Dealing with the Scale Problem

Presented by

Innovative Computing Laboratory

MPI Team

ICL UT
INNOVATIVE COMPUTING
LABORATORY

THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

OAK RIDGE
National Laboratory

Binomial graph

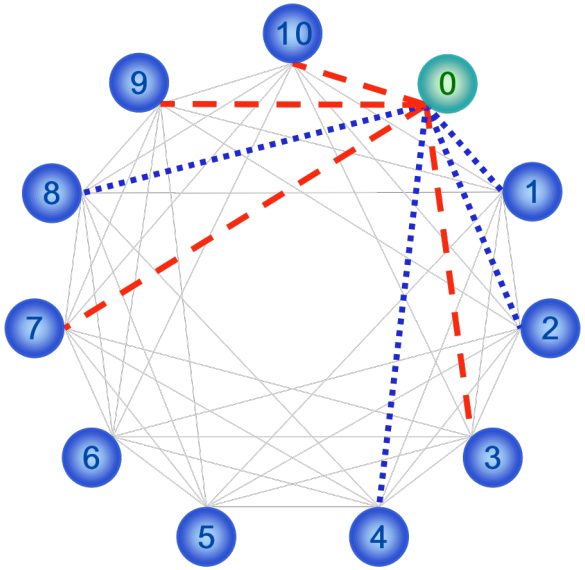
Undirected graph $G:=(V, E)$, $|V|=n$ (any size)

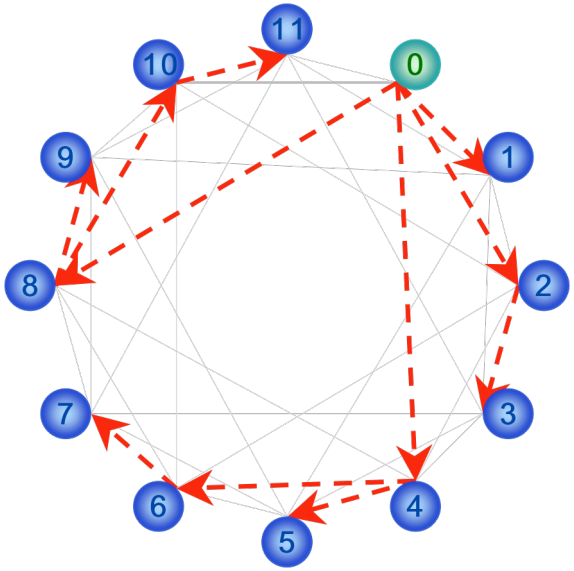
Node $i=\{0,1,2,\ldots,n-1\}$ has links to a set of nodes U

$U=\{i\pm1, i\pm2,\ldots, i\pm2^k \mid 2^k \le n\}$ in a circular space

$U=\{ (i+1)\bmod n, (i+2)\bmod n,\ldots, (i+2^k)\bmod n \mid 2^k \le n \}$ and

$\{ (n+i-1)\bmod n, (n+i-2)\bmod n,\ldots, (n+i-2^k)\bmod n \mid 2^k \le n \}$





Merging all links create binomial graph from each node of the graph

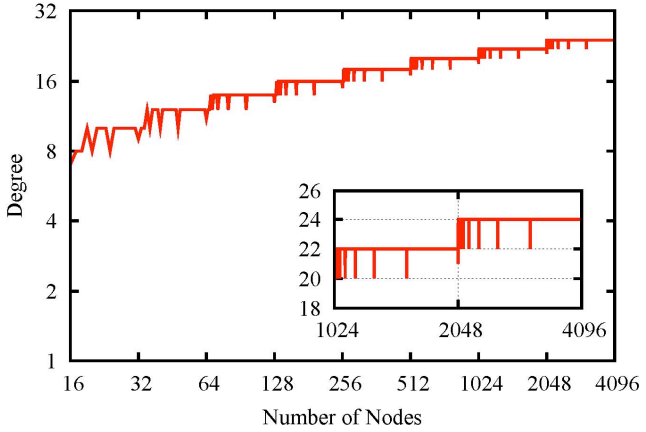
Broadcast from any node in $Log_2(n)$ steps

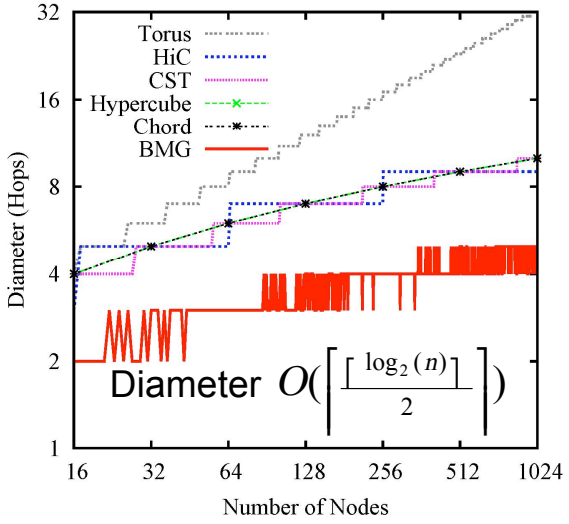
Runtime scalability

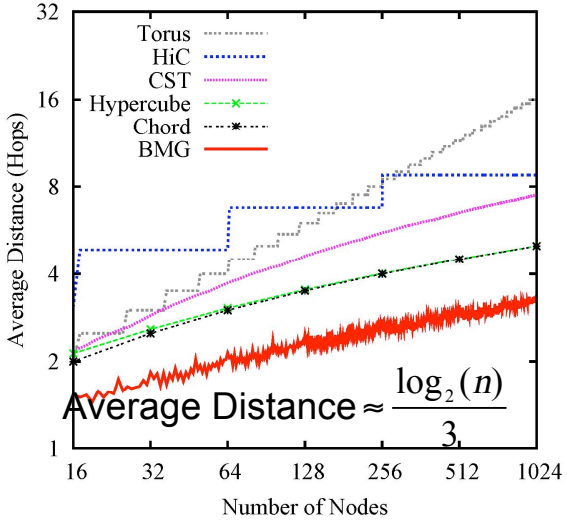
Binomial graph properties

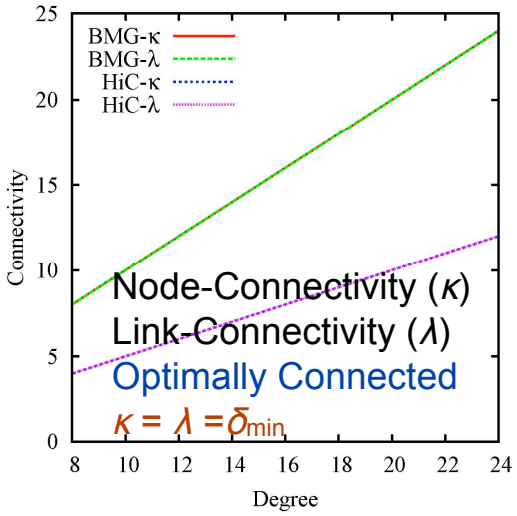
Degree = number of neighbors
= number of connections
= resource consumption

$$\delta = \begin{cases} (2 \times \lceil \log_2 n \rceil) - 1 & \text{For } n = 2^k, \text{where } k \in \mathbb{N} \\ (2 \times \lceil \log_2 n \rceil) - 2 & \text{For } n = 2^k + 2^j, \text{where } k, j \subset \mathbb{N} \wedge k \neq \\ 2 \times \lceil \log_2 n \rceil & \text{Otherwise} \end{cases}$$







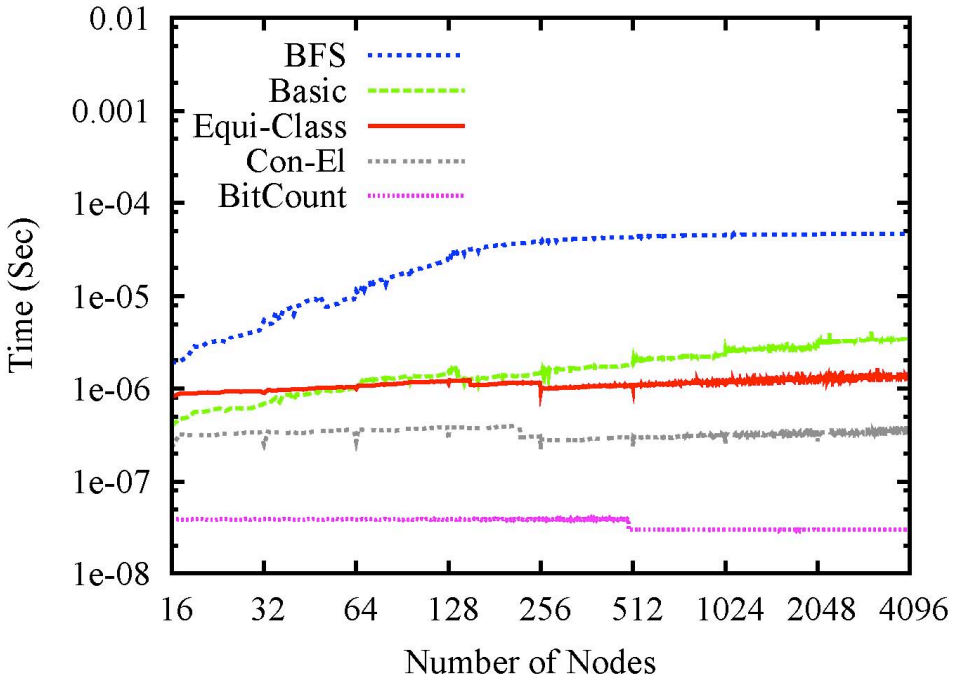


Runtime scalability

Routing cost

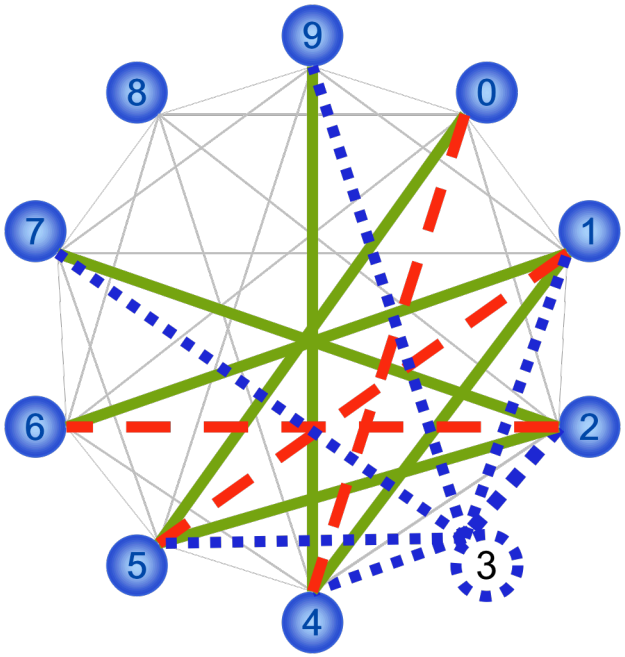
- Because of the counter-clockwise links the routing problem is NP-complete.
- Good approximations exist, with a max overhead of 1 hop.

- Broadcast: optimal in number of steps $\log_2(n)$ (binomial tree from each node)
- Allgather: Bruck algorithm $\log_2(n)$ steps
  - At step s:
    - Node i send data to node i-2s
    - Node i receive data from node i+2s

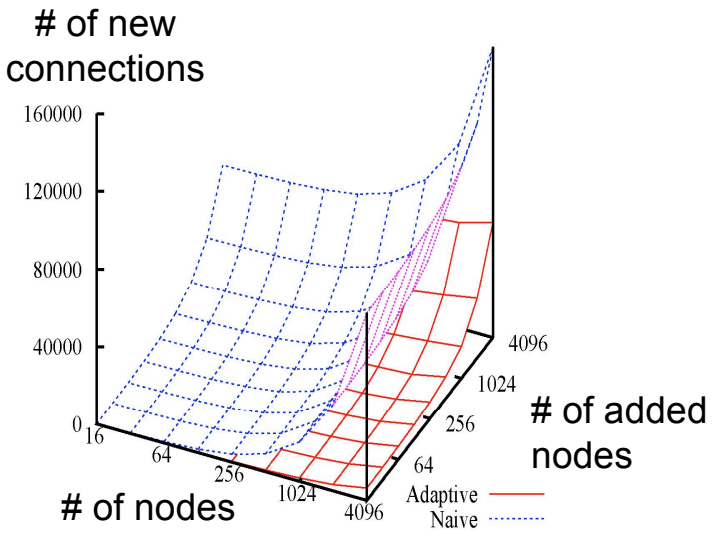


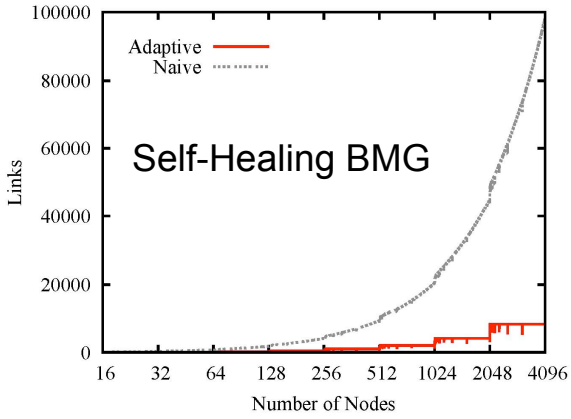
Runtime scalability

Dynamic environments



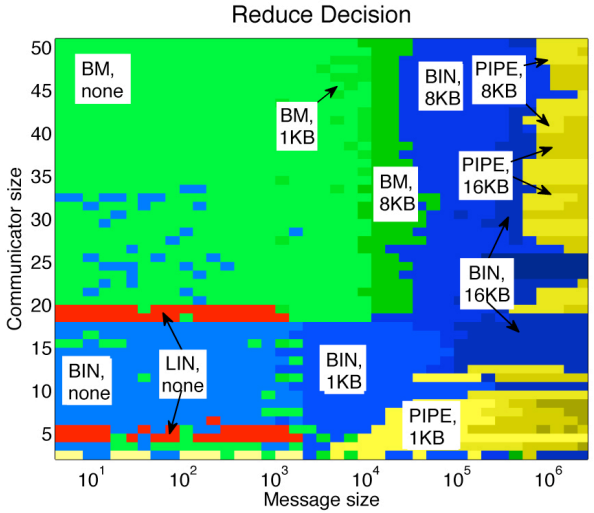
Runtime scalability





Optimization process

- Run-time selection process
  - We use performance models, graphical encoding, and statistical learning techniques to build platform-specific, efficient, and fast runtime decision functions.



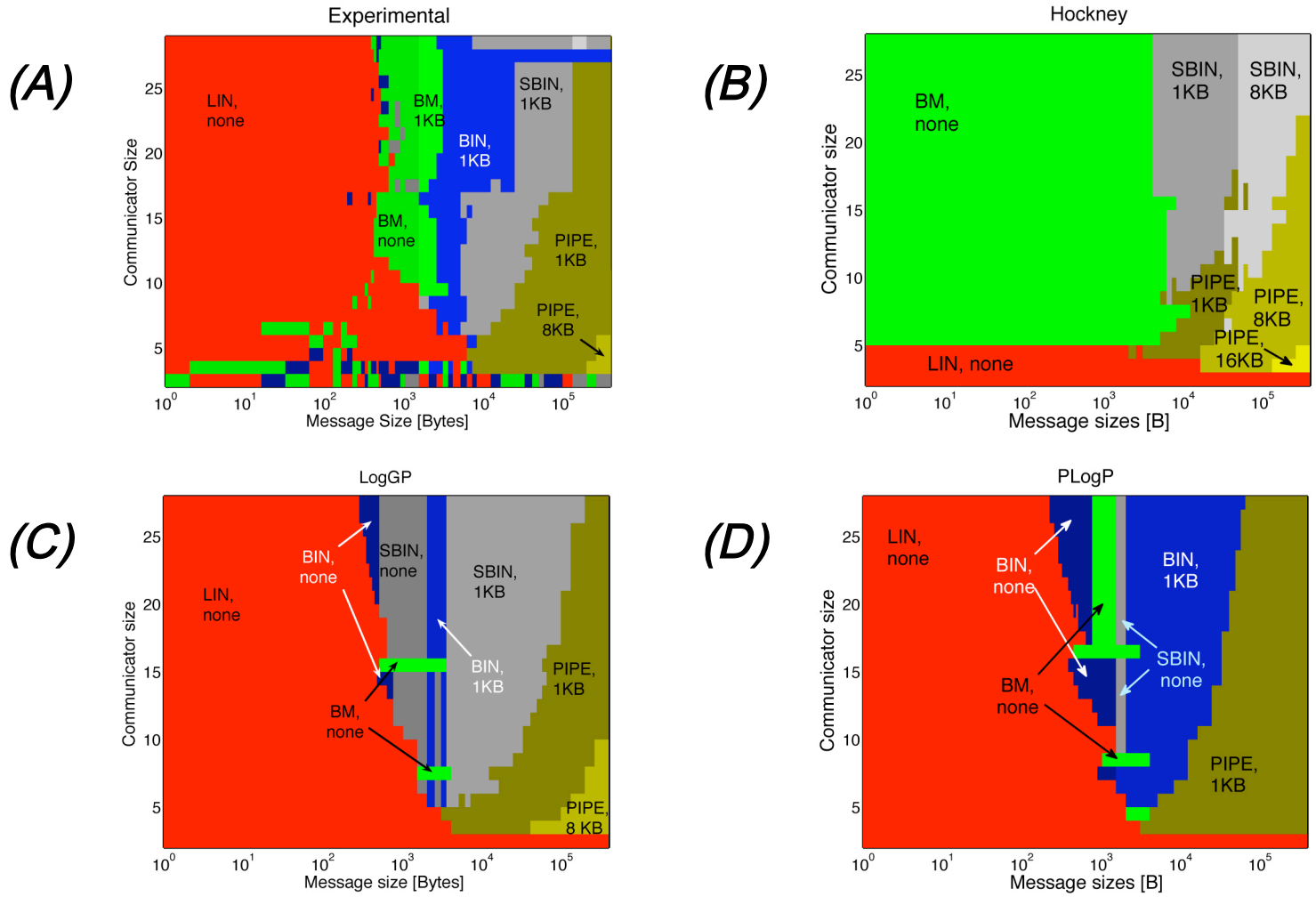
Comm size	Msg size	Algorithm	Seg size	Method index
3	1	Linear	none	15
3	2	Linear	none	15
...	...	...	...	...
50	1MB	Pipeline	8KB	24
...	...	...	...	...

Decision Tree:

```
message_size <= 512 :
|  communicator_size <= 4 :
| |  message_size <= 32 : ring (12.0/1.3)
| |  message_size > 32 : linear (8.0/2.4)
| communicator_size > 4 :
| |  communicator_size > 8 : bruck (100.0/1.4)
| |  communicator_size <= 8 :
| | |  message_size <= 128 : bruck (8.0/1.3)
| | |  message_size > 128 : linear (2.0/1.0)
message_size > 512 :
|  message_size > 1024 : linear (78.0/1.4)
|  message_size <= 1024 :
| |  communicator_size > 56 : linear (5.0/1.2)
| |  communicator_size <= 56 :
| | |  communicator_size <= 8 : linear (3.0/1.1)
| | |  communicator_size > 8 : bruck (5.0/1.2)
```

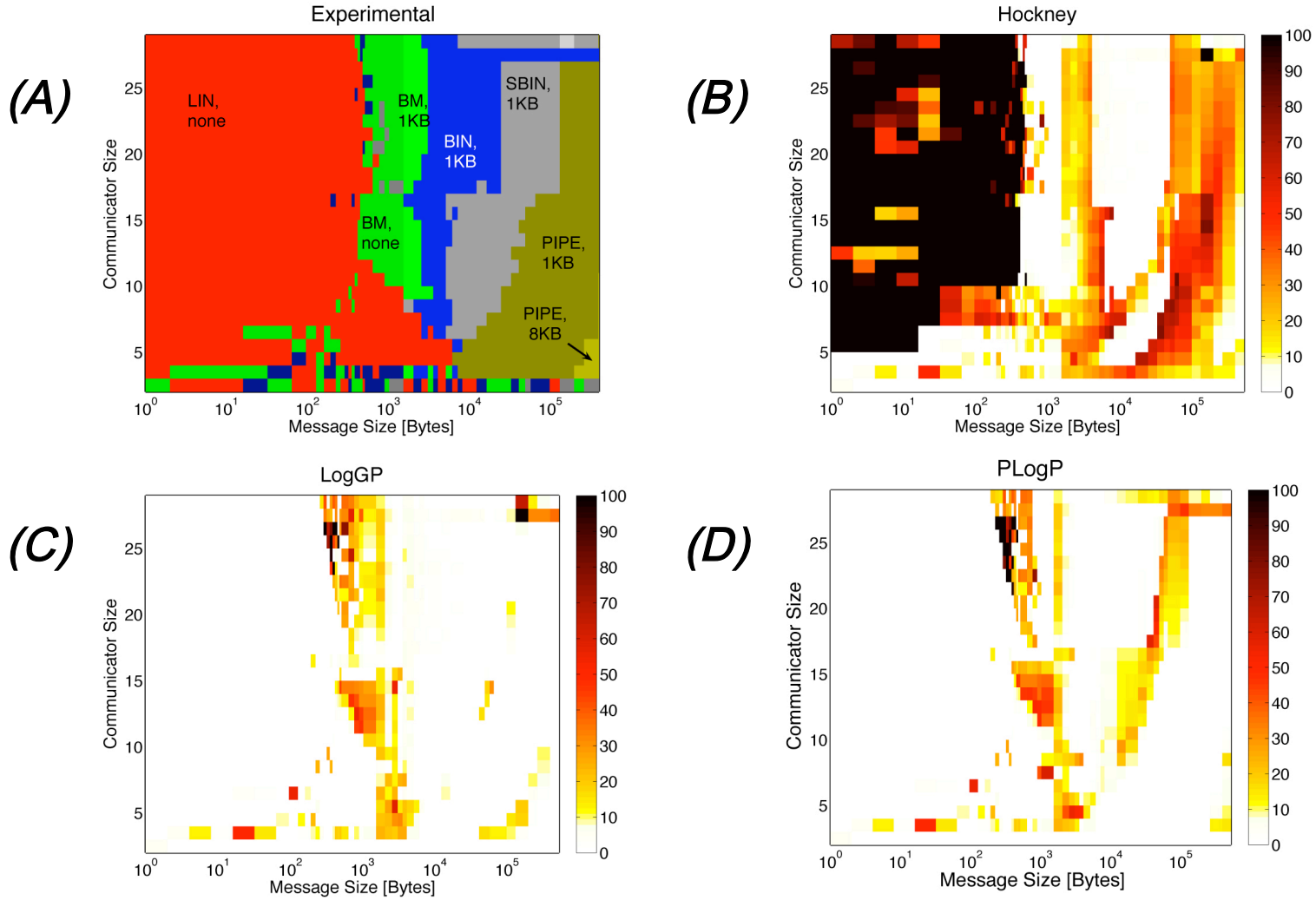
Collective communications

Model prediction



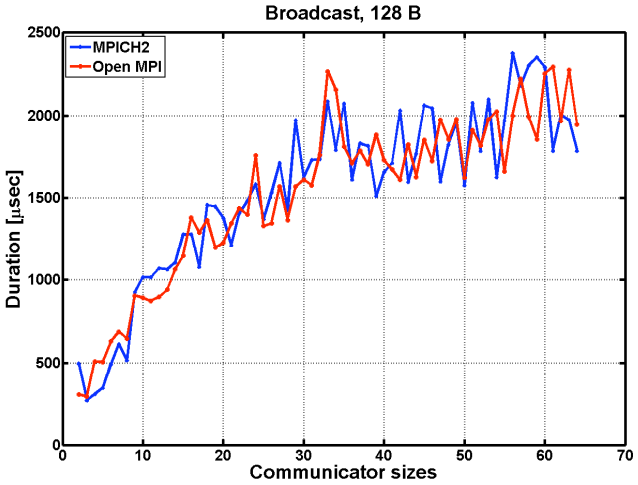
Collective communications

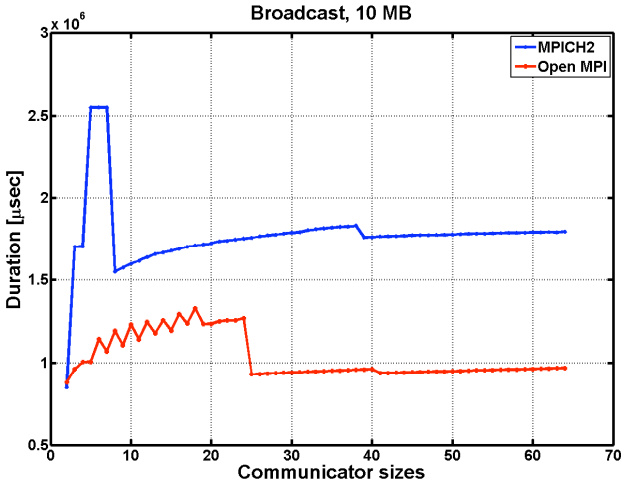
Model prediction



Collective communications

Tuning broadcast

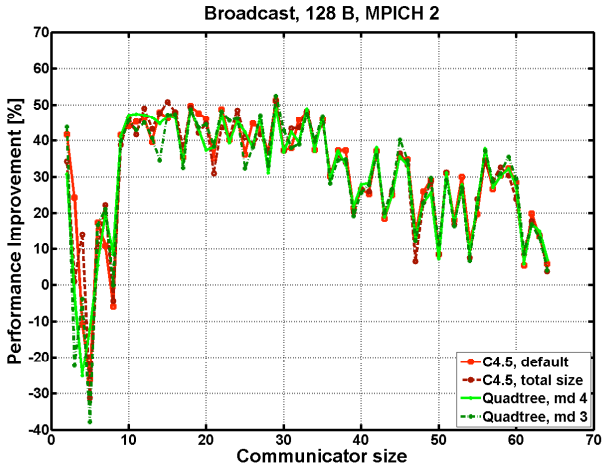


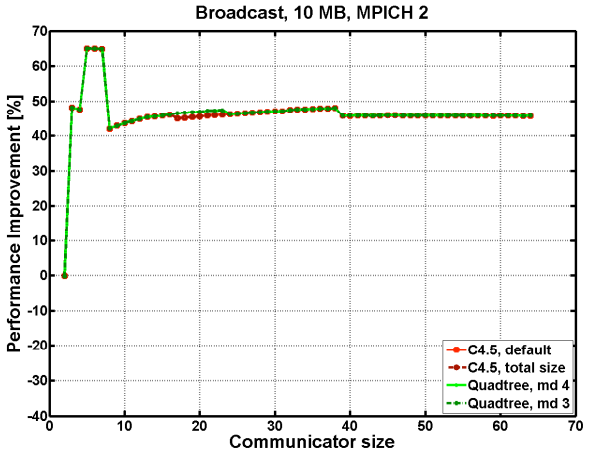


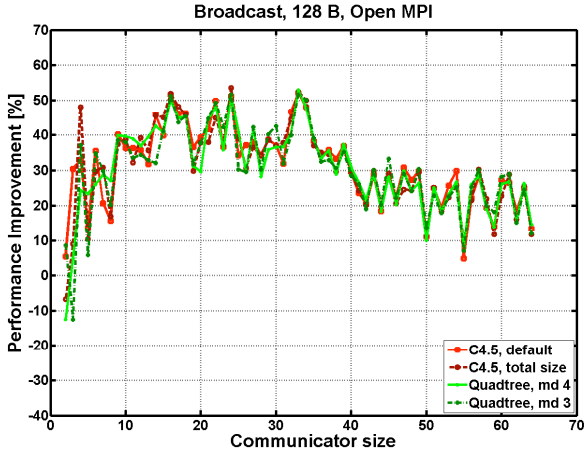
64 Opterons with1Gb/s TCP

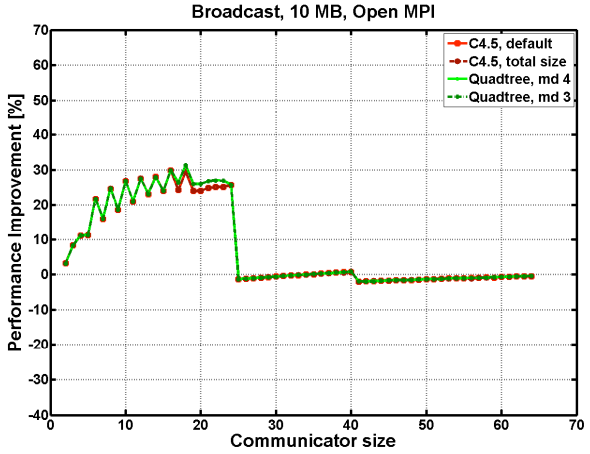
Collective communications

Tuning broadcast









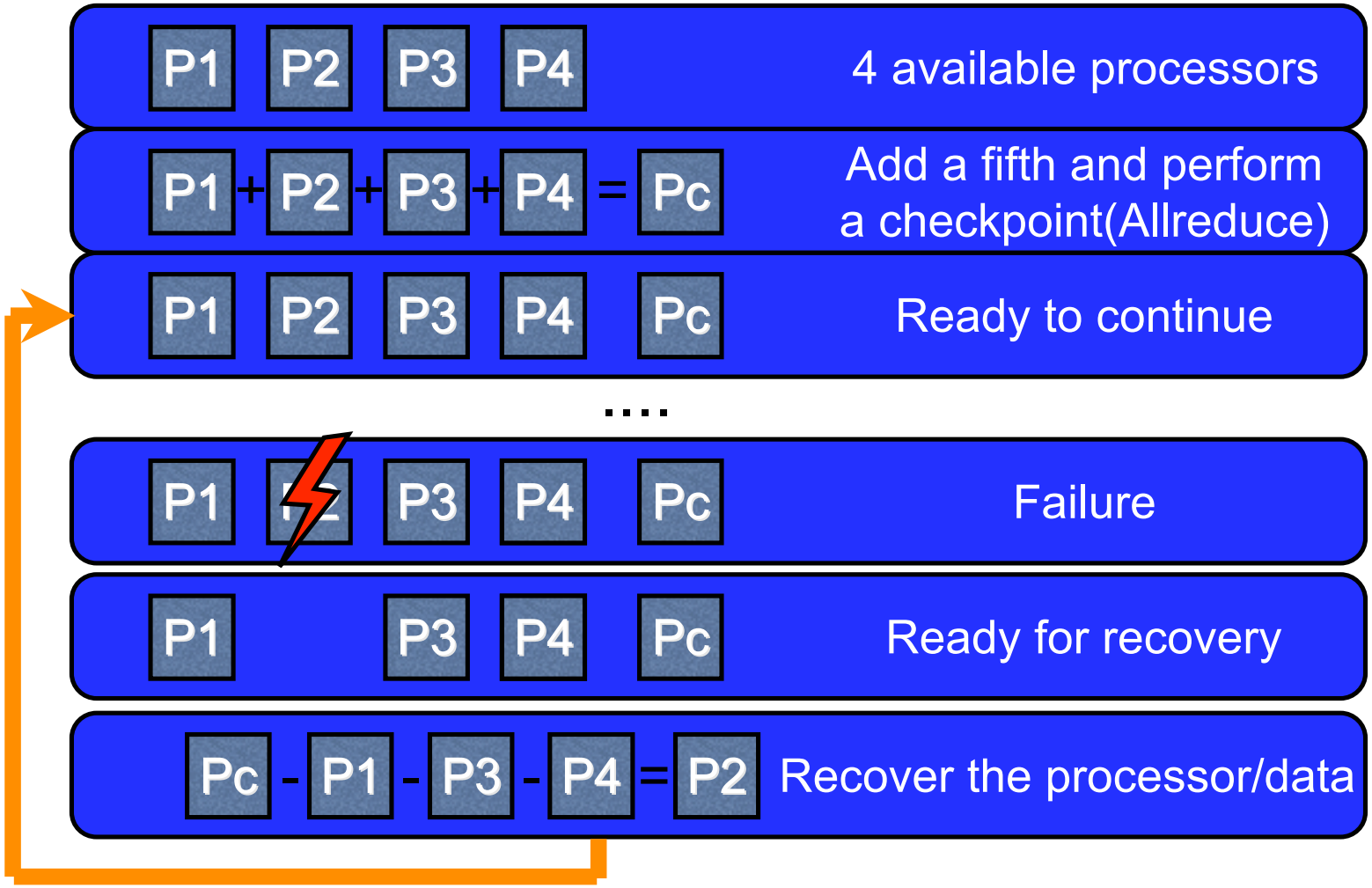
Collective communications

Application tuning

- Parallel Ocean Program (POP) on a Cray XT4.
  - Dominated by MPI_Allreduce of 3 doubles.
- Default Open MPI select recursive doubling
  - Similar with Cray MPI (based on MPICH).
  - Cray MPI has better latency.
  - I.e., POP using Open MPI is 10% slower on 256 processes.
- Profile the system for this specific collective and determine that “reduce + bcast” is faster.
  - Replace the decision function.
  - New POP performance is about 5% faster than Cray MPI.

Collective communications

Diskless checkpointing



Fault tolerance

Diskless checkpointing

- How to checkpoint?
  - Either floating-point arithmetic or binary arithmetic will work.
  - If checkpoints are performed in floating-point arithmetic then we can exploit the linearity of the mathematical relations on the object to maintain the checksums.
- How to support multiple failures?
  - Reed-Salomon algorithm.
  - Support p failures require p additional processors (resources).

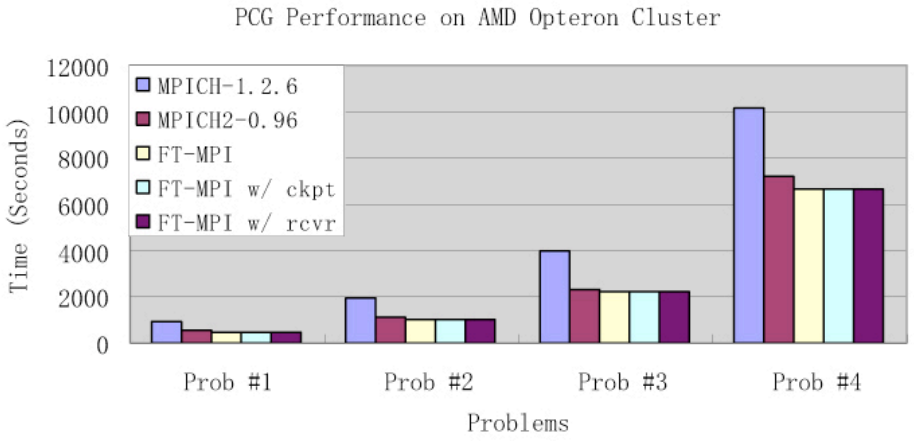
Fault tolerance

PCG

- Fault tolerant CG
- 64x2 AMD 64 connected using GigE

	Size of the Problem	Num. of Comp. Procs
Prob #1	164,610	15
Prob #2	329,220	30
Prob #3	658,440	60
Prob #4	1,316,880	120

Performance of PCG with different MPI libraries



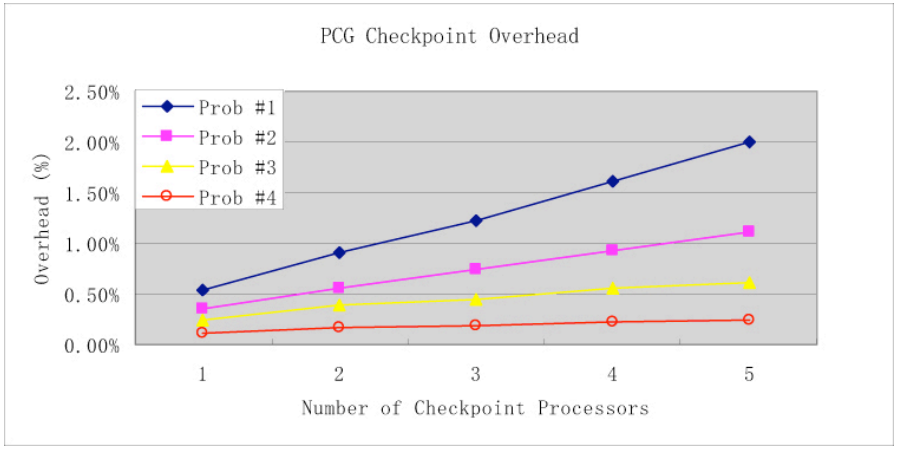
For ckpt we generate one ckpt every 2000 iterations

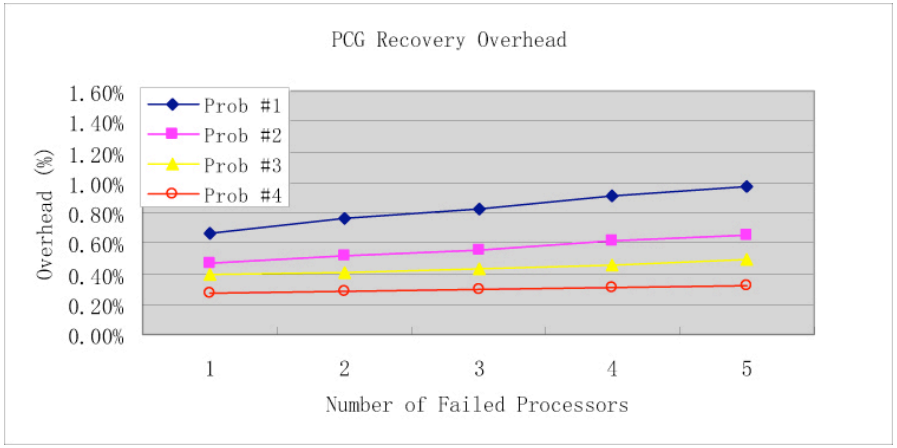
Fault tolerance

PCG

Time	Prob #1	Prob #2	Prob #3	Prob #4
1 ckpt	2.6	3.8	5.5	7.8
2 ckpt	4.4	5.8	8.5	10.6
3 ckpt	6.0	7.9	10.2	12.8
4 ckpt	7.9	9.9	12.6	15.0
5 ckpt	9.8	11.9	14.1	16.8

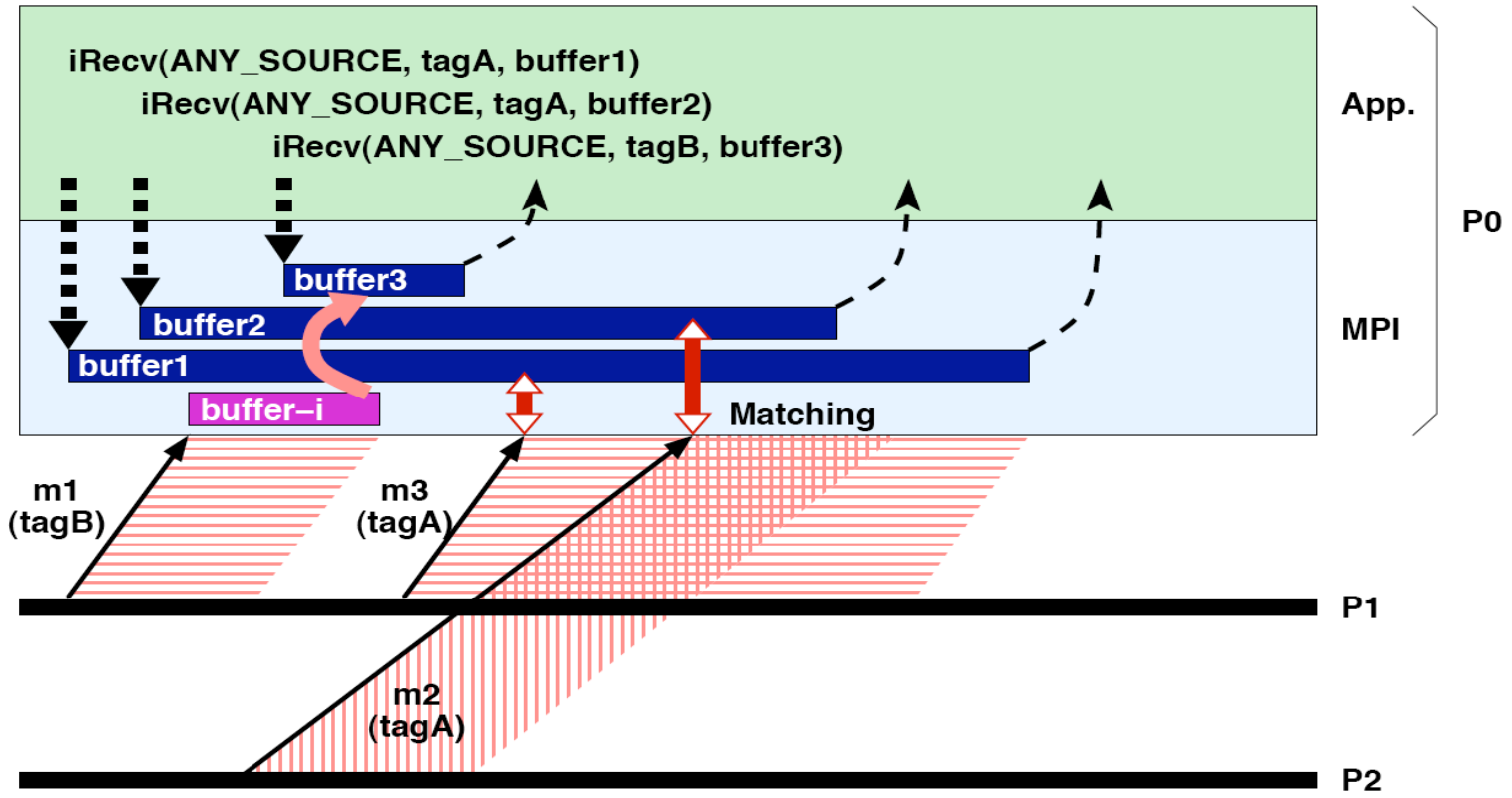
Checkpoint overhead in seconds





Fault tolerance

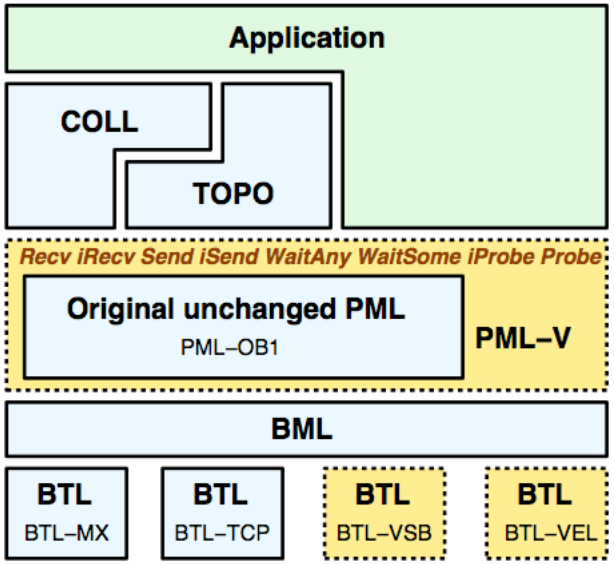
Detailing event types to avoid intrusiveness



Fault tolerance

Interposition in Open MPI

- We want to avoid tainting the base code with #ifdef FT_BLALA.
- Vampire PML loads a new class of MCA components.
  - Vprotocols provide the entire FT protocol (only pessimistic for now).
  - You can use the ability to define subframeworks in your components!
- Keep using the optimized low level and zero-copy devices (BTL) for communication.
- Unchanged message scheduling logic.



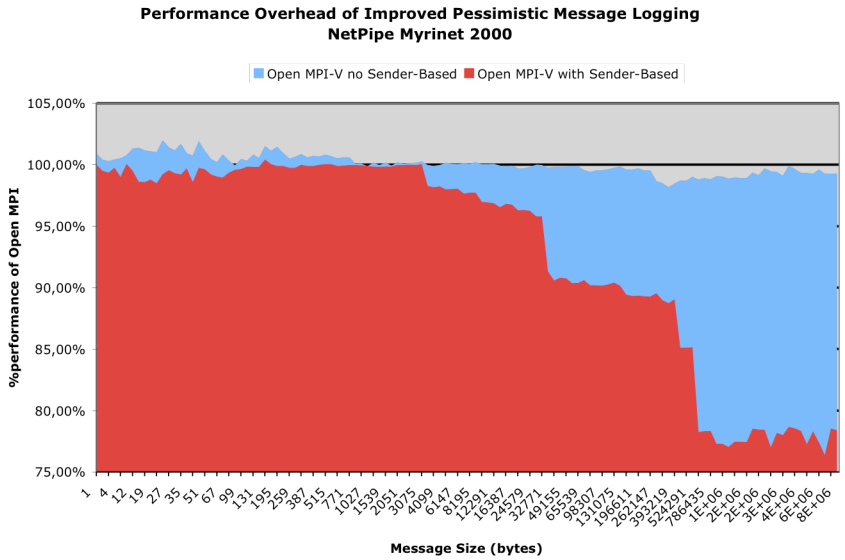
Fault tolerance

Performance overhead

	BT	SP	FT	CG	MG						LU					
#processors	all				4	32	64	256	512	1024	4	32	64	256	512	1024
%non-deterministic	0	0	0	0	40.33	29.35	27.10	22.23	20.67	19.99	1.13	0.66	0.80	0.80	0.75	0.57

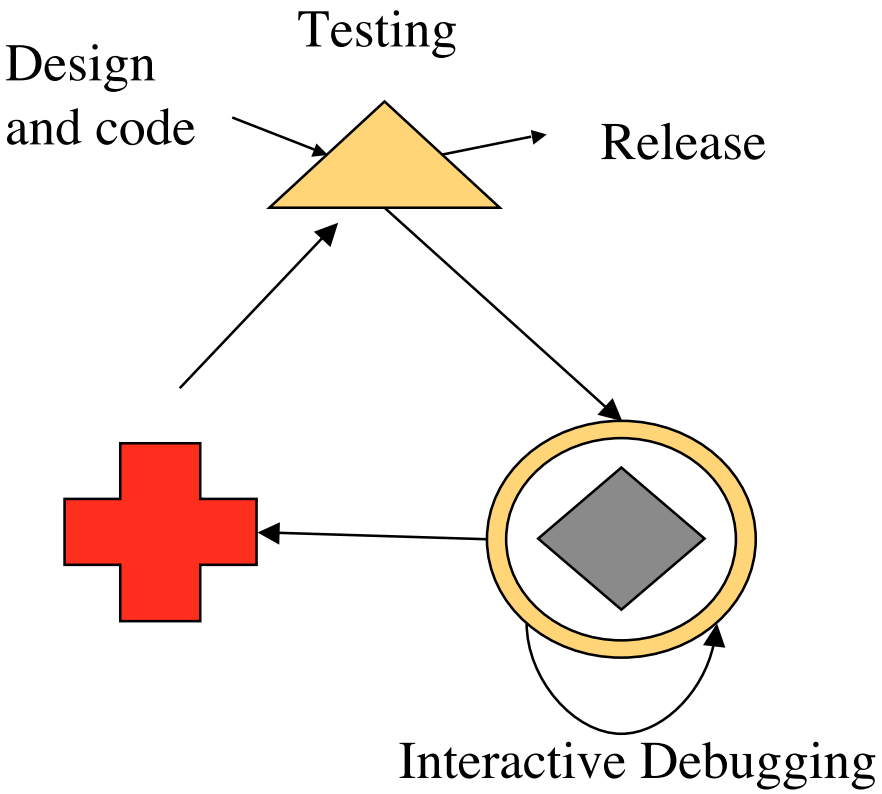
Table 1. Percentage of non-deterministic events to total number of exchanged messages on the NAS Parallel Benchmarks (Giga-Ethernet, class B).

- Myrinet 2G (mx 1.1.5)—Opteron 146x2—2GB RAM—Linux 2.6.18—gcc/gfortran 4.1.2—NPB3.2—NetPIPE.
- Only two application kernels show non-deterministic events (MG, LU).



Fault tolerance

Debugging applications

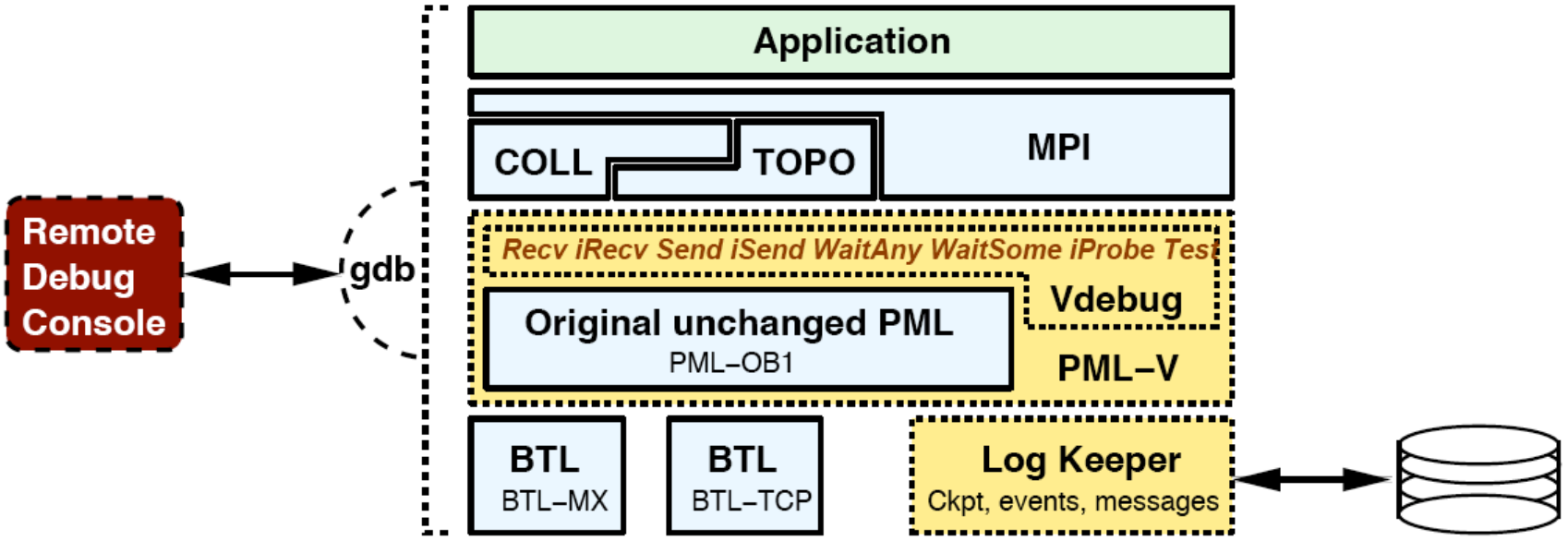


- Usual scenario involves
  - Programmer design testing suite
  - Testing suite shows a bug
  - Programmer runs the application in a debugger (such as gdb) up to understand the bug and fix it
  - Programmer runs the testing suite again
    - Cyclic debugging

Fault tolerance

Interposition Open MPI

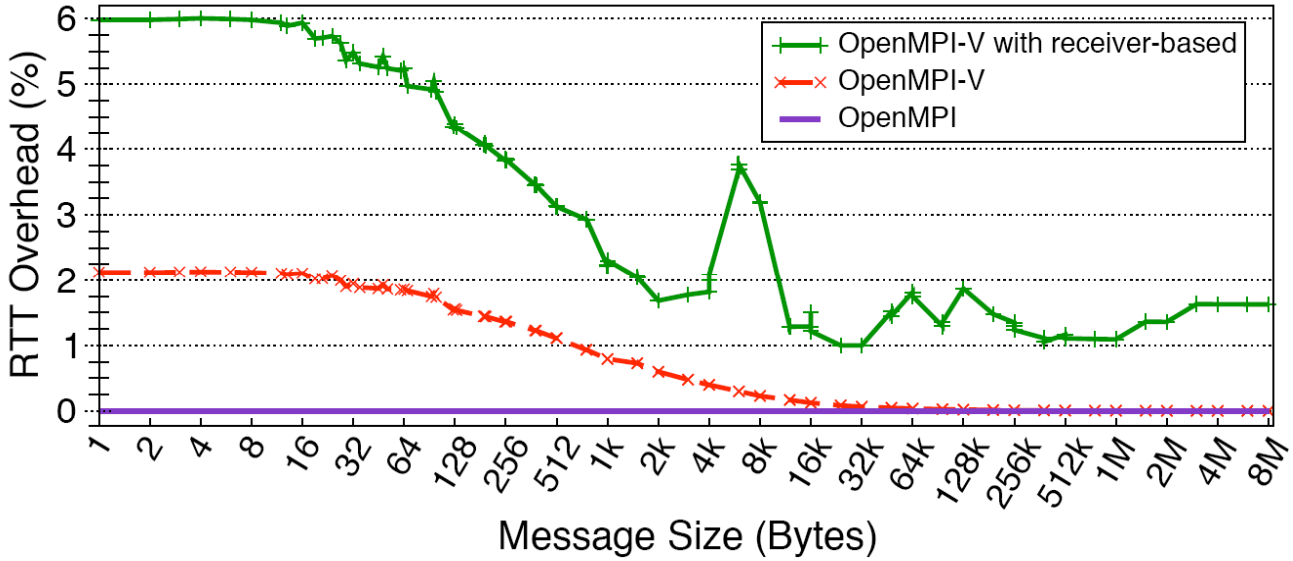
- Events are stored (asynchronously on disk) during initial run.
- Keep using the optimized low level and zero-copy devices (BTL) for communication.
- Unchanged message scheduling logic.
- We expect low impact on application behavior.



Fault tolerance

Performance overhead

- Myrinet 2G (mx 1.0.3)—Opteron 146x2—2GB RAM—Linux 2.6.18—gcc/gfortran 4.1.2—NPB3.2—NetPIPE
- 2% overhead on bare latency, no overhead on bandwidth.
- Only two application kernels show non-deterministic events.
- Receiver-based has more overhead; moreover, it incurs large amount of logged data—350 MB on simple ping-pong (this is beneficial; it is enabled on-demand).



Vdebug Application overhead (%)		
Rcvr-based	None	Yes
BT.B.64	0	6.4
LU.B.64	0	2.2
MG.B.64	2.2	4
CG.B.64	0.8	4
FT.B.64	0	7.3
SP.B.64	0	6.4
EP.B.64	0	0

Fault tolerance

Log size on NAS parallel benchmarks

Log size per process on NAS Parallel Benchmarks (kB)

#procs	4	8	16	32	64	128	256	512	1024	Average
LU.B	11.2	11.2	11.9	10.6	13.9	19	14.9	14.2	12.7	13.3
MG.B	11.2	11.2	10.8	10.6	10.6	9.8	9.5	9.3	9.2	10.24

- Among the 7 NAS kernels, only 2 NPB generates nondeterministic events.
- Log size does not correlate with number of processes.
- The more scalable is the application, the more scalable is the log mechanism.
- Only 287KB of log/process for LU.C.1024 (200MB memory footprint/process).

Fault tolerance

Contact

Dr. George Bosilca

Innovative Computing Laboratory
Electrical Engineering and Computer Science Department
University of Tennessee
bosilca@eecs.utk.edu