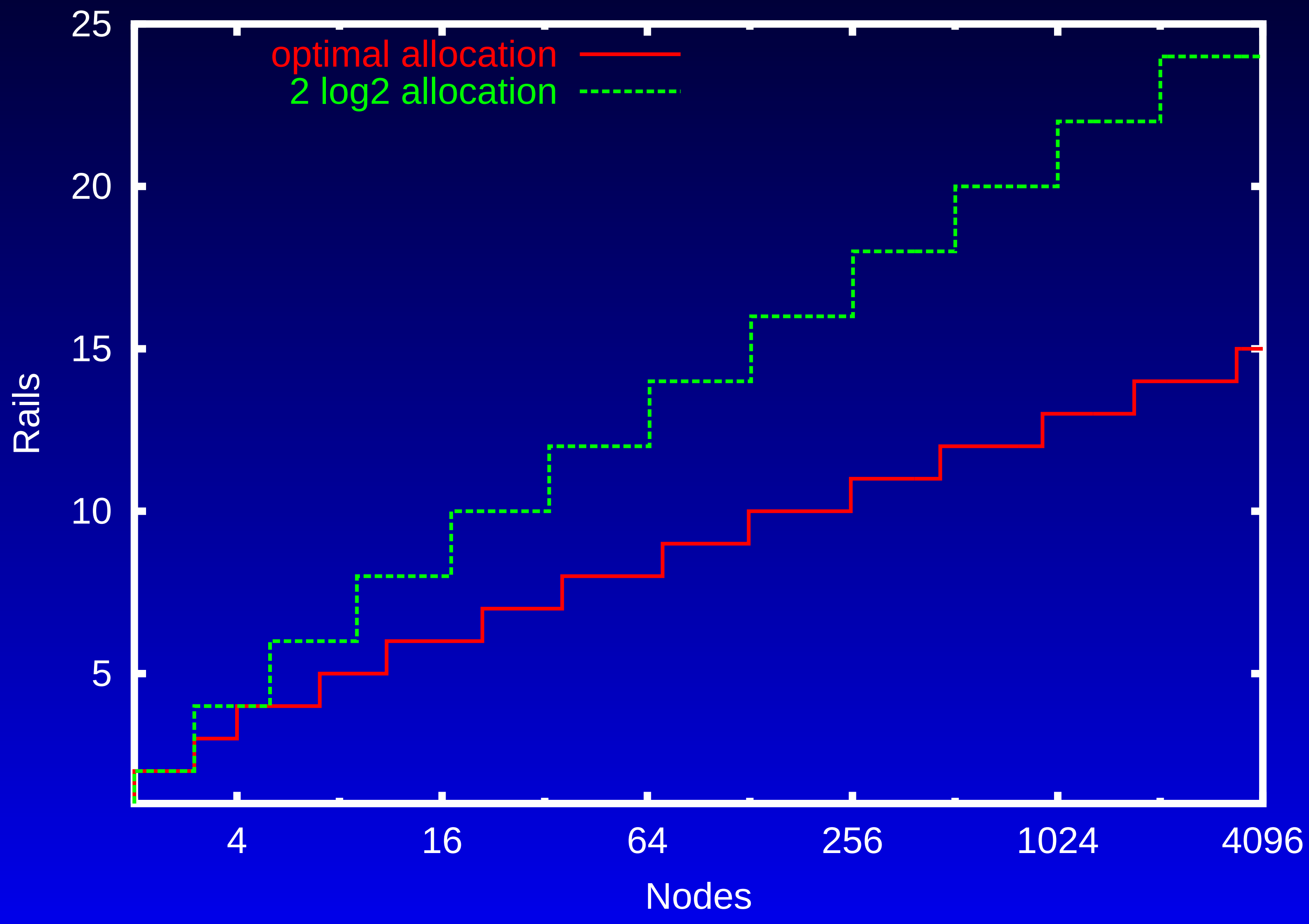
We can reduce this to Sperner’s Lemma, to obtain:

A network with *r* rails can support no more than *n* nodes, where

$$n \leq \binom{r}{\lfloor \frac{r}{2} \rfloor}$$

Los Alamos NATIONAL LABORATORY

Comparison of bounds



Dynamic Algorithm with Local Information

- With the dynamic algorithms the direction in which each network interface is used can change over time
- The *local-dynamic* algorithm allocates the rails in both directions, using local information available on the sender side
- Messages are sent over rails that not sending or receiving other messages
- Messages can be striped over multiple rails
- There is no guarantee that traffic will be unidirectional

Los Alamos NATIONAL LABORATORY

Dynamic Algorithm with Global Information

- The *dynamic* algorithm tries to reserve both end-points before sending a message
- In its core there is a sophisticated distributed algorithm that (1) ensures unidirectional traffic at both ends and (2) avoids deadlocks, potentially generated by multiple requests with a cyclic dependency

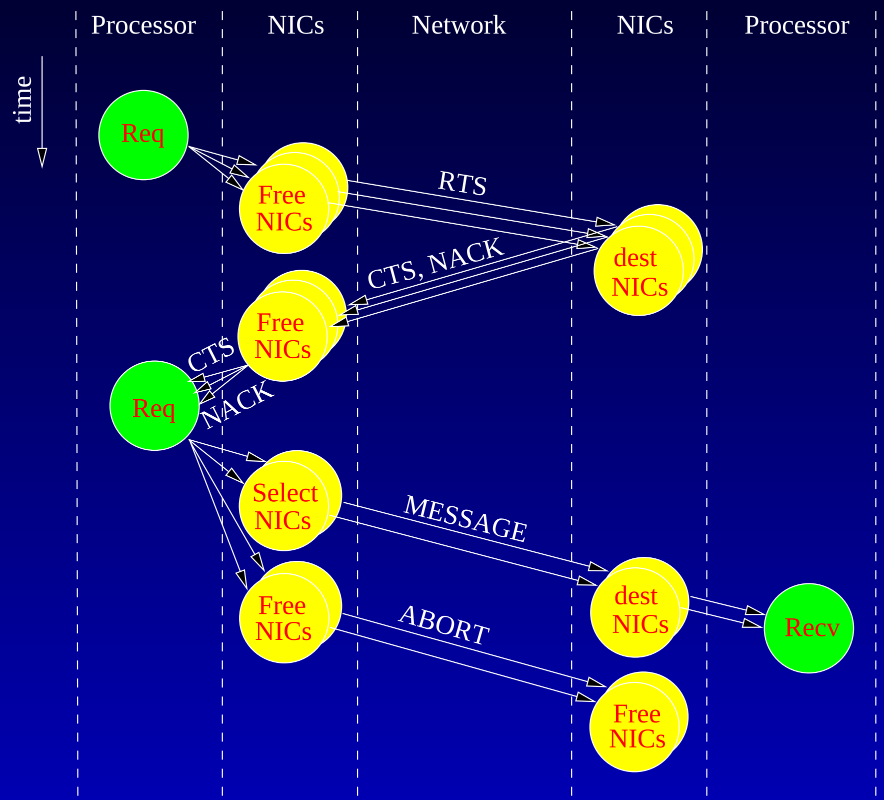
Los Alamos NATIONAL LABORATORY

Dynamic Algorithm: Implementation Issues

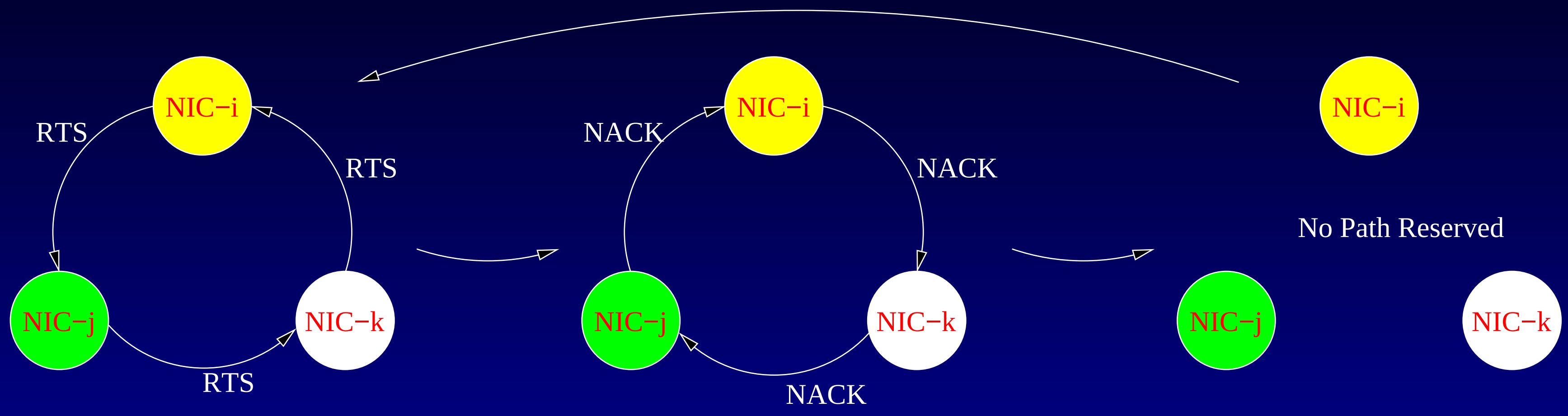
- The efficient implementation of this algorithm requires some processing power in the network interface, which needs to process control packets and perform the reservation protocol without interfering with the host
- For example, the Quadrics network interface is equipped with a thread processor that can process an incoming packet, do some basic processing and send a reply in as few as $2\mu s$

Los Alamos
NATIONAL LABORATORY

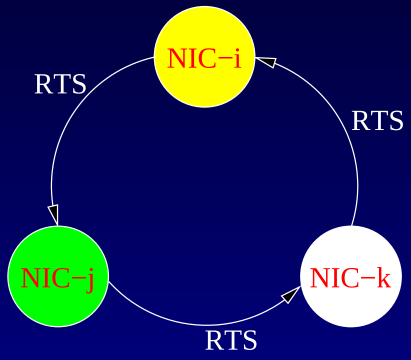
Dynamic Algorithm



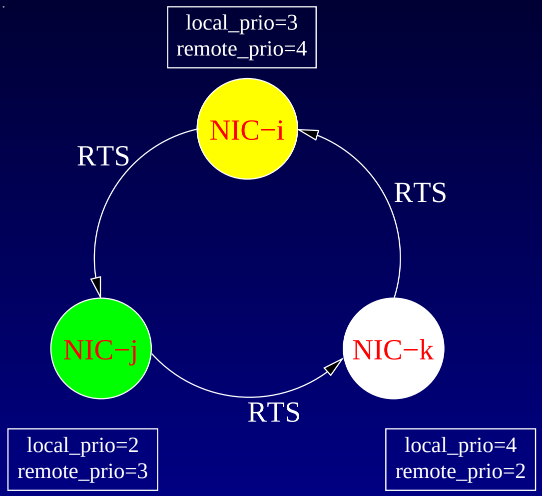
Deadlock (Livelock) in the Dynamic Algorithm



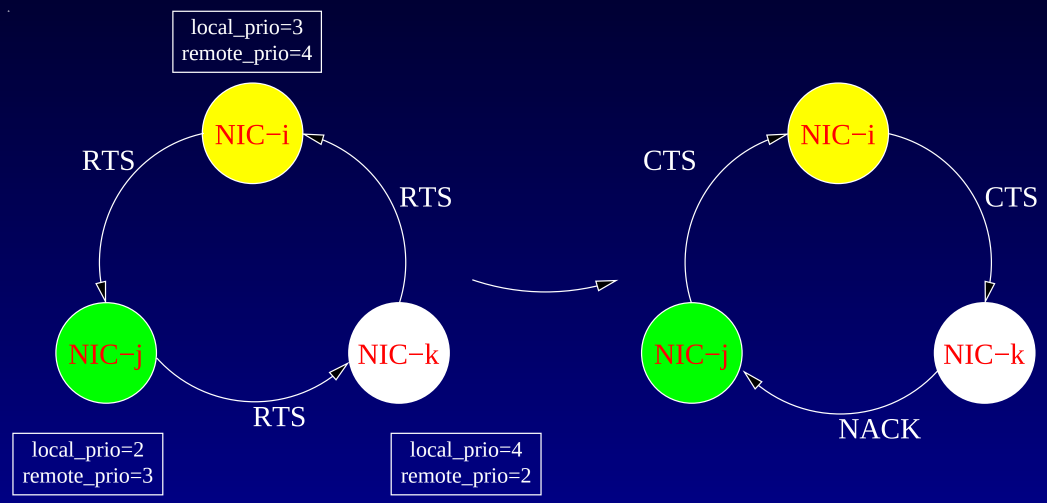
Deadlock Avoidance in the Dynamic Algorithm



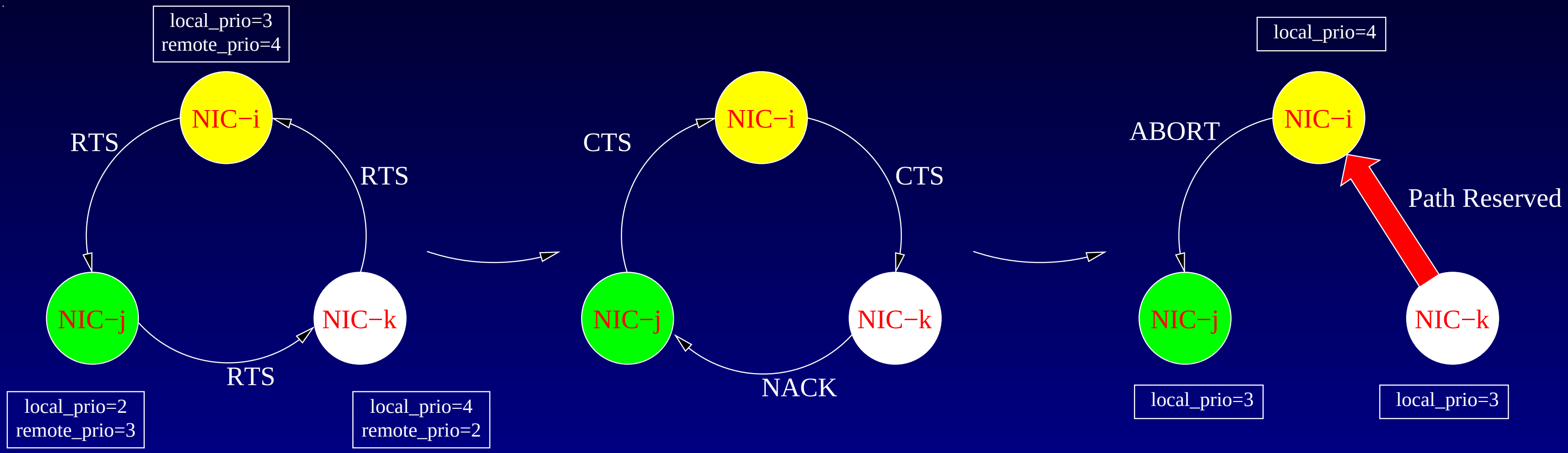
Deadlock Avoidance in the Dynamic Algorithm



Deadlock Avoidance in the Dynamic Algorithm



Deadlock Avoidance in the Dynamic Algorithm



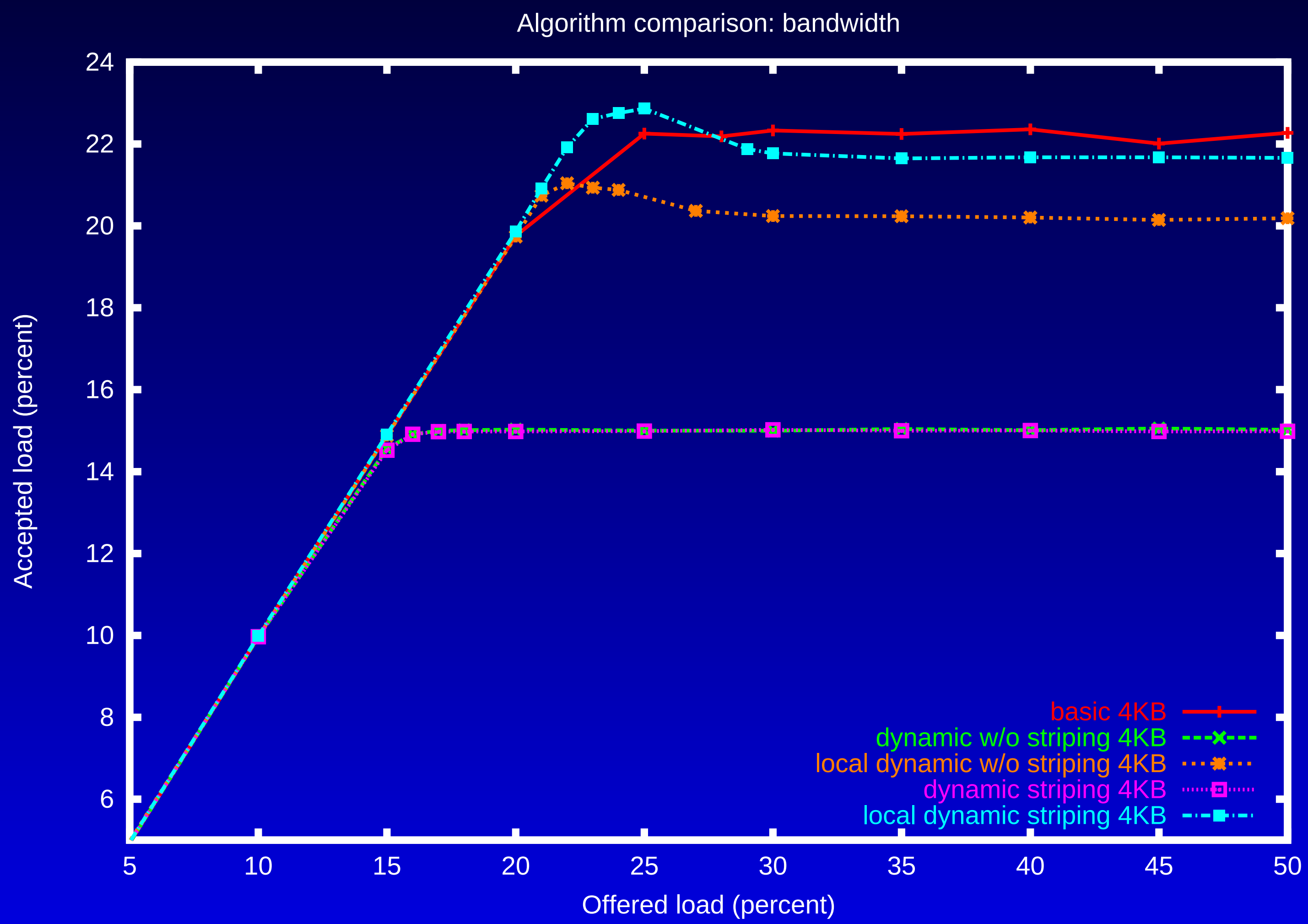
Simulation Framework

- Focus our attention on a family of fat-trees interconnection networks, ranging from 32 to 128 processing nodes (4-way SMPs)
- Up to 8 independent rails
- Low level simulation of the network (network model based on the Quadrics network)
- Simulate the communication processor in the NIC
- Test the network using a synthetic communication benchmark
- Exponential distribution for message size & interarrival time, destinations are chosen randomly (uniform)
- Measure two parameters, the overall *accepted bandwidth* and the message *latency*

Los Alamos
NATIONAL LABORATORY

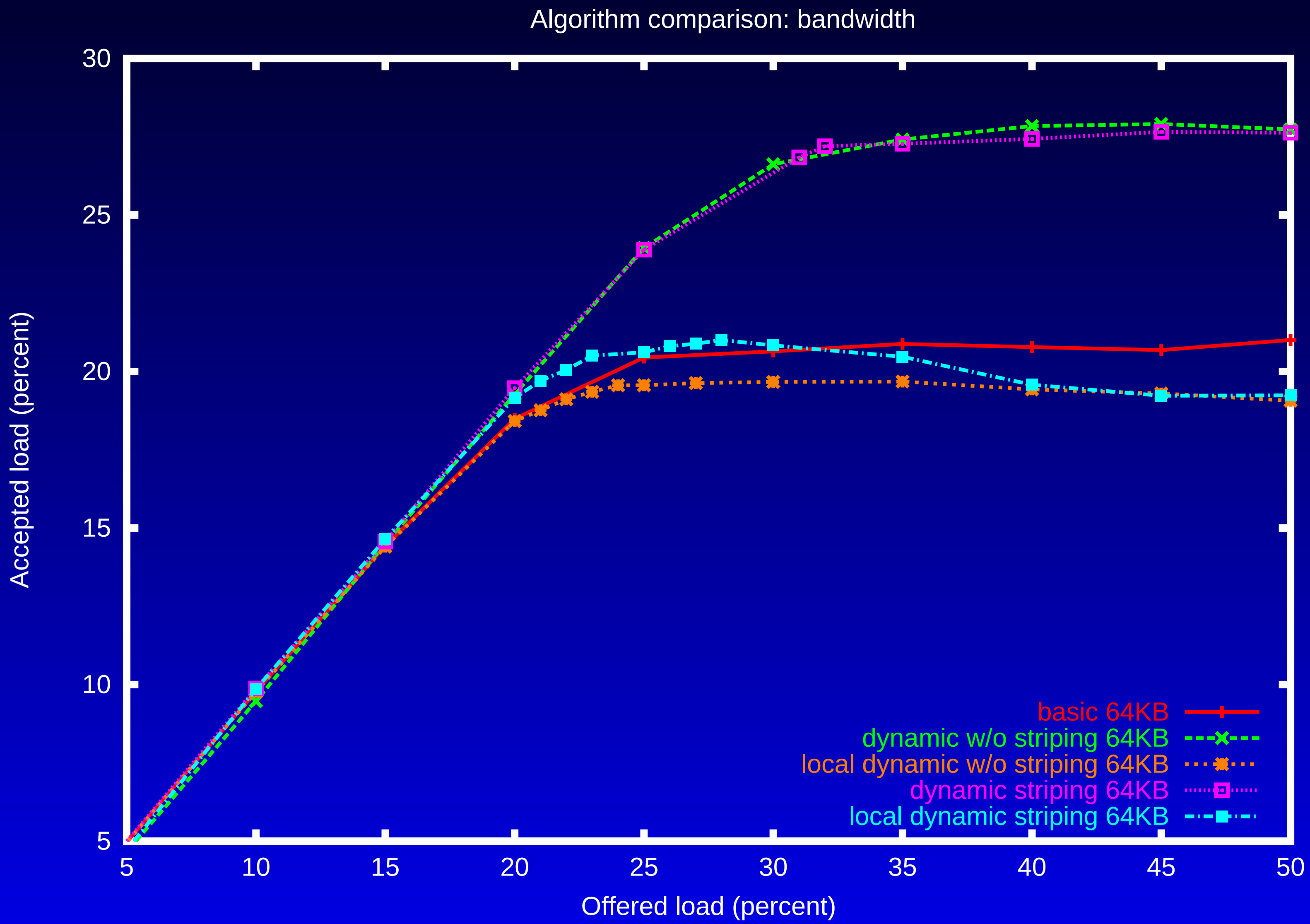
Results - Bandwidth

128 Nodes (4-way), four rails, 4KB average messages

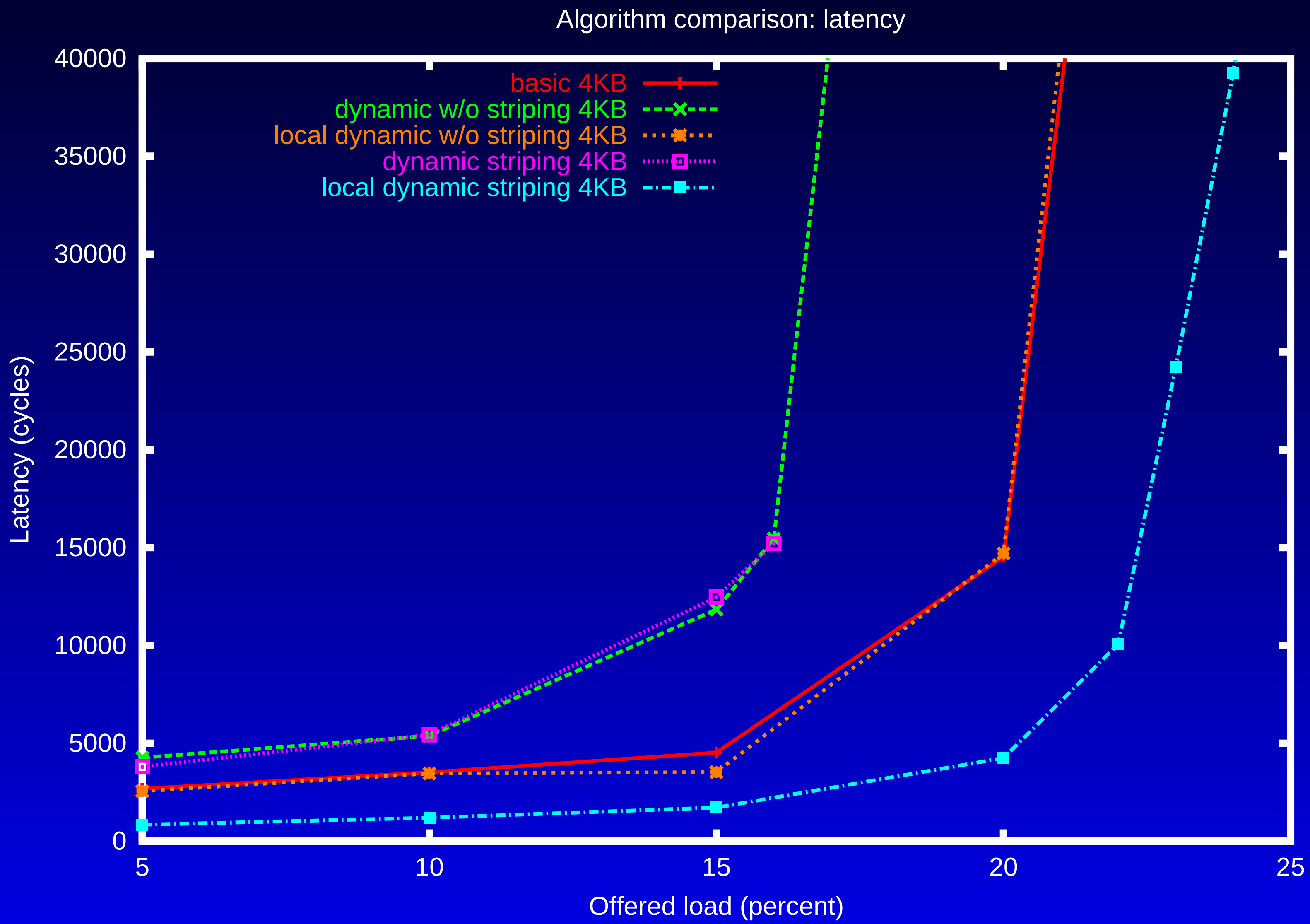


Los Alamos NATIONAL LABORATORY

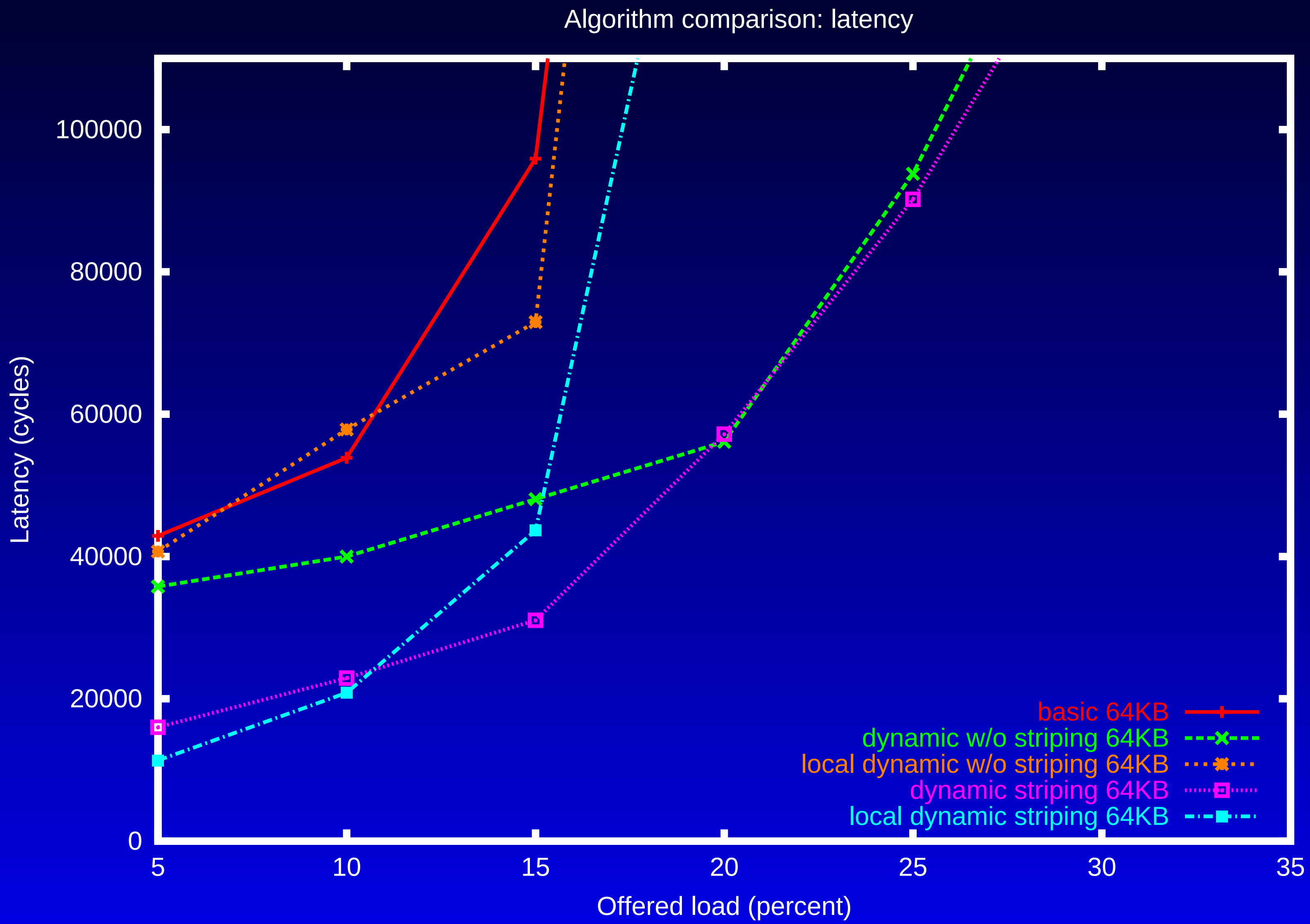
Bandwidth, 64KB Messages



Results - Latency, 4KB Messages

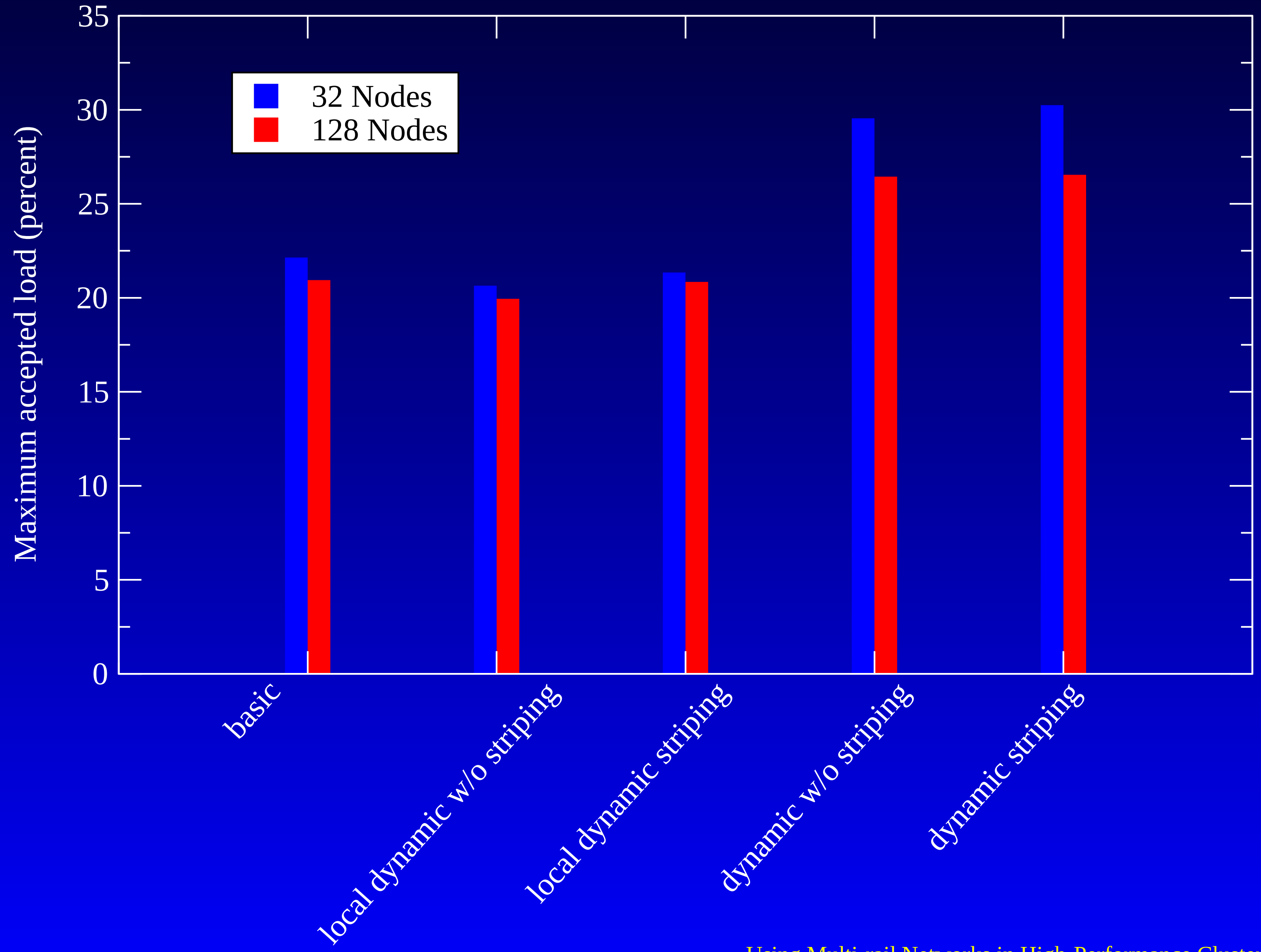


Latency, 64KB messages



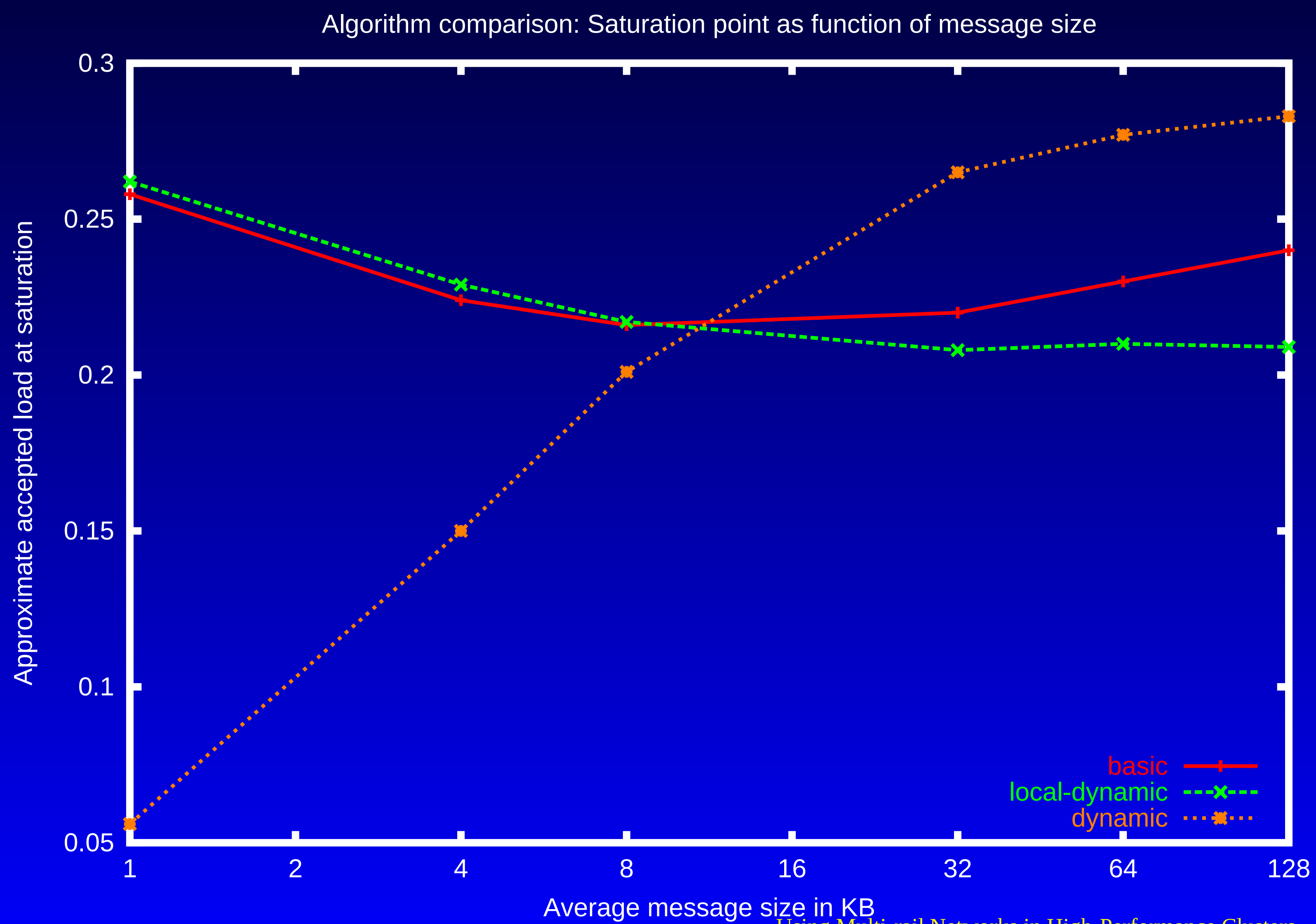
Maximum Accepted Load by Network Size

Using 4 rails and average message size of 32KB

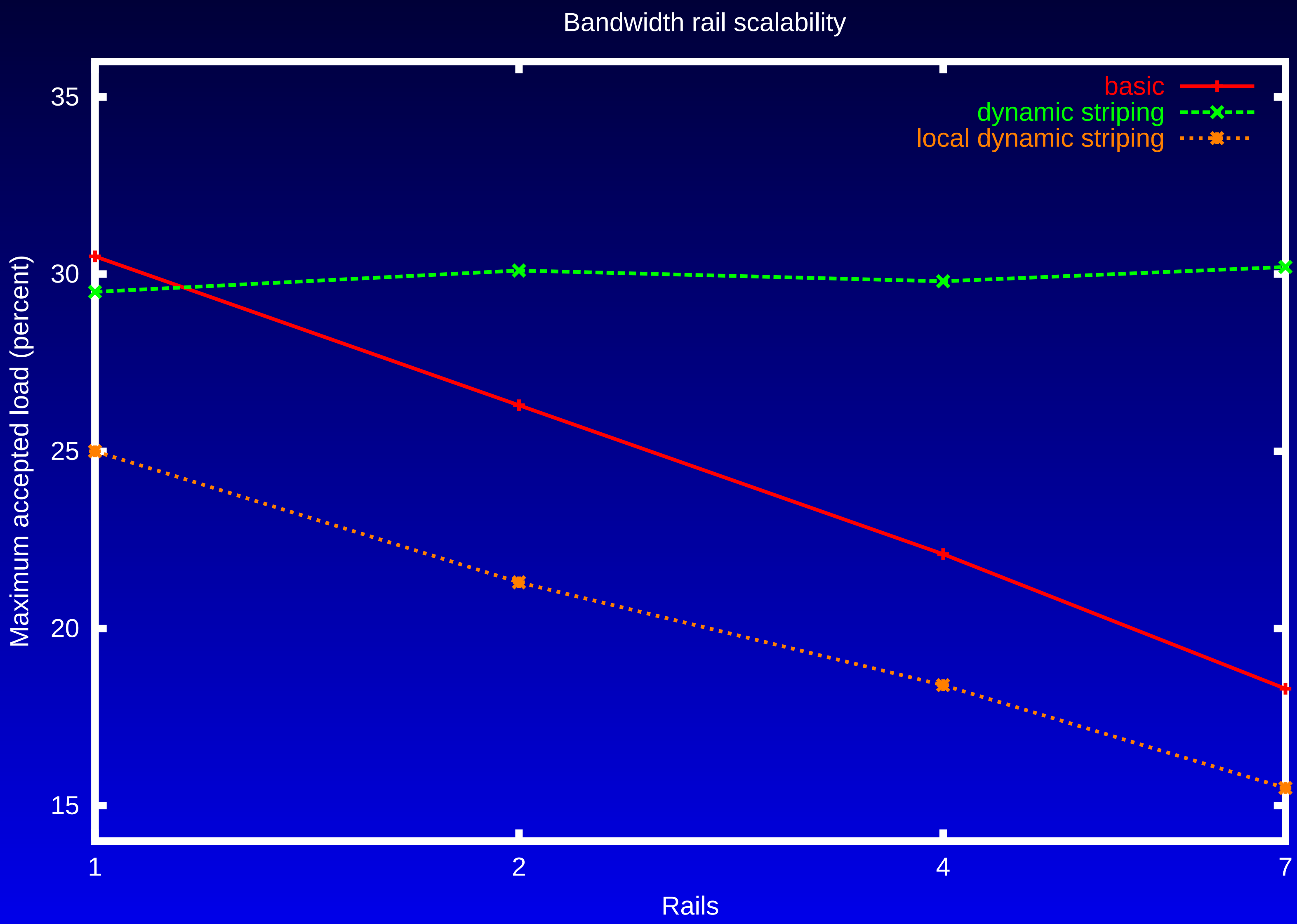


Saturation Points vs. Message Size

Striping and non-striping converge at high loads

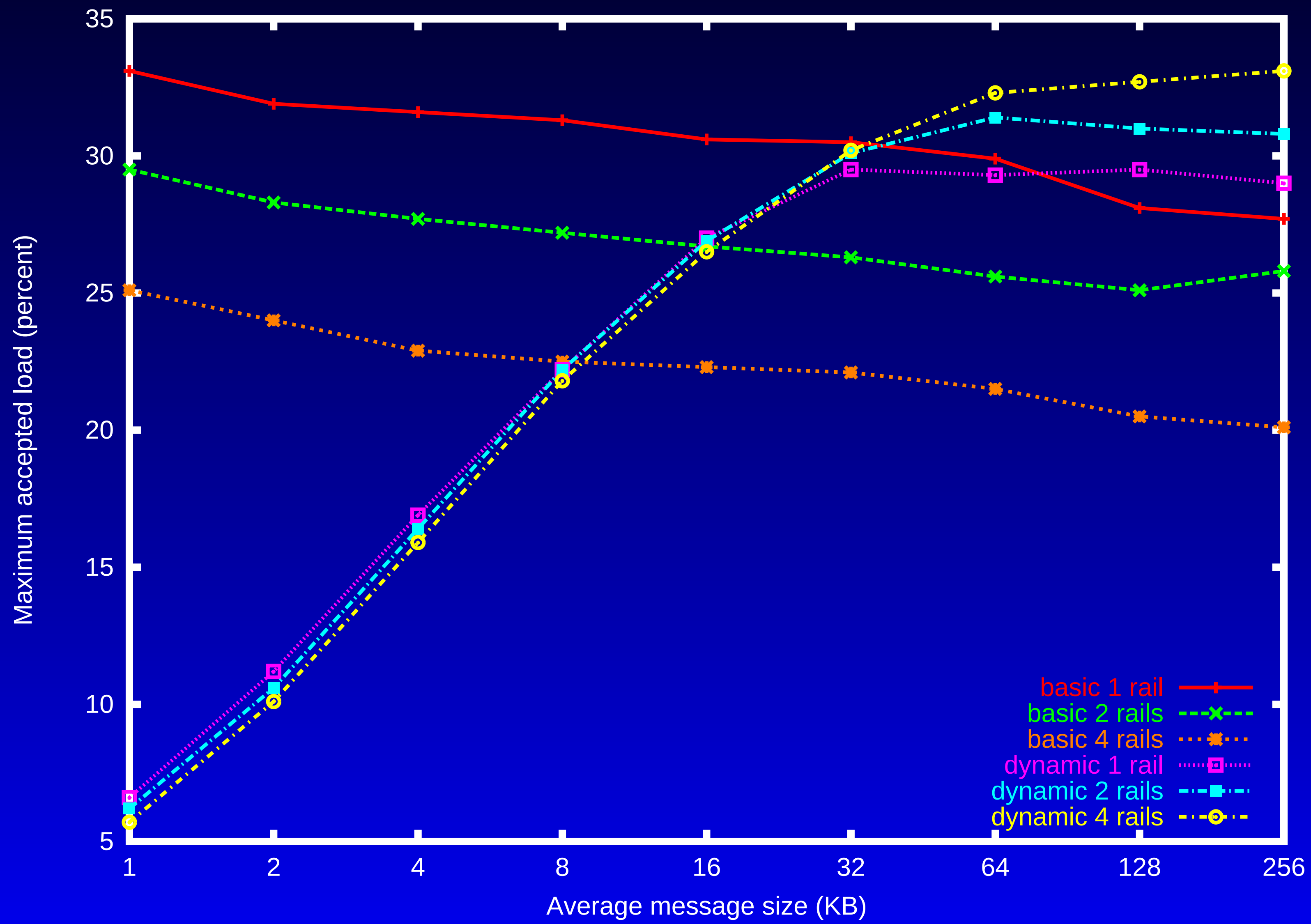


Bandwidth vs. Number of Rails (32 Nodes)



Los Alamos NATIONAL LABORATORY

Rail Scalability vs. Message Size



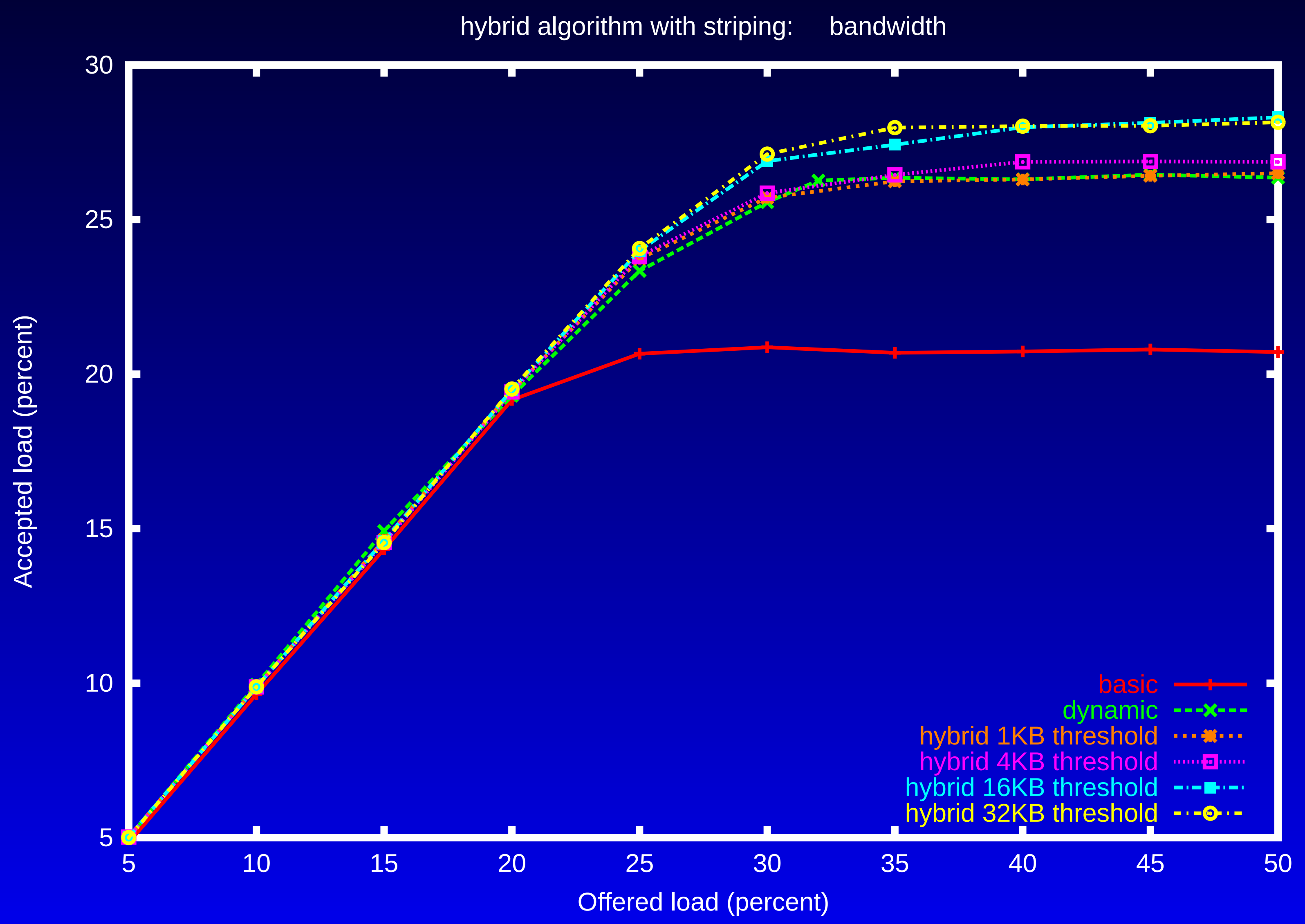
Los Alamos NATIONAL LABORATORY

Hybrid Algorithm

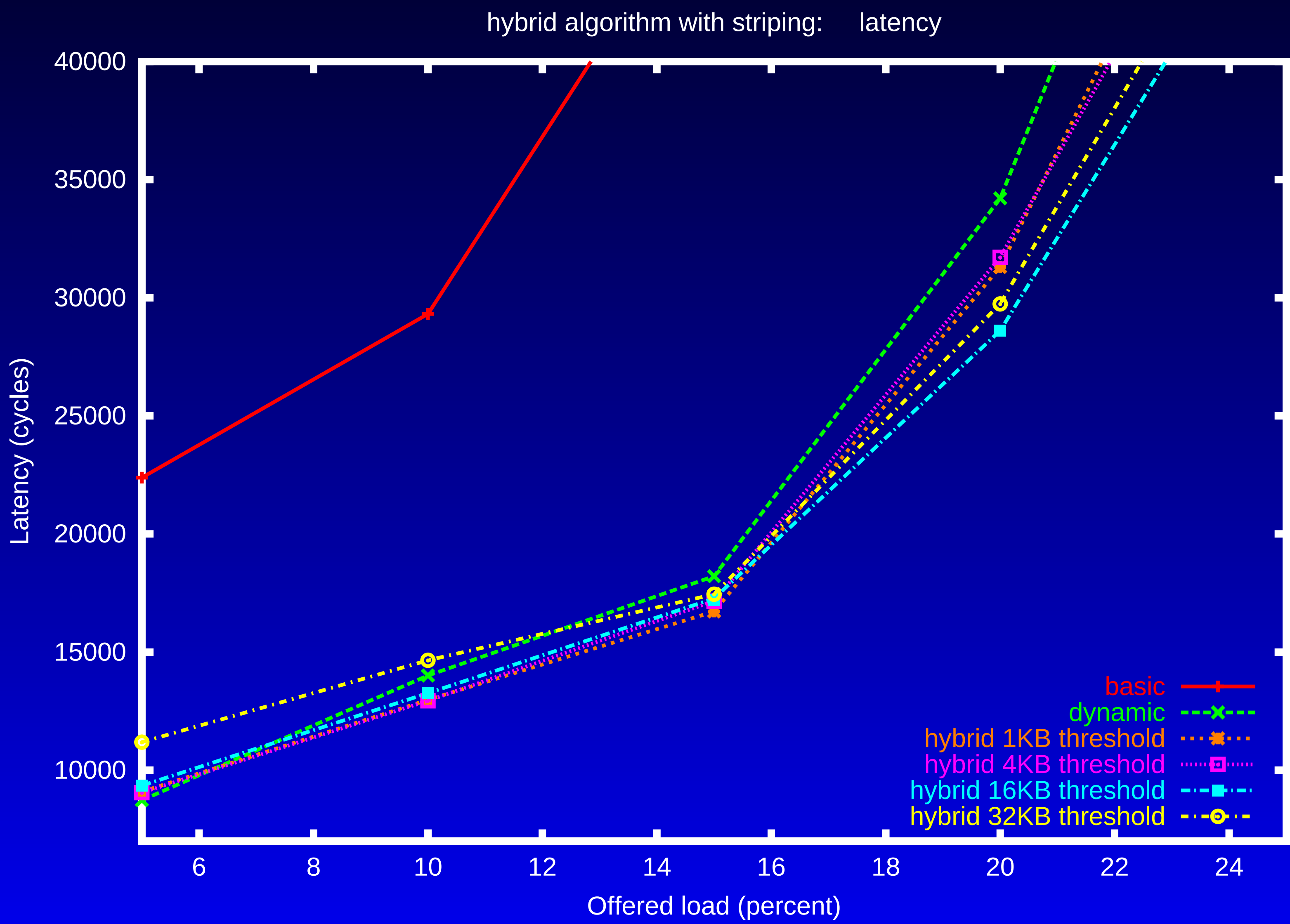
- The dynamic algorithm incurs a substantial overhead, for every message size.
- The hybrid algorithm sends short message without a reservation protocol
- Short messages are not striped
- It can cause bidirectional traffic for a short time
- We evaluate 128 nodes, 32KB average message size, 4 rails

Los Alamos NATIONAL LABORATORY

Hybrid, Bandwidth with Striping



Hybrid, Latency with Striping



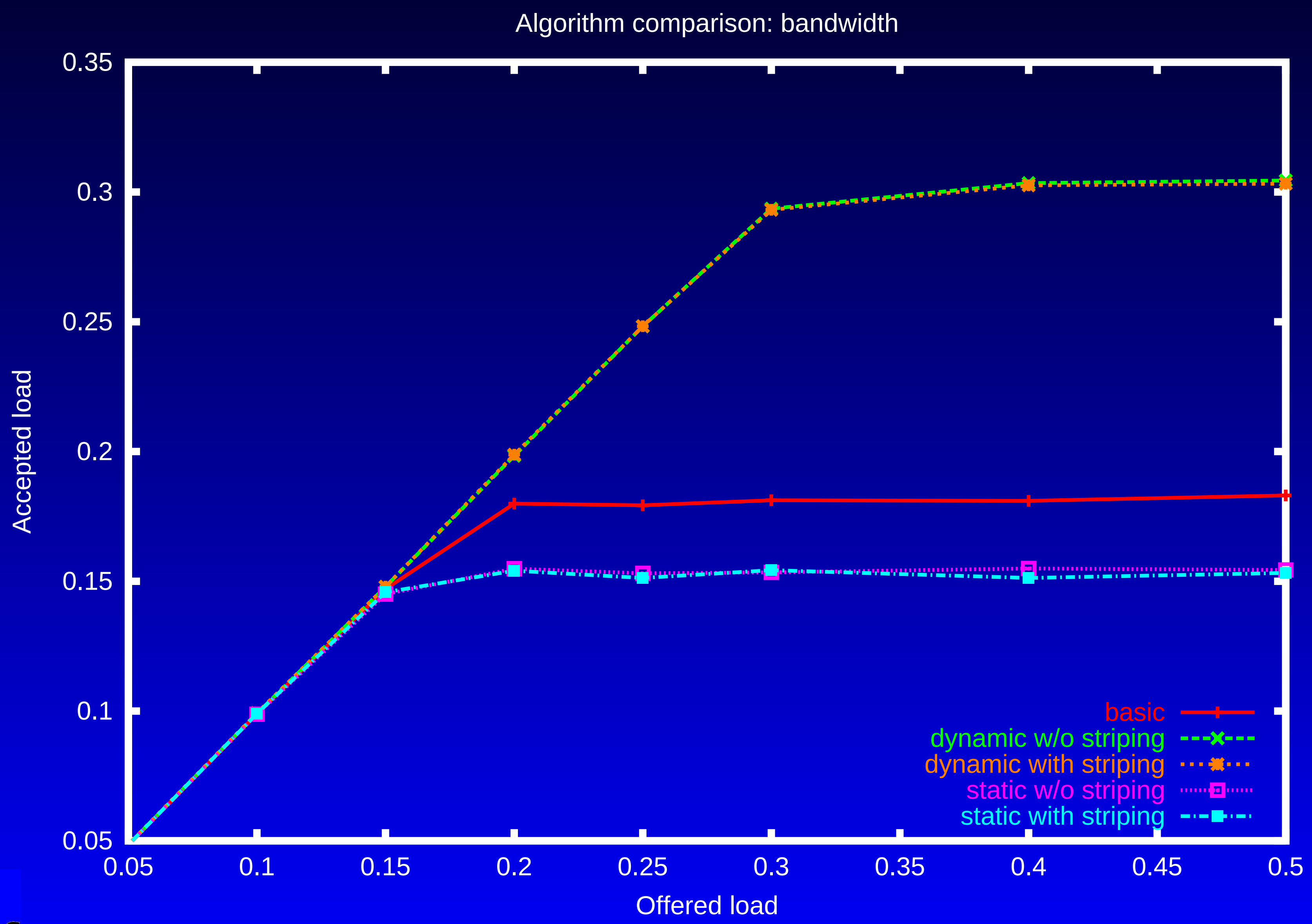
Los Alamos NATIONAL LABORATORY

Static Algorithm

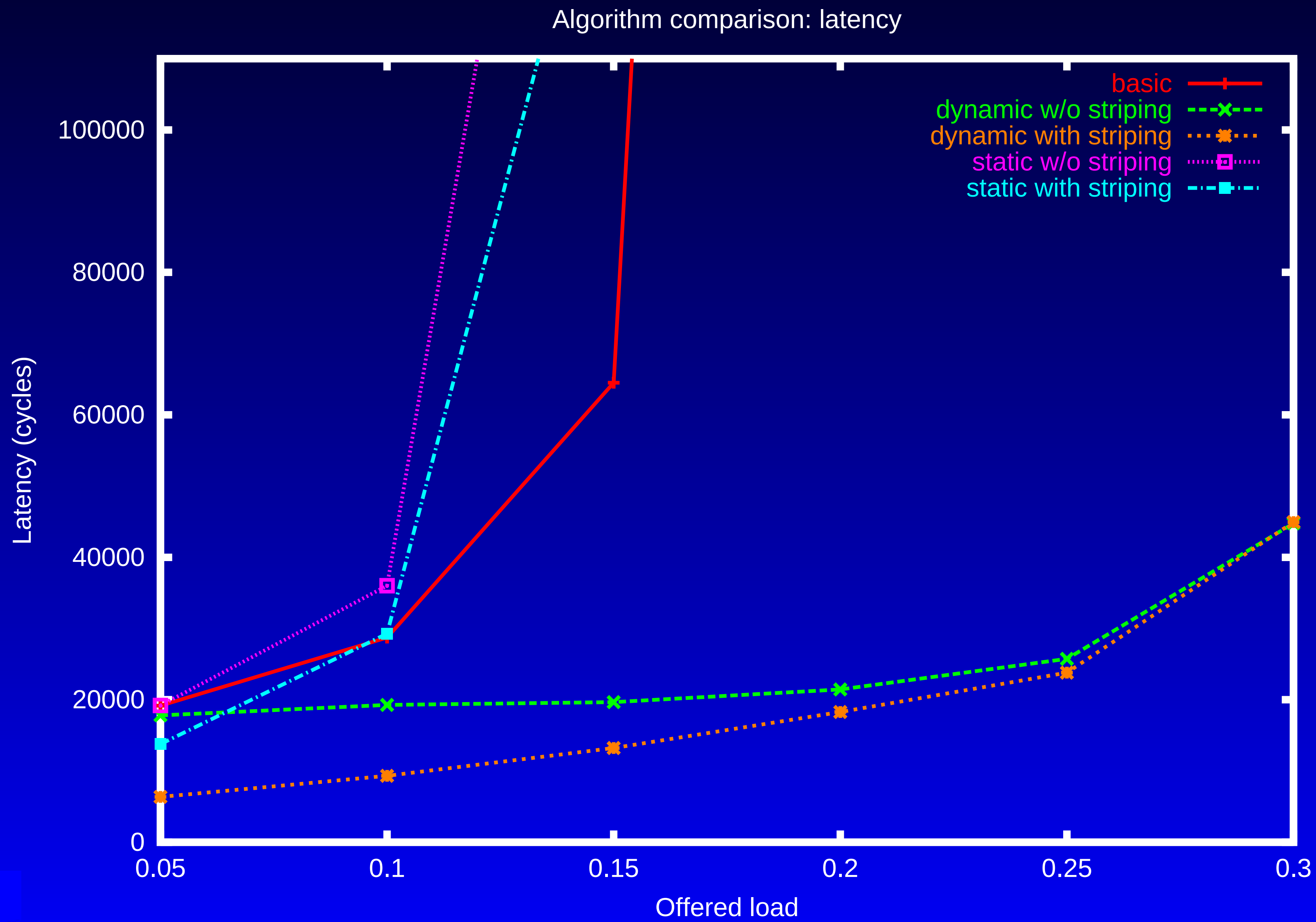
- For completeness, we also evaluated the static algorithm's performance, despite its high resource cost
- We use 32 nodes with 7 rails, 32KB average message size
- Outgoing messages have to compete over fewer rails: no bidirectional traffic, but more contention on the source

Los Alamos NATIONAL LABORATORY

Static Bandwidth



Static Latency



Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid

Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid
- The static algorithm requires a high number of rails, and performs quite poorly (but is easy to implemented)

Los Alamos
NATIONAL LABORATORY

Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid
- The static algorithm requires a high number of rails, and performs quite poorly (but is easy to implemented)
- A local-dynamic algorithm can be used to improve rail allocation over round-robin, with relatively low overhead

Los Alamos NATIONAL LABORATORY

Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid
- The static algorithm requires a high number of rails, and performs quite poorly (but is easy to implemented)
- A local-dynamic algorithm can be used to improve rail allocation over round-robin, with relatively low overhead
- The dynamic algorithm performs relatively well for relatively large message sizes

Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid
- The static algorithm requires a high number of rails, and performs quite poorly (but is easy to implemented)
- A local-dynamic algorithm can be used to improve rail allocation over round-robin, with relatively low overhead
- The dynamic algorithm performs relatively well for relatively large message sizes
- This algorithm is scalable with the number of rails

Los Alamos NATIONAL LABORATORY

Conclusion

- We presented several algorithms to allocate multiple rails, static, dynamic and hybrid
- The static algorithm requires a high number of rails, and performs quite poorly (but is easy to implemented)
- A local-dynamic algorithm can be used to improve rail allocation over round-robin, with relatively low overhead
- The dynamic algorithm performs relatively well for relatively large message sizes
- This algorithm is scalable with the number of rails
- Incorporating protocol-free short message handling in the hybrid algorithm furtherly increases performance of the dynamic algorithm

Resources

- More information can be found at or

http://www.cs.huji.ac.il/~etcs

- Or by sending an email to

eitanf@lanl.gov