Hybrid PDES Simulation of HPC Networks Using Zombie Packets

Elkin Cruz-Camacho
Rensselaer Polytechnic Institute
Troy, New York, USA
cruzce@rpi.edu

Kevin A. Brown
Argonne National Laboratory
Lemont, Illinois, USA
kabrown@anl.gov

Xin Wang
Illinois Institute of Technology
Chicago, Illinois, USA
xwang149@hawk.iit.edu

Xiongxiao Xu
Illinois Institute of Technology
Chicago, Illinois, USA
xxu85@hawk.iit.edu

Kai Shu
Illinois Institute of Technology
Chicago, Illinois, USA
kshu@iit.edu

Zhiling Lan
Illinois Institute of Technology
Chicago, Illinois, USA
lan@iit.edu

Robert B. Ross
Argonne National Laboratory
Lemont, Illinois, USA
rross@mcs.anl.gov

Christopher D. Carothers
Rensselaer Polytechnic Institute
Troy, New York, USA
chrisc@cs.rpi.edu

ABSTRACT

High-fidelity network simulations provide insights into new realms for high-performance computing (HPC) architectures, although at a high cost. Surrogate models offer a significant reduction in runtime, yet they cannot serve as complete replacements and should be only used when appropriate. Thus the need for hybrid modeling, where high-fidelity simulation and surrogates run side-by-side. We present a surrogate model for HPC networks in which packets bypass the network, and the network state itself is suspended when switching to the surrogate. To bypass the network, every packet is scheduled to arrive at a predicted time in the future estimated from historical data; to suspend the network, all in-flight packets are delivered to their destinations, but they are kept in the system to awaken as zombies when switching back to high-fidelity. Speedup for a hybrid model is relative to the proportion of surrogate to high-fidelity. We obtained a 3× speedup for a simulation where 70% of virtual time was spent in surrogate mode. When considering the surrogate portion only, the speedup jumps to nearly 20× on a uniform random network traffic example. The accuracy of the overall simulation increased when the network state was suspended instead of ignored, which demonstrates the need for modeling the network state when transitioning from surrogate back to high-fidelity mode.

CCS CONCEPTS

• **Networks → Network simulations**; **Network performance modeling**; • **Computing methodologies → Parallel computing methodologies**.

KEYWORDS

Parallel Discrete-event Simulation, HPC networks, Surrogate simulation

ACM Reference Format:
Elkin Cruz-Camacho, Kevin A. Brown, Xin Wang, Xiongxiao Xu, Kai Shu, Zhiling Lan, Robert B. Ross, and Christopher D. Carothers. 2023. Hybrid PDES Simulation of HPC Networks Using Zombie Packets. In *ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM-PADS '23), June 21–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3573900.3591122

1 INTRODUCTION

Parallel discrete event simulations (PDES) are used to accurately model complex HPC networks and drive important co-design studies for emerging systems [2, 8, 11, 14]. However, simulating a single millisecond of network traffic in exquisite detail may well take hours, even for static, well-behaved traffic patterns such as uniform random. Skipping ahead in the simulation is a fundamental step in cutting down simulation time, yet it is also crucial to keep the network state as consistent and accurate as possible.

We have extended CODES [12] and its underlying PDES framework, ROSS [1], with the ability to pause a simulation at an arbitrary point in time on parallel execution. During this pause, we switch the underlying behavior of the simulation from high-fidelity packet routing into a faster, coarser-grain surrogate simulation. Instead of simulating all network activities, the surrogate predicts how long a packet will take to arrive at its destination and schedules its arrival for the predicted time.

To ensure that the accuracy of the overall simulation is maintained upon the restarting of the high-fidelity simulation, we suspend the network state when switching to surrogate mode and later we re-animate it on the switch back to high-fidelity. Any in-flight packet in the network at the switch is frozen in place until re-animation. Frozen in-flight packets are tagged as *zombies* while copies of each one are sent to their destinations bypassing the routers. Reanimated zombie packets are never delievered to their destination computing nodes, yet they are treated as normal packets by the routers as they hop across the network. In parallel optimistic simulations, our suspension strategy requires working around optimistic executions and other optimizations for parallelization.

We demonstrate our approach on a balanced dragonfly network congested with uniform random traffic, and show its capabilities through a comparison using buffer occupancy and packet latency changes. The speedup for the surrogate portion of the simulation was 19.7×, while the speedup of overall the simulation was 2.98× with a surrogate to a high-fidelity ratio of 7:3. We compare a hybrid simulation where the network is re-animated upon restart to a hybrid simulation where the network is not re-animated, instead, the network is vacated. Re-animating the network with zombie packets showed an improvement in the MSE (mean squared error) from 1058us^2 to 145.8us^2. Thus, the network accuracy was significantly higher, *i.e*, the hybrid model with zombie packets was more representative of the baseline high-fidelity simulation.

2 BACKGROUND AND RELATED WORK

2.1 HPC Network Traffic

HPC application workloads running on different nodes use the network to exchange information and coordinate. Workload messages are sent as network packets with help of terminals, which split the packets into *flits*. These flits are injected into the network via terminals into routers; then, they travel across the network from router to router until their destination, where they are delivered to the computing node by a terminal. A packet, for example, might be of size 4096 bytes, while flits are often small and in the order of dozens of bytes (*e.g*, 64 bytes). The time that takes to move a packet from one end to another is called end-to-end packet latency. This time is dependent on how many other packets/flits are already in the network, called in-flight packets. In certain common scenarios, the network can enter a steady state where packet latency and the number of in-flight packets are relatively stable. It is under these steady states that we can implement a surrogate model able to accurately approximate the stable behavior of the network.

2.2 Accelerating Discrete Event Simulations

Various multi-resolution and hybrid PDES models have been proposed to accelerate high-fidelity PDES simulations. Liu [9] and Gu et al. [5] demonstrated one such approach by combining a fluid flow model with an event-based model to simulate network traffic. Both models predict the hop-by-hop activity of the flows and packets as well as the updated buffer states at each hop. He et al. [6] switches between an end-to-end TCP flow model and a packet-based discrete event model for simple TCP networks. To ensure the discrete event model remains statistically accurate throughout the simulation, packet buffers are checkpointed (frozen) when switching to the flow model, and buffers are restored when switching back to the event-based model. While this approach successfully maintains statistically accurate buffer states, the work does not explore the challenges in the context of parallelized simulation execution, which is required for simulating large-scale HPC networks.

3 HYBRID MODELING USING *ZOMBIE* EVENTS

We used the CODES/ROSS simulation toolkit [12] to demonstrate the feasibility of our hybrid modeling approach for HPC network simulations. CODES supports high-fidelity flit-level network simulations that enable investigating low-level activities such as adaptive adapting routing [7] and quality of service [2]. A single millisecond of network traffic can take hours to simulate in high-fidelity mode. Our goal is to train a model on-the-fly to approximate the

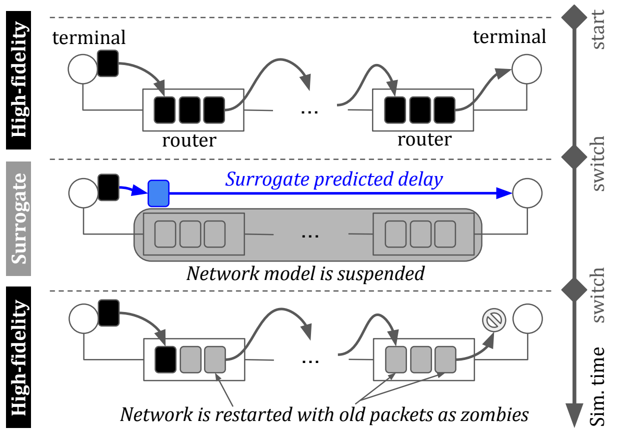
High-fidelity
terminal
terminal
router
router
start
Surrogate
Surrogate predicted delay
Network model is suspended
switch
High-fidelity
switch
Network is restarted with old packets as zombies
Sim. time

Figure 1: Hybrid simulation's modeling phases.

behavior of the network such that it can replace the slower PDES network model with an accurate, lightweight surrogate. To this end, we extended both ROSS and CODES to pause and continue the simulation at user-defined timestamps. During these periods, the underlying network routing model is swapped for a surrogate and back.

Our surrogate bypasses the network routers. A terminal behaves differently depending on the simulation mode: high-fidelity or surrogate. When we switch to surrogate mode, the terminal does not inject the packet into the network, but instead, it asks a *predictor* to estimate the end-to-end latency for the packet and schedules its arrival at its destination. This switch from high-fidelity to surrogate mode and back to high-fidelity mode is shown in Figure 1. During surrogate mode, the predicted packet can be seen bypassing the network. Switching back to high-fidelity mode requires only switching how the terminal handles packets so that it can inject them into the network again. Notice that if we ignore the in-flight packets when we switch to surrogate mode, the routers will continue their journey, moving packets to their destinations and eventually vacating the network. If this happens, the network will be empty when we switch back to high-fidelity mode. In some cases, like when the network is barely utilized, this will not be a problem. However, if the network is *hot*, *i.e*, congested with packets in-flight, we have to keep the network *hot* on restart in order to preserve high-fidelity model accuracy. This is accomplished by suspending the network model state and re-animating it on the switch back. To make the hybrid modeling accurate, we propose the following solutions to the challenges that parallel hybrid modeling for networks presents.

Challenge 1. Switching parallel simulation modes. Under the paradigm of optimistic parallel simulation, which coordinates a simulation across a pool of computing nodes, all nodes run their portion of the simulation for a given period of time or steps, which often results in them advancing farther ahead than they should. When this happens, the nodes have to rollback to a common agreed virtual time at which all the nodes can definitely say the simulation has progressed, called global virtual time (GVT). We take advantage of this consistent global state point (at GVT) to trigger a director function in charge of deciding when to switch and enforcing such a switch. We call this function within the optimistic loop after finding GVT, a point where events have been rolled back. The director function informs the terminal of the current simulation mode via a global boolean variable.

To make our approach useful for a range of PDES use cases, we can trigger any arbitrary function at GVT and not only a director function. Our implementation is flexible and can be used for any other task that requires a consistent state across the whole simulation model. It can be used for example, as a starting point for checkpoint restart of the simulation, gather statistics on PDES, and, even, switch global PDES parameters on-the-fly to influence its progress such as restricting optimistic execution [13], batch event processing size, or resize the preallocated space for events.

Challenge 2. Packet-latency predictor. To bypass the network, we need a predictor to estimate packet latency. This function can be virtually any function that returns positive real numbers. We chose a classical statistical measure to predict future behavior: the average of the packet latency to a specific destination terminal. Each terminal is informed of the latency of all packets that have been delivered via a custom notification event. The terminal aggregates this information for each destination terminal to which it has sent packets. The packet latency of a new packet to transmit in surrogate mode is a look-up table. Within CODES, packets start their journey in a *generate* function handler. Surrogate and high-fidelity code are separated inside this function by a single *if* statement. Solving these two challenges (1 and 2) leaves us with our first complete surrogate strategy, which we nickname: “hybrid-lite”.

Challenge 3. Network suspension. Hybrid-lite suffers from low network accuracy due to ignoring the network state and letting it cool it down upon restart. To suspend the network, we require a more involved director function — one that is capable of moving events for in-flight packets into the future and handle the intricacies of the PDES engine. Particularly, we need to make sure the state of the simulation can be altered, which requires rolling all events back to GVT. This involves enforcing a user-directed rollback, or user-level rollback [4]. Next, we will focus on two additional challenges presented by suspending the network.

Challenge 4. Process in-flight events. When the network is suspended, in-flight packets have to be delivered to their destinations to guarantee the correct progression of the simulation. To determine what packets are in-flight, we have two alternatives: keep a tally of which events have been sent or inspect every router for the in-flight events. The former methodology is far simpler but not massively scalable. The latter requires each router to trigger an event to the source terminal for it to handle it. Either way, once the terminal knows of an in-flight packet, the terminal predicts its packet latency and schedules it to its destination.

Challenge 5. Prevent duplicated packets—zombifying. Because the state of the system has been suspended, all in-flight packets will be re-animated on the switch back. All re-animated packets/flits must behave like any other packet/flit so that the network keeps hot, except that they cannot be delivered as it would make for duplicated packets. To prevent this, packets are tagged as zombies and are discarded on arrival at a terminal. We call this strategy simply “hybrid”.

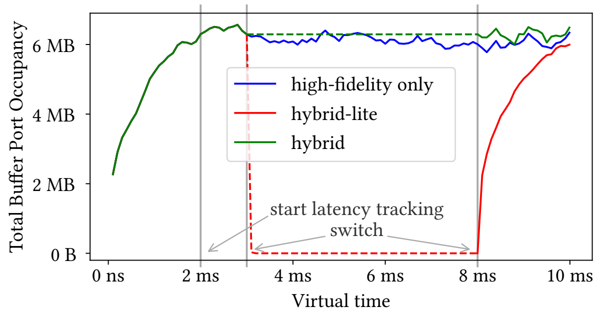

Figure 2: Aggregated router port buffer occupancy for uniform random traffic pattern.

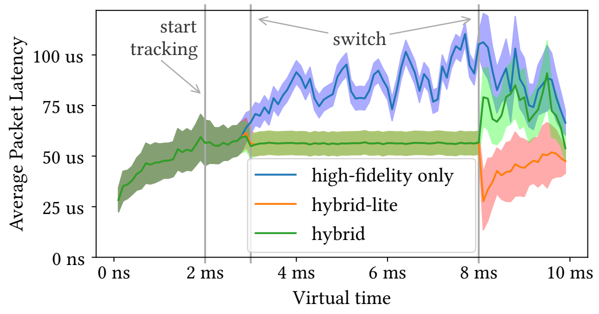

Figure 3: Average global packet latency with a window of 100**us for uniform random traffic. Error bars represent** 10% **of the standard deviation.**

Implementation Validation. To validate hybrid CODES, we compare it to the execution of the native high-fidelity only CODES simulation and observe the high-level behavior of the models. We use a simple ping-pong pattern, where the simulation will only advance if a ping or pong message is delivered, and the simulation will halt prematurely otherwise. In all our tests, the number of pings sent and pongs received was the same for all nodes without ever halting the hybrid simulation.

Another test of integrity is deterministic model execution. ROSS and CODES have previously been enhanced with a tie-breaker mechanism that allows for deterministic executions of the same experiment [10]. This means that running the same experiment multiple times should produce the same result. Otherwise, there is a potential problem with the event rollback mechanism. In our validation efforts, we found no differences in the net number of events processed or other metrics between the high-fidelity only and hybrid simulations.

4 EXPERIMENTS AND DISCUSSION

Experimental setup. We simulated a 72-node 1D dragonfly network with two nodes per router, four routers per group, nine groups, and two global channels per router. Each pair in a group is connected using one global channel. Runs were configured with 64 bytes chunks (equivalent to flits), 4096 bytes packets, and 2 GB/s link bandwidths. All experiments were conducted on a single node on the AiMOS

Table 1: Validating against high-fidelity only for the Uniform Random application pattern. Parallel run on 9 cores.

Total Virtual Time	Virtual Time Tracking Latencies	Virtual time in Surrogate	Mode	Throughput (GB/s)	Throughput discrepancy	Total Wall-time	MSE
10 ms	1 ms	5 ms	High-fidelity only	138.0	-	157.6 s	-
			Hybrid-lite	141.3	+1.024%	83.5 s	1966 us^2
			Hybrid	140.0	+1.015%	83.9 s	207.2 us^2
100 ms	10 ms	70 ms	High-fidelity only	141.1	-	1649.3 s	-
			Hybrid-lite	143.0	+1.013%	558.6 s	1058 us^2
			Hybrid	143.9	+1.012%	553.4 s	145.8 us^2

supercomputer at CCI (Center for Computational Innovations) [3]. A node on AiMOS consists of a 20-core IBM Power9 processor with a 3.15 GHz clock and 512GiB of RAM.

Methodology. For each set of experiments, three simulation modes were explored: high-fidelity only, hybrid-lite, and hybrid. We use the uniform random pattern, where nodes send each message to a random destination. Each node injects 1024-byte messages at 100% the injection rate, ensuring the network is congested. The simulations are run for 10 and 100 virtual ms. For the 10 ms hybrid simulation, the first 3 ms and last 2 ms are run in high-fidelity mode, while the corresponding period in the 100 ms hybrid run are the first 20 ms and the final 10 ms. Terminals track packet latencies for 1 ms before the switching to the surrogate mode in the 10 ms case and for 20 ms before the switch in the 100 ms case.

Results. Figure 2 shows the change in buffer occupancy as time progresses, where the occupancy measurement is the summation of data queued in port buffers on all routers in the system. For the hybrid case, suspending the network allows it to restart hot, *i.e*, the network starts from a loaded state. Figure 3 shows the impact of network load on the packet latency performance. For the hybrid-lite case, the packet latency drops significantly when the network restarts empty at 8 ms. However, the packet latency for the hybrid case is closely aligned with the baseline high-fidelity only case.

Table 1 shows the different experiment configurations, simulated throughput, and simulation runtime. For the first case, 10 ms, the overall speedup was $\frac{157.6\text{ s}}{83.9\text{ s}} \approx 1.90\times$ of a maximum of $\frac{10\text{ ms}}{5\text{ ms}} \approx 2.0\times$. The speedup for the surrogate was $\frac{78.8\text{ s}}{5.1\text{ s}} \approx 15.5\times$ where 78.8 s is the time that it took to run 5 ms in high-fidelity ($\frac{157.6\text{ s}}{2}$) and 5.1 s the remaining 5 ms in the surrogate (83.9 s − 78.8 s). The speedup results change little between hybrid-light and hybrid.

To measure the accuracy of the hybrid models, we can estimate the mean squared error (MSE) of the packet latency estimates with "high-fidelity only" run as the true value. To compute the MSE, we use the last 2 ms of high-fidelity packet latency data. The MSE for hybrid-lite was 1966 us^2 while 207.2 us^2 for hybrid which is a considerable difference in overall model error. The same patterns can be seen in the longer run of 100 ms: an overall speedup of 2.98× ($\approx \frac{1649.3\text{ s}}{553.4\text{ s}}$), surrogate speedup of 19.7× ($\approx \frac{1649.3\text{ s}\times 7/10}{553.4\text{ s}-1649.3\text{ s}\times 3/10} = \frac{1154.51\text{ s}}{58.61\text{ s}}$), and MSE for hybrid-lite of 1058 ns^2 and 145.8 ns^2 for the hybrid simulation. The speedup of the simulation, therefore, depends on the fraction of the simulation run in surrogate mode.

Discussion. Both hybrid-lite and hybrid incur very little error in throughput, as seen in Table 1. This occurs because throughput indicates the total amount of data arriving at each terminal per second, and, in both cases, the total amount of valid data (non-zombie) packets is not significantly different. The main difference is that in the post-surrogate phase, packets in hybrid-lite will experience fewer queuing delays and report a lower latency than packets injected at the same virtual time in the hybrid case.

In Figure 2, the trend of the high-fidelity only run suggests that the network reaches steady-state congestion and becomes "stable" enough to switch into surrogate mode, namely at 3 ms. A stable enough buffer occupancy can inform us of a possible timestamp to switch programmatically. However, the average packet latency trend for the high-fidelity only case in Figure 3 demonstrates that the traffic performance does not stabilize at 3 ms as the latency trends upward. This discrepancy in buffer occupancy and latency motivates the need for a more insightful approach to identifying steady-state using network data.

Suspending the network gives a more accurate model on the switch back, especially for congested networks (see Figure 3). However, a caveat is, as indicated in Figure 3, that the variance in packet latency is attributed to higher delay in some outlier packets as the simulation continues.

5 CONCLUSION AND FUTURE WORK

We demonstrate two surrogate model strategies to incorporate packet-latency prediction models inside of a PDES HPC network simulation, namely: *hybrid-lite*, in which the network is ignored, and packets are sent to their destination directly instead of injecting them into the network; and, *hybrid*, in which the state of the network is not ignored and instead is suspended, to be later re-animated full of zombie packets. We observe that the hybrid strategy demonstrated better model network accuracy. We have shown that suspending the network is a straightforward strategy to keep the network model consistent for hybrid simulations, and special considerations should be given for parallel implementations.

Future work will focus on evaluating our approach with a variety of traffic patterns such as 1D-stencil, nearest-neighbors, and meaningful application workloads.

Acknowledgement: This work is supported by the Director, Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-06CH11357 and the "Kronos: Enabling Long Timescale PDES Simulations via Multi-Resolution Methods" project.

REFERENCES

[1] P. D. Barnes, C. D. Carothers, D. R. Jefferson, and J. M. LaPre. 2013. Warp speed: Executing Time Warp on 1,966,080 cores. In *Proc. of the 2013 ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (PADS)*. 327–336.

[2] Kevin A. Brown, Neil McGlohon, Sudheer Chunduri, Eric Borch, Robert B. Ross, Christopher D. Carothers, and Kevin Harms. 2021. A Tunable Implementation of Quality-of-Service Classes for HPC Networks. In *ISC High Performance 2021: High Performance Computing (ISC)* ((Virtual) Frankfurt, Germany).

[3] Center for Computational Innovations. [n. d.]. Artificial intelligence multiprocessing optimized system (AiMOS). cci.rpi.edu. https://cci.rpi.edu/aimos (accessed April, @Misccci, author = Center for Computational Innovations, howpublished = cci.rpi.edu, note = https://cci.rpi.edu/aimos (accessed Feb 1, 2022), title = Artificial intelligence multiprocessing optimized system (AiMOS), 2023).

[4] Samir R. Das and Richard M. Fujimoto. 1993. A Performance Study of the Cancelback Protocol for Time Warp. In *Proceedings of the Seventh Workshop on Parallel and Distributed Simulation*. ACM, San Diego California USA, 135–142. https://doi.org/10.1145/158459.158476

[5] Yu Gu, Yong Liu, and D. Towsley. 2004. On integrating fluid models with packet simulation. In *IEEE INFOCOM 2004*, Vol. 4. 2856–2866 vol.4. https://doi.org/10.1109/INFCOM.2004.1354702

[6] Q. He, M. Ammar, G. Riley, and R. Fujimoto. 2002. Exploiting the predictability of TCP's steady-state behavior to speed up network simulation. In *Proceedings. 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*. 101–108. https://doi.org/10.1109/MASCOT.2002.1167066

[7] Yao Kang, Xin Wang, and Zhiling Lan. 2022. Study of Workload Interference with Intelligent Routing on Dragonfly. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis* (Dallas, Texas) *(SC '22)*. Article 20, 14 pages.

[8] Yao Kang, Xin Wang, Neil McGlohon, Misbah Mubarak, Sudheer Chunduri, and Zhiling Lan. 2019. Modeling and Analysis of Application Interference on Dragonfly+. In *SIGSIM PADS*.

[9] Jason Liu. 2007. Parallel Simulation of Hybrid Network Traffic Models. In *21st International Workshop on Principles of Advanced and Distributed Simulation (PADS'07)*. 141–151. https://doi.org/10.1109/PADS.2007.26

[10] Neil McGlohon and Christopher D. Carothers. 2021. Toward Unbiased Deterministic Total Ordering of Parallel Simulations with Simultaneous Events. In *Proceedings of the Winter Simulation Conference* (Phoenix, Arizona) *(WSC '21)*. IEEE Press.

[11] Neil McGlohon, Christopher D. Carothers, K. Scott Hemmert, Michael Levenhagen, Kevin A. Brown, Sudheer Chunduri, and Robert B. Ross. 2021. Exploration of Congestion Control Techniques on Dragonfly-class HPC Networks Through Simulation. In *2021 International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*. 40–50. https://doi.org/10.1109/PMBS54543.2021.00010

[12] Misbah Mubarak, Christopher D Carothers, Robert B Ross, and Philip H Carns. 2017. Enabling Parallel Simulation of Large-Scale HPC Network Systems. *IEEE Trans. Parallel Distrib. Syst.* 28, 1 (2017), 87–100.

[13] Sudhir Srinivasan and Paul F. Reynolds. 1998. Elastic Time. *ACM Trans. Model. Comput. Simul.* 8, 2 (April 1998), 103–139. https://doi.org/10.1145/280265.280267

[14] Noah Wolfe, Misbah Mubarak, Nikhil Jain, Jens Domke, Abhinav Bhatele, Christopher D. Carothers, and Robert B. Ross. 2017. Preliminary Performance Analysis of Multi-Rail Fat-Tree Networks. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (Madrid, Spain) *(CCGrid '17)*. IEEE Press, 258–261. https://doi.org/10.1109/CCGRID.2017.102