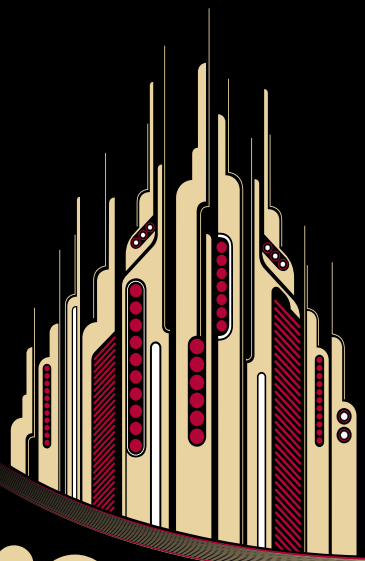
Open MPI Tutorial

Andrew Lumsdaine
Joshua Hursey
Jeffrey M. Squyres
Abhishek Kulkarni

Thurs., Nov. 19, 2009
10:00 a.m. – 12:00 p.m.

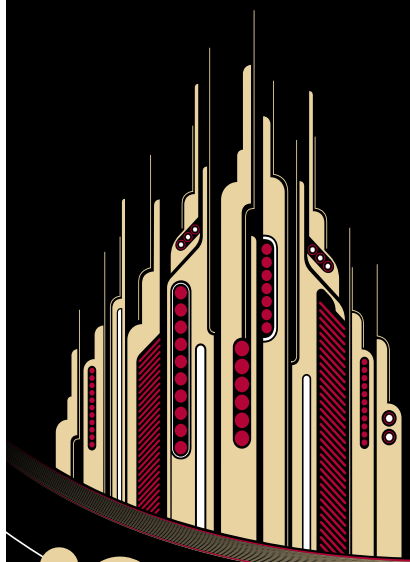
http://osl.iu.edu/research/ft



Look to the future of high-performance computing.

Open MPI Project Overview

Jeff Squyres
Open MPI Architect
Cisco Systems



Look to the future of high-performance computing.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

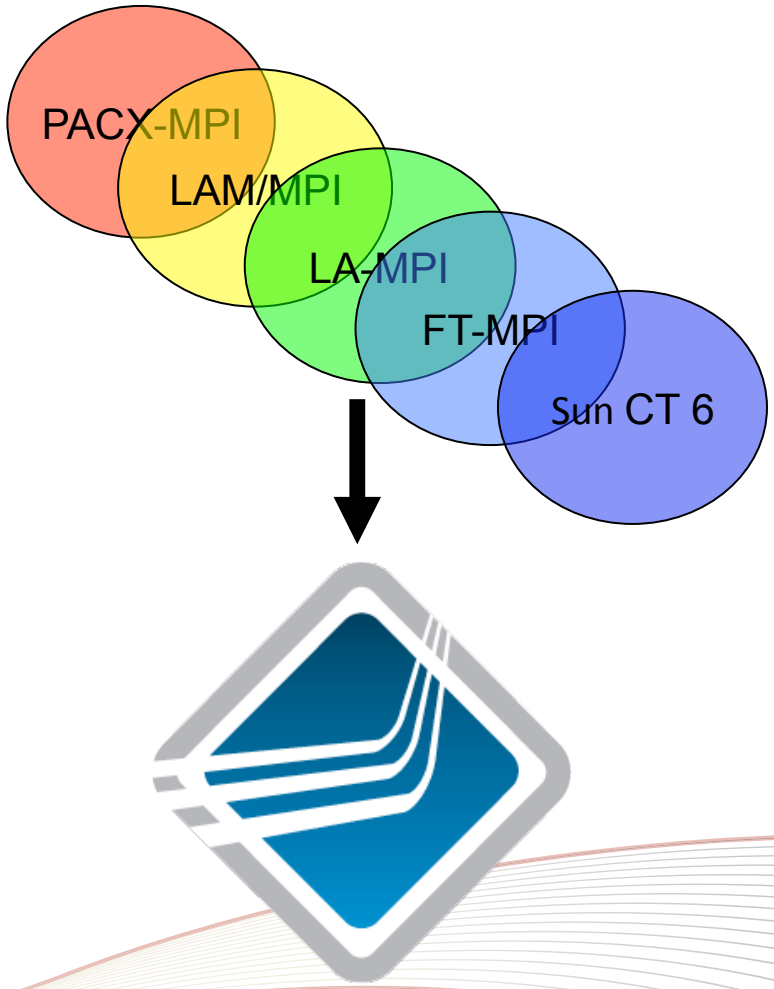
SC Portland, OR 2009

Before We Start…

- It's two words!
  - Open MPI
  - **NOT** "OpenMPI"
- Frequently abbreviated "OMPI"
  - Pronounced "oom-pee"

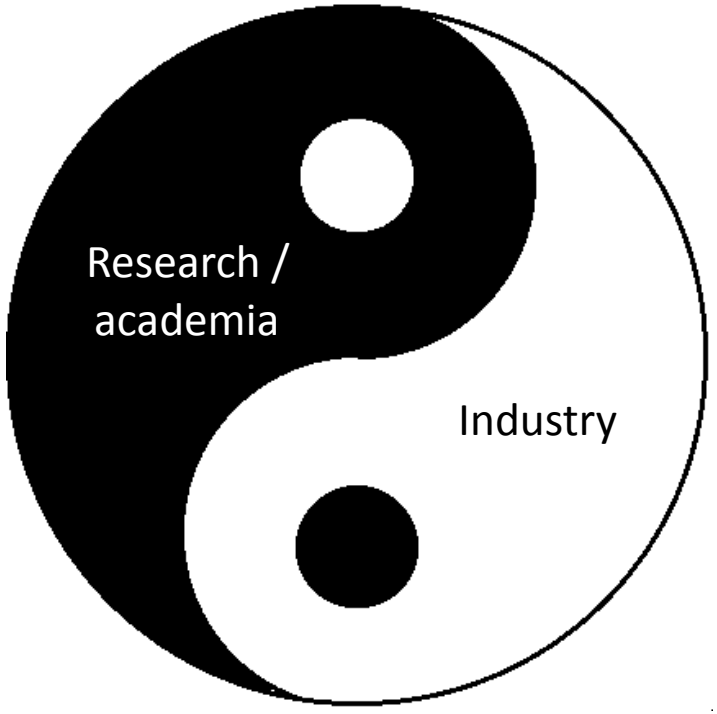
Open MPI Is…

- Evolution of several prior MPI implementations
- Open source project and community
  - Production quality
  - Vendor-friendly
  - Research- and academic-friendly
- MPI-2.1 compliant



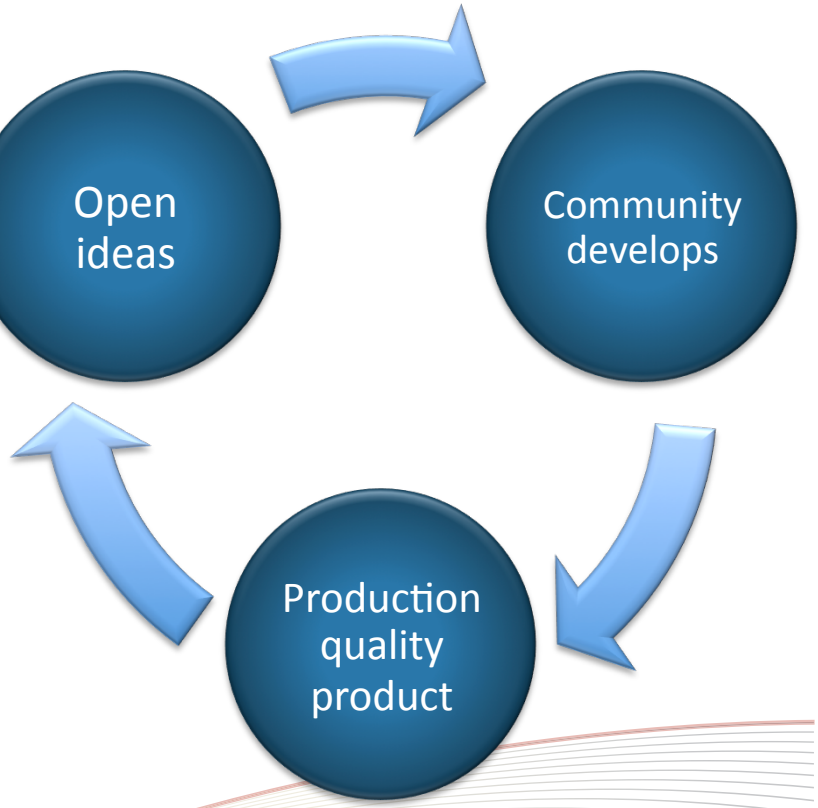
Why Does Open MPI Exist?

- Maximize all MPI expertise, including:
  - Research / academia
  - Vendors
  - Customers, enterprise
- Utilize years of MPI research and experience
- The sum is greater than the parts



Open Source HPC

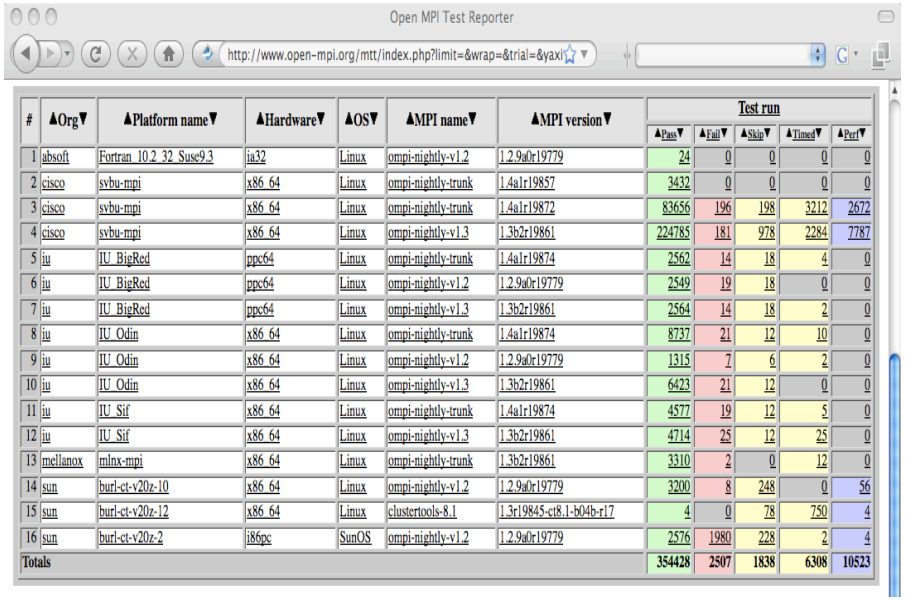
- Open source HPC is good for everyone
  - Open information transfer
  - Feed them back into production
- Shorten the cycle from research to commodity
- Researchers have ideas; industry has production capability
  - There are smart people in both!



16 Members, 9 Contributors, 2 Partners

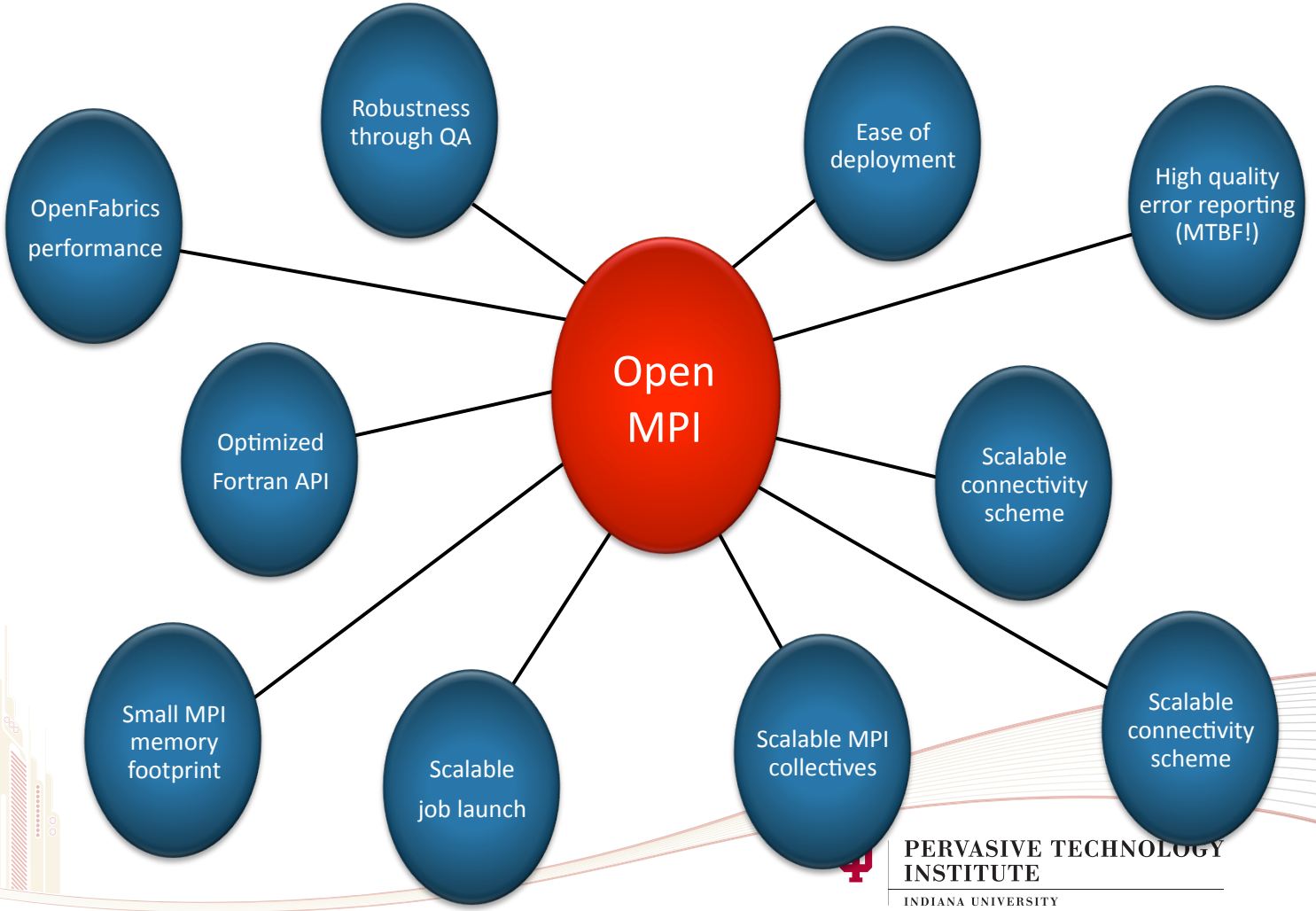
Give To Get

- Nightly community regression testing
  - 100k's tests per night
  - Web-based analysis tools
- Strive for consensus
  - But realize it isn't always possible (or necessary)
- Perform "community service"
  - Example: Fortran API maintenance



#	Org	Platform name	Hardware	OS	MPI name	MPI version	Test run				
							Pass	Fail	Skip	Timed	Perf
1	absoft	Fortran_10.2_32_Suse9.3	ia32	Linux	ompi-nightly-v1.2	1.2.9a0r19779	24	0	0	0	0
2	cisco	svbu-mpi	x86_64	Linux	ompi-nightly-trunk	1.4a1r19857	3432	0	0	0	0
3	cisco	svbu-mpi	x86_64	Linux	ompi-nightly-trunk	1.4a1r19872	83656	196	198	3212	2672
4	cisco	svbu-mpi	x86_64	Linux	ompi-nightly-v1.3	1.3b2r19861	224785	181	978	2284	7787
5	iu	IU_BigRed	ppc64	Linux	ompi-nightly-trunk	1.4a1r19874	2562	14	18	4	0
6	iu	IU_BigRed	ppc64	Linux	ompi-nightly-v1.2	1.2.9a0r19779	2549	19	18	0	0
7	iu	IU_BigRed	ppc64	Linux	ompi-nightly-v1.3	1.3b2r19861	2564	14	18	2	0
8	iu	IU_Odin	x86_64	Linux	ompi-nightly-trunk	1.4a1r19874	8737	21	12	10	0
9	iu	IU_Odin	x86_64	Linux	ompi-nightly-v1.2	1.2.9a0r19779	1315	7	6	2	0
10	iu	IU_Odin	x86_64	Linux	ompi-nightly-v1.3	1.3b2r19861	6423	21	12	0	0
11	iu	IU_Sif	x86_64	Linux	ompi-nightly-trunk	1.4a1r19874	4577	19	12	5	0
12	iu	IU_Sif	x86_64	Linux	ompi-nightly-v1.3	1.3b2r19861	4714	25	12	25	0
13	mellanox	mlnx-mpi	x86_64	Linux	ompi-nightly-trunk	1.3b2r19861	3310	2	0	12	0
14	sun	burl-ct-v20z-10	x86_64	Linux	ompi-nightly-v1.2	1.2.9a0r19779	3200	8	248	0	56
15	sun	burl-ct-v20z-12	x86_64	Linux	clustertools-8.1	1.3r19845-ct8.1-b04b-r17	4	0	78	750	4
16	sun	burl-ct-v20z-2	i86pc	SunOS	ompi-nightly-v1.2	1.2.9a0r19779	2576	1980	228	2	4
Totals							354428	2507	1838	6308	10523

Putting It All Together



Open MPI Community

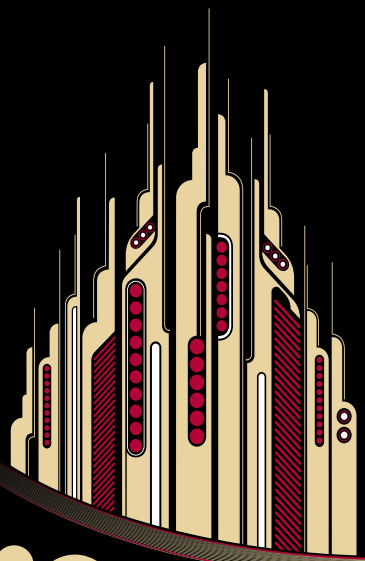
- Multiple software projects
  - Open MPI
  - Hardware Locality (hwloc)
  - MPI Testing Tool (MTT)
  - Open Resilient Cluster Manager
- We **NEED** your ideas, creativity, and talent
  - Come join us!

Open MPI Community

- Contributors must sign "Apache" agreement
  - Allows Open MPI to redistribute your work
  - …but you still own it
  - I am not a lawyer!
- Agreements available on OMPI web site
  - http://www.open-mpi.org/

Open MPI v1.3.4

Jeff Squyres
Open MPI Architect
Cisco Systems



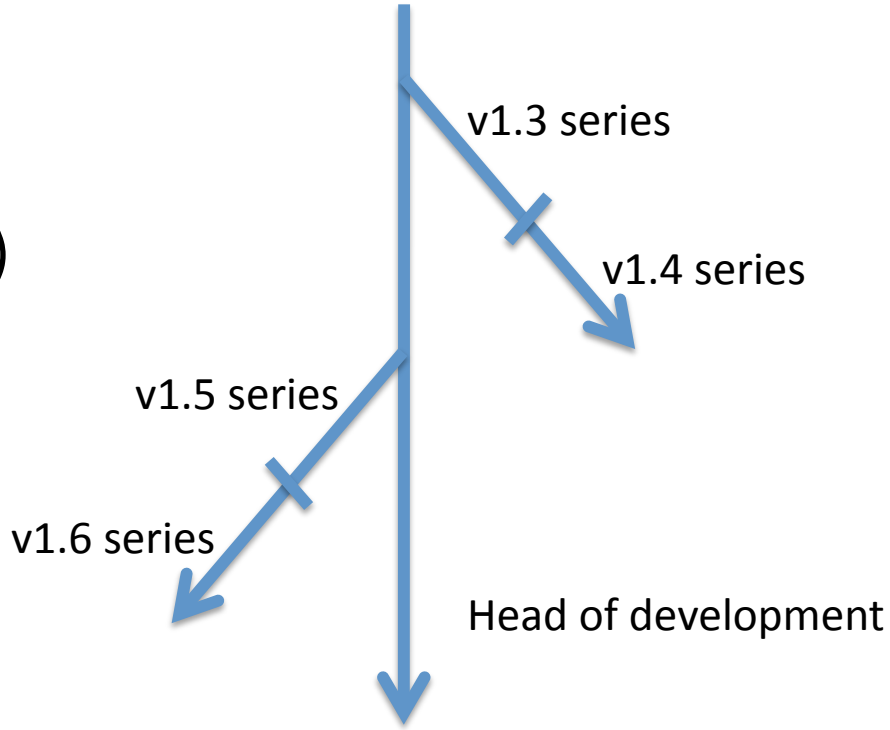
Look to the future of high-performance computing.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

SC Portland, OR 2009

Open MPI Releases

- Concurrent releases
  - Feature series (odd)
  - Super-stable series (even)
- Currently:
  - v1.3 is feature series
  - v1.2 is stable series
- Just about to transition
  - v1.5 will be feature series
  - v1.4 will be stable series

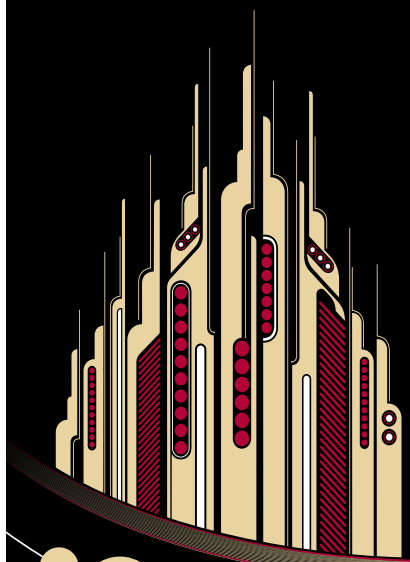


Open MPI v1.3.3

- Current stable release: v1.3.3
  - v1.3.4 to be released shortly after SC09
  - …so let's talk about v1.3.4
- Supports:
  - Linux, Solaris, OS X, MS Windows
  - TCP, shared memory, Myricom MX, OpenFabrics, uDAPL, Portals, Quadrics Elan, SCTP, QLogic PSM

Building and Installing Open MPI

Jeff Squyres
Open MPI Architect
Cisco Systems



Look to the future of high-performance computing.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

SC Portland, OR 2009

Download

- Download latest from Open MPI web site
  - http://www.open-mpi.org/
  - "Download" link on the left
  - Save source tarball locally
- Development code also available
  - Anonymous / read only access to Subversion
  - Mercurial cloning
  - Nightly snapshot tarballs

Extract the Tarball

- On Linux (GNU tar)

```
$ tar zxf openmpi-<version>.tar.gz
```

or

```
$ tar jxf openmpi-<version>.tar.bz2
```

Build Process

- GNU tools-driven
  - "configure / make / make install"
  - Extremely common to open source projects
- Save all output
  - ***Vital*** for diagnosing problems later
  - http://www.open-mpi.org/community/help/

Configuring

- Almost always want to specify a prefix
- Bourne shell flavors

```
$ cd openmpi-<version>
$ ./configure --prefix=$HOME/local 2>&1 \
    | tee config.out
```

- C shell flavors

```
% cd openmpi-<version>
% ./configure --prefix=$HOME/local \
    |& tee config.out
```

- This will take several minutes

Configuring

- There are many configure options available
- Most common:
  - Specifying network driver locations
  - Specifying resource manager locations
- Configure “usually” finds what it needs
  - …if accompanying software is installed into default locations

Configuring

- But sometimes configure needs some help

```
$ ./configure --with-tm=/opt/local/torque \
    --with-mx=/opt/local/myricom-mx …
```

- Command-line switches specify locations
  - Generally need “devel” libraries and header files

Common Configure Options

- --prefix: specify where to install Open MPI
- Common support packages
  - --with-<name>=<directory>
  - --with-mx
  - --with-openib (OFED)
  - --with-portals
  - --with-psm (QLogic)
  - --with-tm (Torque)
  - --with-lsf
  - --with-sge

Using Different Compilers

- Open MPI works with several compiler suites
- Specify which compilers to use via configure
  - It is generally best to use single compiler suite
  - Use CC, CXX, FC, F77 to specify compilers
  - Can also specify CFLAGS, CXXFLAGS, FCFLAGS, F77FLAGS

```
$ ./configure CC=icc CXX=icpc FC=ifort F77=ifort \
    "FCFLAGS=-O3 -i8" "F77FLAGS=-O3 -i8" …
```

Build and Install

- Open MPI ready to be built and installed
- Simple command

```
make all install 2>&1 | tee make.out
```

- Both builds and installs
  - Can also do parallel builds
  - For example (GNU make):

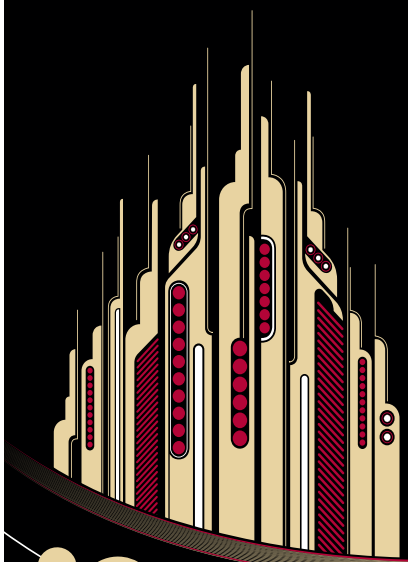
```
make -j 4 all 2>&1 | tee make.out
make install  2>&1 | tee install.out
```

Common Support Areas

- “It didn’t work!”
- First questions:
  - Did it configure properly?
  - Did it compile properly?
  - Did it install properly?
  - Is the environment setup properly?
  - http://www.open-mpi.org/community/help/
- First place to look is the files you saved

Tuning Your MPI Application with Open MPI

Jeff Squyres
Open MPI Architect
Cisco Systems



Look to the future of high-performance computing.

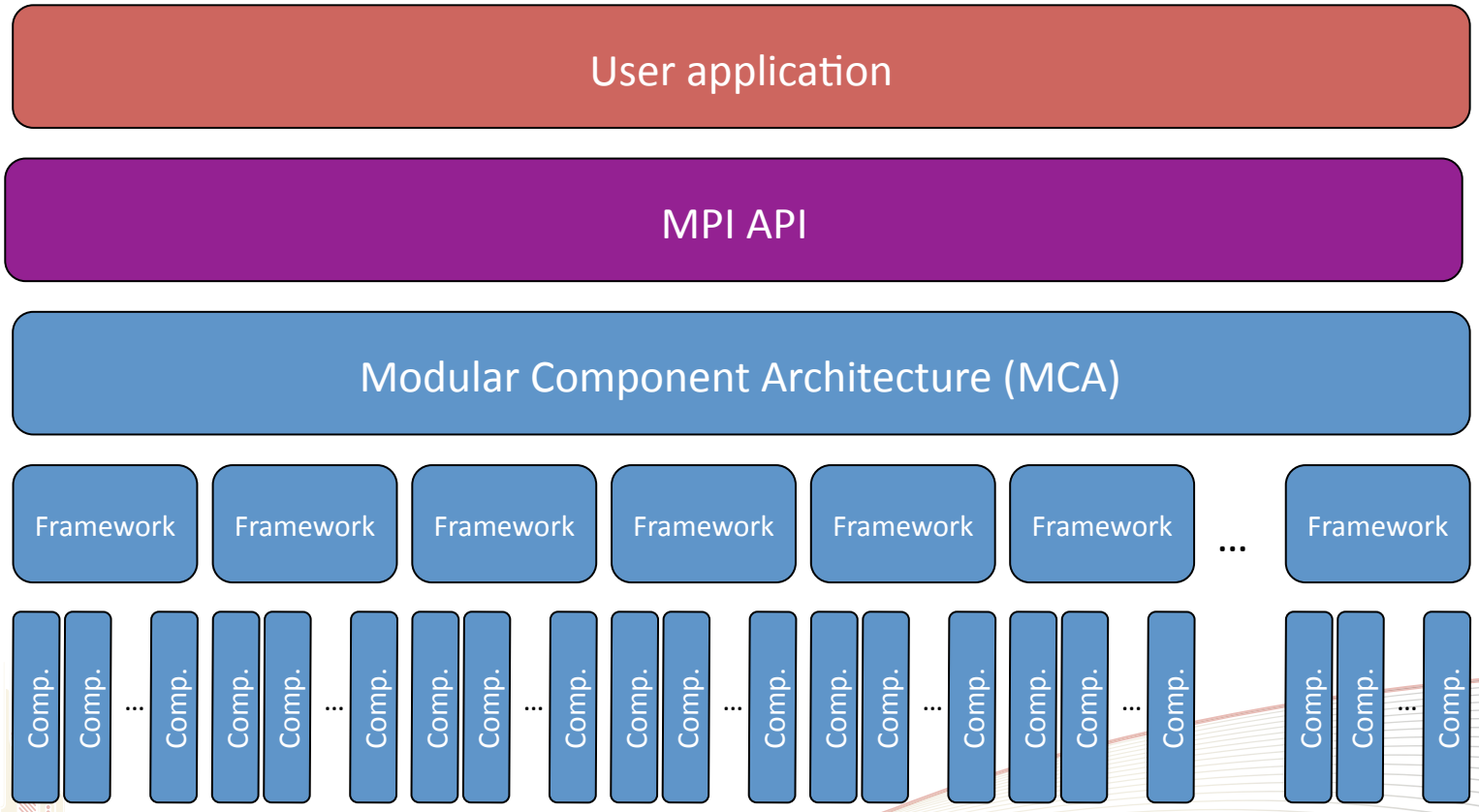
PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

SC Portland, OR 2009

Open MPI is Based on Plugins

- Lots and lots of plugin types
  - Back-end network
  - Resource manager support
  - Operating system support
- All can be loaded (or not) at run-time
  - Choice of network is a run-time decision
  - *User applications no longer linked against network libraries (e.g., libibverbs)*
  - Companion concept: run-time parameters

Plugin High-Level View



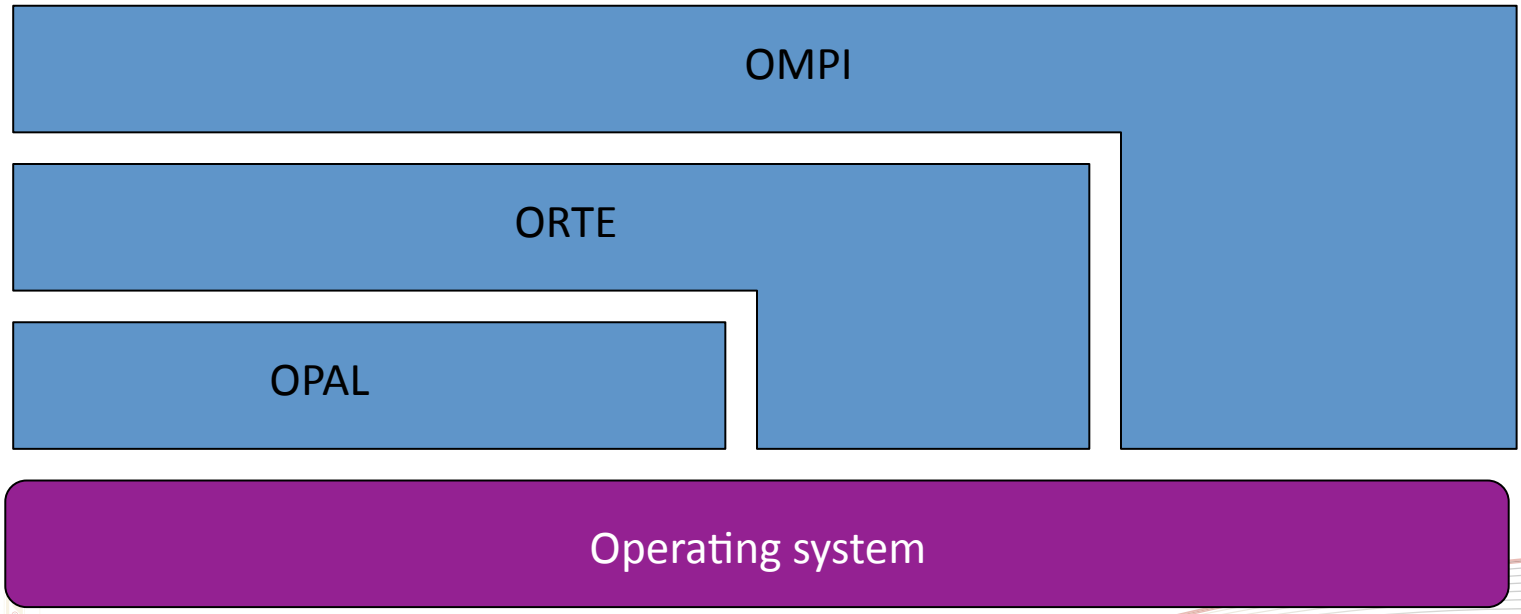
Performance

- "Good" performance defaults
  - …no such thing as a "reasonable default" that will work everywhere
  - Balance between scalability and performance
  - Can't always have both!
- Hence: run-time tunable parameters
  - Allows per-system and per-application tuning
  - More on this later

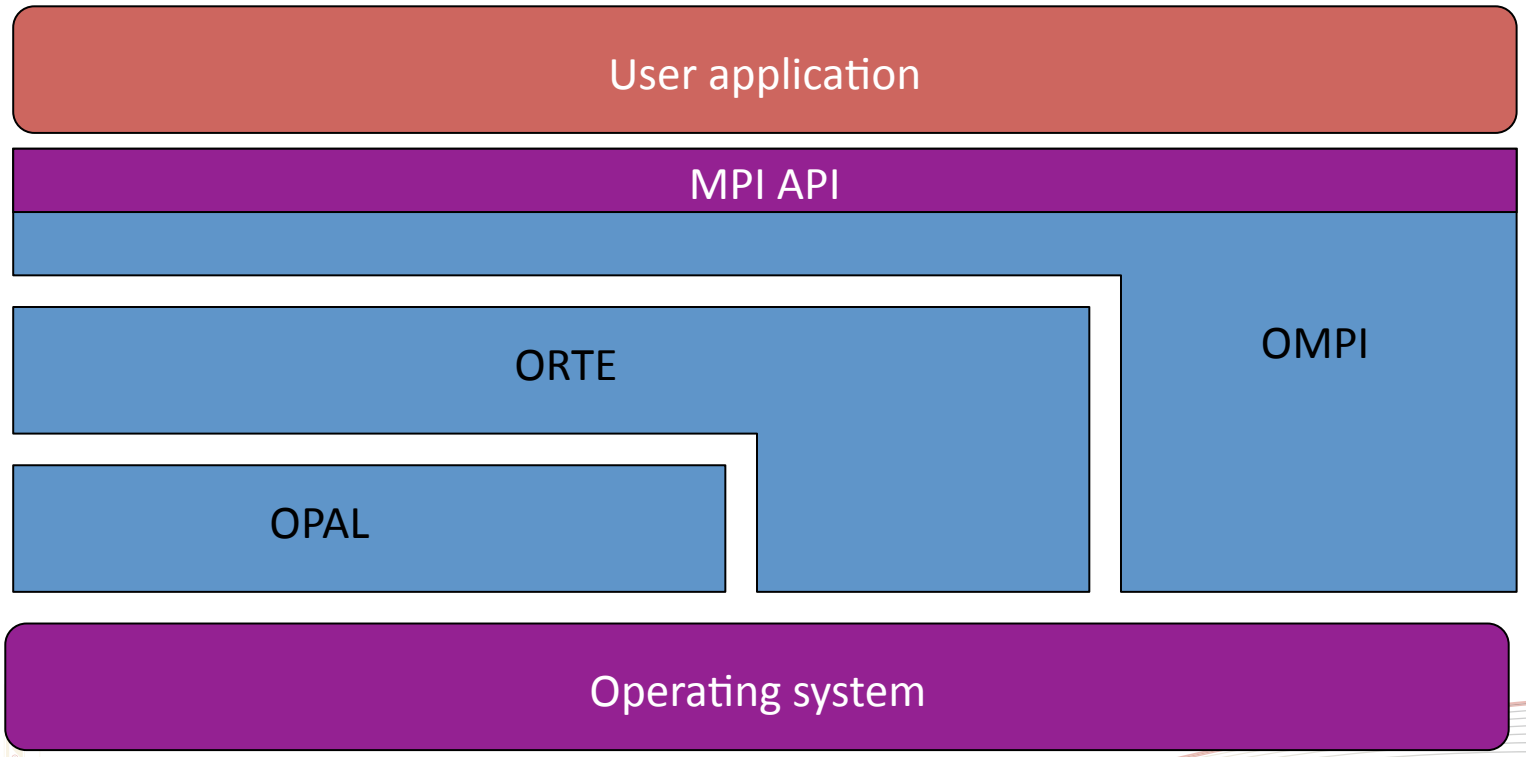
Three Main Code Sections

- Open MPI layer (OMPI)
  - Top-level MPI API and supporting logic
- Open MPI Run-Time Environment (ORTE)
  - Interface to back-end run-time system
- Open Portability Access Layer (OPAL)
  - OS / utility code (lists, reference counting, etc.)
- Dependencies - not layers
  - OMPI → ORTE → OPAL

Three Main Code Sections



Three Main Code Sections



MCA Parameters

- Run-time tunable values
  - Per layer
  - Per framework
  - Per component ("plugin")
- Change behaviors of code at run-time
  - Does *not* require recompiling / re-linking
- Simple example
  - Choose which network to use for MPI communications

MCA Parameter Lookup Order

1. mpirun command line

```
mpirun --mca <name> <value>
```

2. Environment variable

```
export OMPI_MCA_<name>=<value>
```

3. File
   - $HOME/.openmpi/mca-params.conf
   - $prefix/etc/openmpi-mca-params.conf

   (these locations are themselves tunable)
4. Default value

So Much Information…

- Open MPI has:
  - ~30 frameworks
  - 100+ components
  - Each component has run-time tunable parameters

- How to know what to use / how to use it?

ompi_info Command

- Tells everything about OMPI installation
  - Finds all components and all params
  - Great for debugging
- Can look up specific component

```
ompi_info --param <type> <plugin>
```

  - Shows parameters and current values
  - Can also use keyword “all”
- “--parsable” option

Example: Specify BTL

- BTL: Byte Transfer Layer
  - Framework for MPI point-to-point communications
  - Select which network to use for MPI communications

```
mpirun --mca btl tcp,self -np 4 ring_c
```

- Framework-level MCA parameter
  - Specifies which components to load

Example: Specify TCP BTL

```
mpirun --mca btl tcp,self -np 4 ring_c
```

- Components
  - tcp: TCP sockets
  - self: Loopback (send-to-self)

Example: Specify openib BTL

```
mpirun --mca btl openib,self -np 4 ring_c
```

- Components
  - openib: OpenFabrics verbs (InfiniBand)
  - self: Loopback (send-to-self)

Example: Specify sm+openib BTLs

```
mpirun --mca btl sm,openib,self -np 4 ring_c
```

- Components
  - openib: OpenFabrics verbs (InfiniBand)
  - self: Loopback (send-to-self)
  - sm: Shared memory (on-host communication)

What Does This Do?

```
mpirun -np 4 ring_c
```

What Does This Do?

```
mpirun -np 4 ring_c
```

- Use all available components
  - tcp, sm, openib, …
- TCP too?
  - Yes -- and no
  - TCP will automatically disable itself in the presence of low latency components (e.g., openib)

What Does This Do?

```
mpirun -np 4 ring_c
```

- More specifically:
  - Open each BTL component
  - Query if it wants to be used
  - Keep all that say “yes”
  - Rank by bandwidth and latency rating

What Does This Do?

```
mpirun -np 4 --mca btl ^tcp ring_c
```

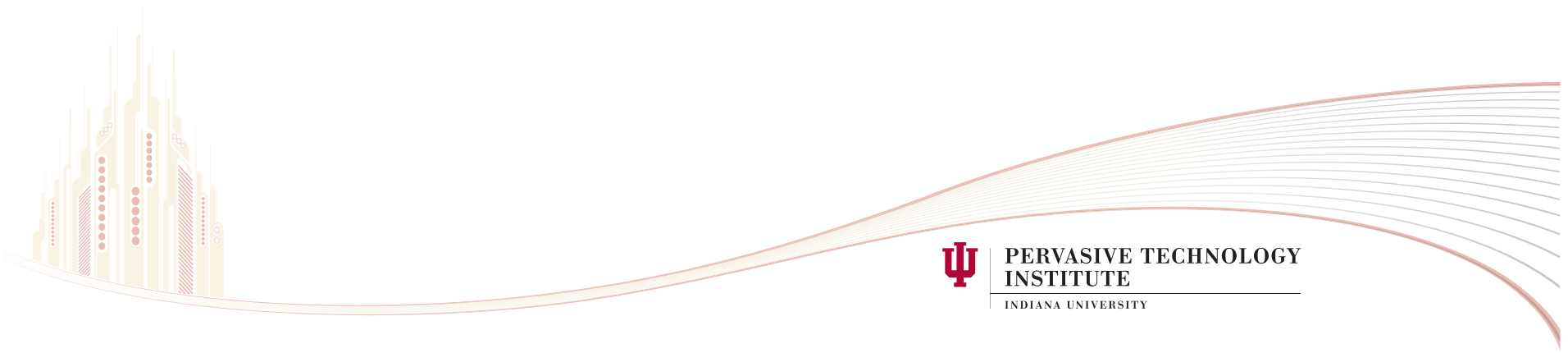
What Does This Do?

```
mpirun -np 4 --mca btl ^tcp ring_c
```

- Use all available components except tcp
- More specifically:
  - Open each BTL component except tcp
  - Query if it wants to be used
  - Keep all that say “yes”
  - Rank by bandwidth and latency rating

What Does This Do?

```
man MPI_Send
```



What Does This Do?

```
man MPI_Send
```

- Trick question!
- Shows Open MPI's excellent man pages
  - MPI API specification
  - And Open MPI-specific information

- Back to our regularly scheduled programming…

Specifying TCP Networks

```
mpirun --mca btl_tcp_if_include eth1 \
       --mca oob_tcp_if_include eth0 …
```

- Force all of Open MPI's TCP communications onto specific networks
  - Good for complex TCP scenarios
  - Also have "…if_exclude" forms
  - Can only use "include" or "exclude" – not both

Building MPI Applications

- Use "wrapper" compilers
  - Adds in MPI compiler / linker flags
  - Then invokes underlying compiler
  - Does *not* actually compile the program
- "--showme" option

```
mpicc --showme
mpicc hello_c.c -o hello_c -g \
      --showme
```

Other Languages

- mpiCC (only on case-sensitive filesystems)
  - a.k.a. mpic++
  - a.k.a. mpicxx
- mpif77
- mpif90
- All do similar functions

mpirun, mpiexec, Oh My!

- mpirun and mpiexec
  - Completely identical (in Open MPI)
- General form

```
mpirun [-np X] your_app
```

- If not using a scheduler, need a hostfile

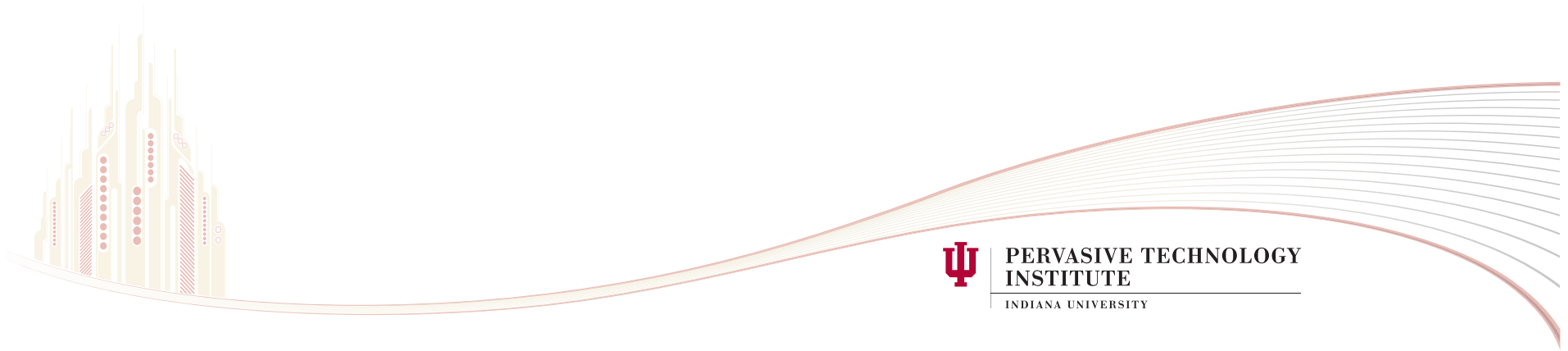
```
mpirun [-np X] --hostfile hostfile
your_app
```

- If using a scheduler, no need for hostfile or -np

Run “Hello World”

- Now run

```
$ mpirun -np 4 --hostfile \
    my_hostfile hello_c
Hello, world!  I am 0 of 4
Hello, world!  I am 1 of 4
Hello, world!  I am 2 of 4
Hello, world!  I am 3 of 4
```



Run-Time Tuning

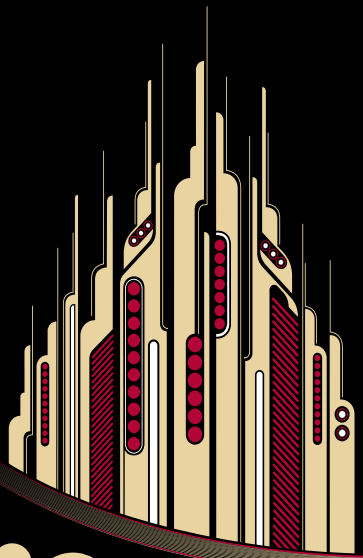
- There are many MCA parameters
- Which to use depends on:
  - Your local setup: scheduler, network, etc.
  - Your environment: sysadmin defaults, etc.

Open MPI Tutorial: Fault Tolerance

Joshua Hursey
Open Systems Lab.
Indiana University
jjhursey@osl.iu.edu

Abhishek Kulkarni
Open Systems Lab.
Indiana University
adkulkar@cs.indiana.edu

http://osl.iu.edu/research/ft

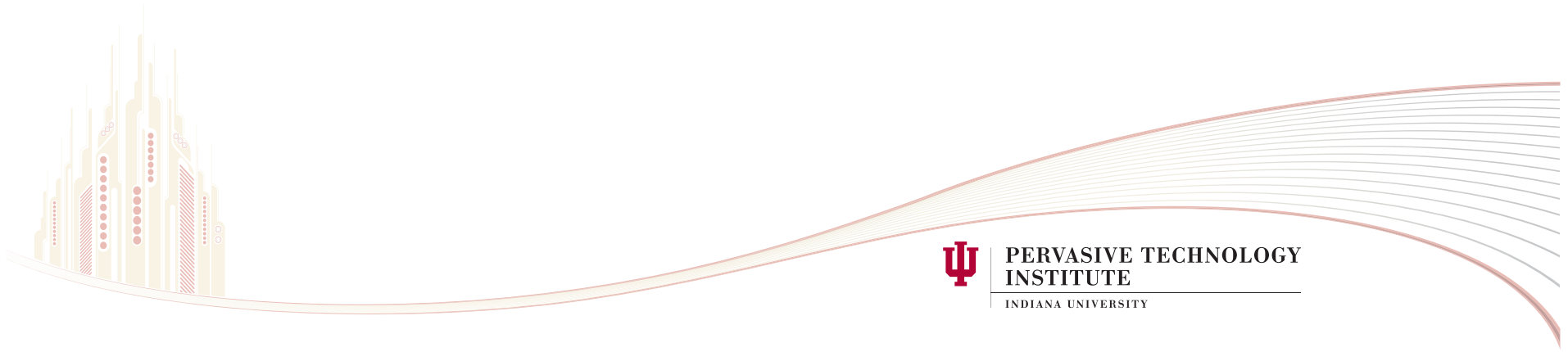


Look to the future of high-performance computing.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

SC Portland, OR 2009

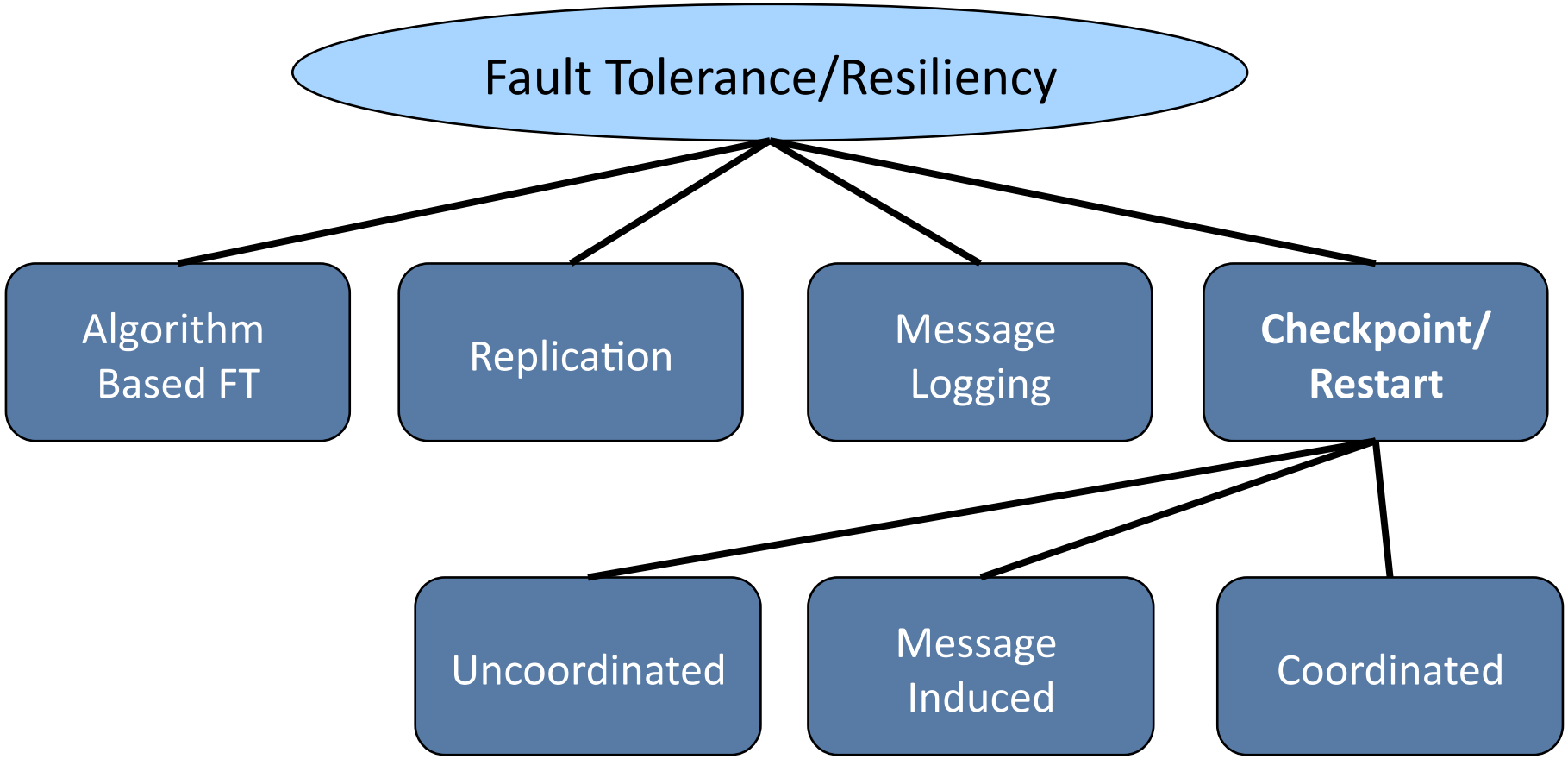
The unfortunate reality of next generation HPC environments is the high probability of process loss due to hardware failure for large scale and/or long running scientific applications.

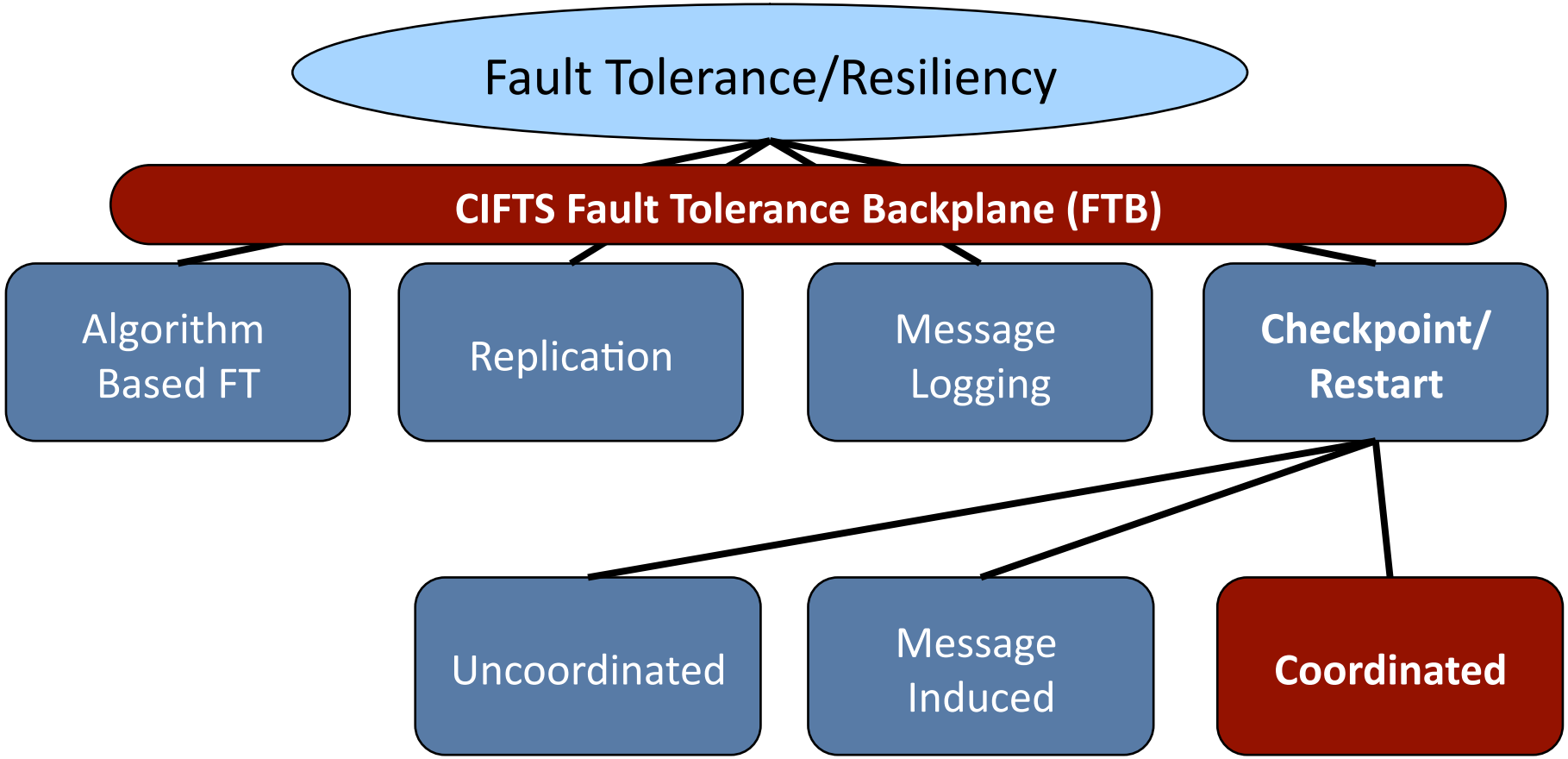


Fault Tolerance/Resiliency

The unfortunate reality of next generation HPC environments is the high probability of process loss due to hardware failure for large scale and/or long running scientific applications.

- Two ways of looking at the solution space
  - **Reactive**: Preparing for unexpected failure
  - **Proactive**: Anticipating near future failure


Fault Tolerance/Resiliency
Algorithm Based FT
Replication
Message Logging
Checkpoint/ Restart
Uncoordinated
Message Induced
Coordinated



Fault Tolerance, Process Migration, Parallel Debugging

MPI Transparency

- Focus on application transparent solutions
  - MPI interface restricts our ability to interact with the application for FT purposes
- Transparent solutions under investigation:
  - Coordinated checkpoint/restart
  - CIFTS Fault Tolerance Backplane
  - Network failover, integrity checking
  - Message logging

Current MPI 2.2 Standard

- 2.8 Error Handling (MPI Terms & Conventions)

  MPI provides the user with ***reliable message transmission***. A message sent is always received correctly, and the user does not need to check for transmission errors, time-outs, or other error conditions.

- 8.3 Error Handling (MPI Env. Mgmt.)

  "After an error is detected, the ***state of MPI is undefined***. … An MPI implementation is free to allow MPI to continue after an error ***but is not required to do so.***"

  "A good quality implementation will, to the greatest possible extent, circumscribe the impact of an error, so that normal processing can continue after an error handler was invoked."

Fault Tolerance Working Group

Investigate extensions to the MPI standard to enable the implementation of portable Fault Tolerant solutions for MPI based applications.

- Error Handling & Reporting
- Datatype Piggybacking
- Quality of Service & Performance
- Transactional Messages
- Quiescence Interface
- Process (re-)creation

Fault management & recovery

Aside: MPI-2.2 and MPI-3.0

- MPI-2.2 is complete, September 2009
  - $25 printed books (647 pages, $0.04/page!)
  - Take it home with you! ☺
  - **HLRS booth #2245**
- The MPI Forum wants your feedback
  - User survey: http://mpi-forum.questionpro.com/
  - Password: mpi3

Integrating Open MPI with the CIFTS Fault Tolerance Backplane (FTB)

Abhishek Kulkarni
Indiana University
adkulkar@cs.indiana.edu

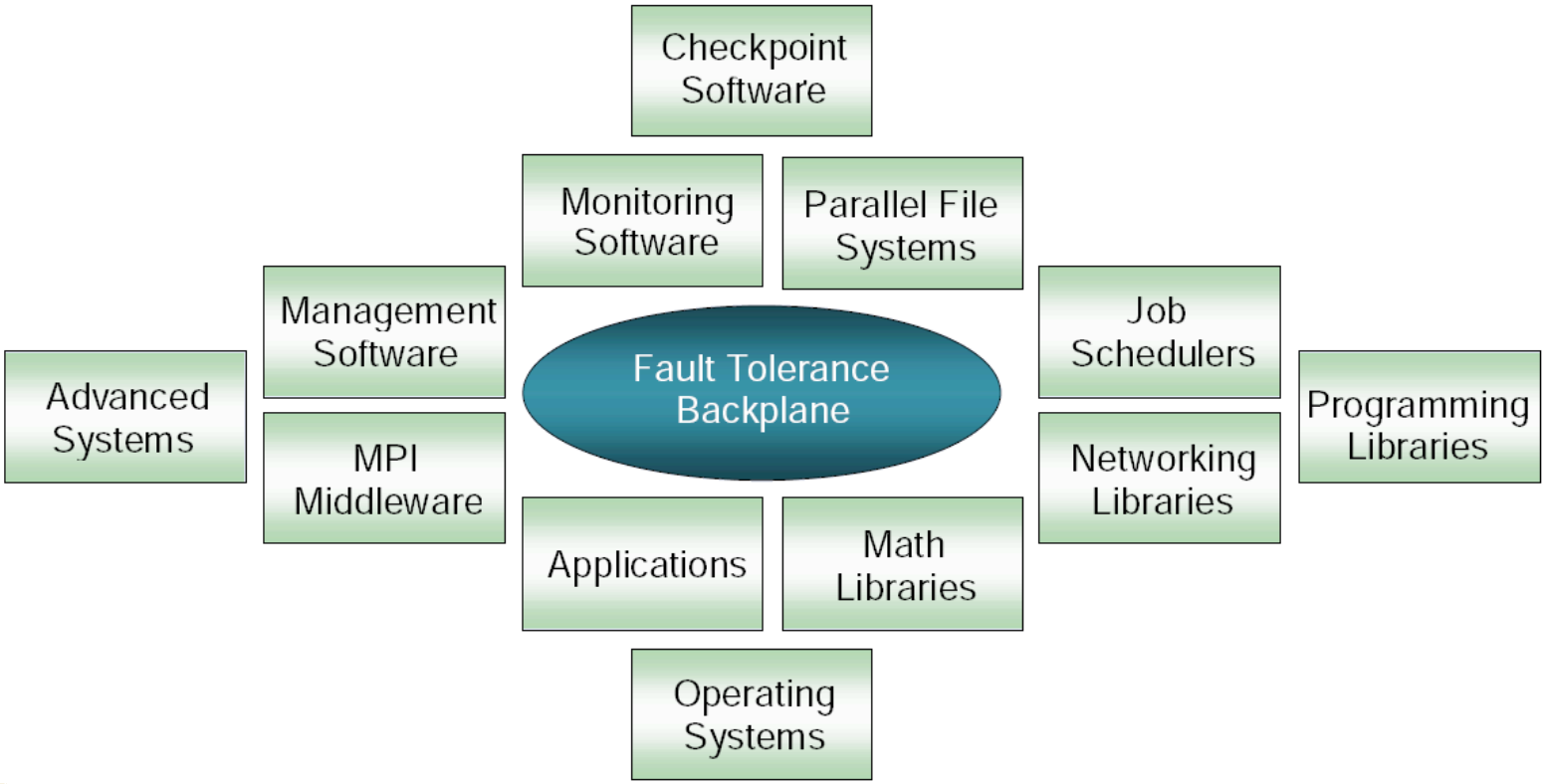
http://osl.iu.edu/research/ft

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

CIFTS Fault Tolerance Backplane

- Coordinated Infrastructure for Fault Tolerant Systems (CIFTS)
- Fault Tolerance Backplane (FTB)
  - Fault awareness and notification backplane to provide uniform event handling and notifications
  - Common interface specification for components to utilize the backplane
  - Based on a publish/subscribe interface
  - Components express their interest in “catching” or “throwing” a particular type of system event

Fault Tolerance Backplane



FTB facilitates the exchange of fault related information among the various systems software components in the HPC software ecosystem.

FTB support in Open MPI

- Transparent interfaces to “catch” and “throw” MPI-specific events to the FTB
- Implemented as an ORTE *notifier* component
- Event specification through a configurable Open MPI FTB events schema file
- Leverages the error reporting framework OPAL SOS to intercept Open MPI errors and relay them to the FTB.
- Targeted towards Open MPI 1.5 release

Open MPI FTB events

FTB Event	Severity	Description
MPI_INIT	info	Initialize the MPI execution environment
MPI_FINALIZE	info	Finalize the MPI execution environment
MPI_NODE_DEAD	error	Node X is unreachable
MPI_NODE_RESTORED	info	Node X is back to service
MPI_RANK_DEAD	error	Rank Y (on Node X) is presumably dead
MPI_RANK_RESTORED	info	Rank Y (on Node X) is back to service
MPI_NODE_MIGRATE_DONE	info	Ranks migrated from Node X to Node Q
MPI_JOB_ABORTED	error	MPI Job Z has been aborted
MPI_JOB_RESUMED	info	MPI Job Z has been resumed
MPI_MSG_CORRUPT	error	Message corruption on interface P
MPI_IFACE_DEAD	error	Mark physical interface P as dead
MPI_IFACE_RESTORED	info	Add P to available physical interfaces
MPI_PROC_DEAD	error	Process P on Rank Y (on Node X) is dead

FTB support in Open MPI

1. Configuring Open MPI to use the FTB backplane

Terminal

```
$ ./configure --with-ftb=/opt/
```

--with-ftb

This configure option specifies the path to the installation of the FTB library. Care should be taken to ensure that the path specified to the FTB libraries is correct so that the Open MPI installation can locate the FTB shared libraries.

--with-ftb-libdir

This configure option specifies the library path to the installation of the FTB library.

```
$ ./configure –with-ftb=/opt/ftb --with-ftb-libdir=/opt/ftb/lib64
```

FTB support in Open MPI

2. Running Open MPI to use the FTB backplane

- Start the FTB database server and the FTB agent
- Run the MPI application by enabling the Open MPI FTB notifier component

Terminal

```
shell$ mpirun –np 16 –mca notifier ftb my-app
```

```
--mca notifier_ftb_subscription_style
Set the subscription style of the Open MPI FTB client. This dictates
the way in which the Open MPI client interacts with the FTB daemons.

shell$ mpirun –mca notifier_ftb_subscription_style
“FTB_SUBSCRIPTION_NOTIFY” <args> my-app
```

```
--mca notifier_ftb_priority
Set the priority of the Open MPI FTB notifier component. The
component with the highest priority is given preference.
```

ORTE notifier framework

- Notifies events of varying severities
- Per-component severity thresholds can be specified in runtime
- Existing notifier components include
  - syslog, smtp, command, hnp, ftb, file, twitter(!)

Example

```
orte_notifier.help (ORTE_NOTIFIER_INFRA, ORTE_ERR_COMM_FAILURE,
                        "help-mpi-btl-openib.txt",
                        BTL_OPENIB_QP_TYPE_PP(qp) ?
                        "pp rnr retry exceeded" :
                        "srq rnr retry exceeded",
                        orte_process_info.nodename, device_name,
                        peer_hostname);
```

ORTE FTB notifier component

- FTB notifier component maps events from the Open MPI event space to the common MPI event space defined by the FTB
- Events are thrown in the "FTB.MPI.OPENMPI" event space

ORTE ERROR	CORRESPONDING FTB ERROR
ORTE_ERR_OUT_OF_RESOURCE	MPI_OUT_OF_RESOURCE
ORTE_ERR_TEMP_OUT_OF_RESOURCE	
ORTE_ERR_COMM_FAILURE	MPI_COMM_FAILURE

OPAL SOS

- Enhanced error event reporting
- Reduce cascading error messages
- Build and aggregate stacks of error events offering explicit control over event history
- Associate and define relationships between distinct SOS events
- Allows registration of custom callbacks to intercept events
- Transparently relay events to the notifier components

Using OPAL SOS

- Register a failure event into the SOS framework with severity “ERROR”.

Example

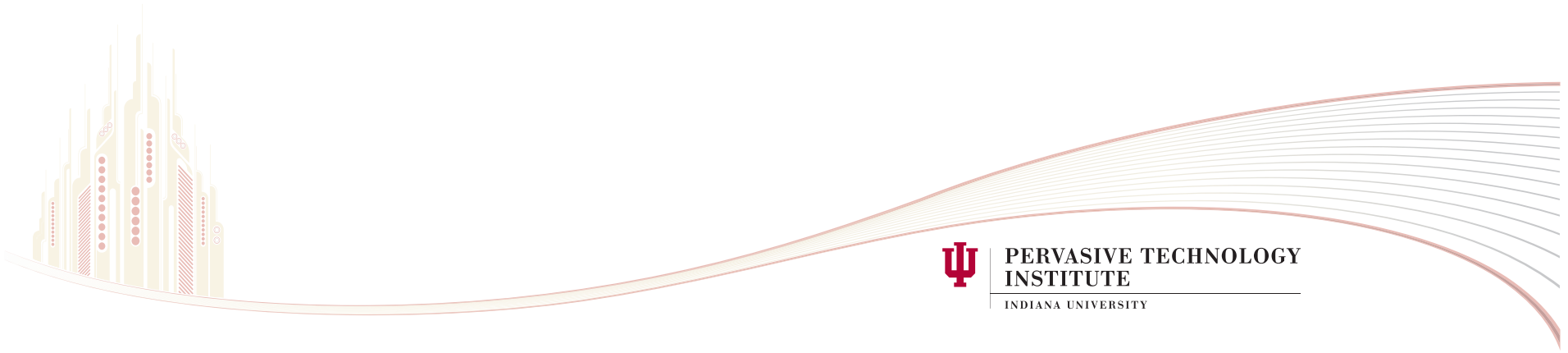
```
errcode = OPAL_SOS_ERROR((ORTE_ERR_PROC_DEAD, false,
                          ”Errmgr Called job=%s,rank=%u”
                          “,proc=%s,state=0x%x\n",
                          ORTE_JOBID_PRINT(jdata->jobid),
                          orte_ess.proc_get_daemon(proc),
                          ORTE_NAME_PRINT(proc), state));
```

- Print the registered error

```
OPAL_SOS_PRINT(errcode, false)

------------------------------------------------------------------------
| |--<ERROR> at recos_ignore_module.c:195:recos_ignore_errmgr_restart():
| | Errmgr Called job=[19308,1],rank=0,proc=[[19308,1],2]],state=0x400
| |
```

Distributed Ray-tracing using POVRAY and FTB-enabled Open MPI

Demo!



Future Work

- Extended mapping of events between Open MPI event space and FTB MPI event space
- Standardize the payloads for events in the FTB MPI event space
- Integrate with other Open MPI FT and resiliency components
- Fault predictor and autonomic components internal to Open MPI

http://www.osl.iu.edu/research/ft/cifts/

Transparent Checkpoint/Restart

Joshua Hursey
Indiana University
jjhursey@osl.iu.edu

http://osl.iu.edu/research/ft

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

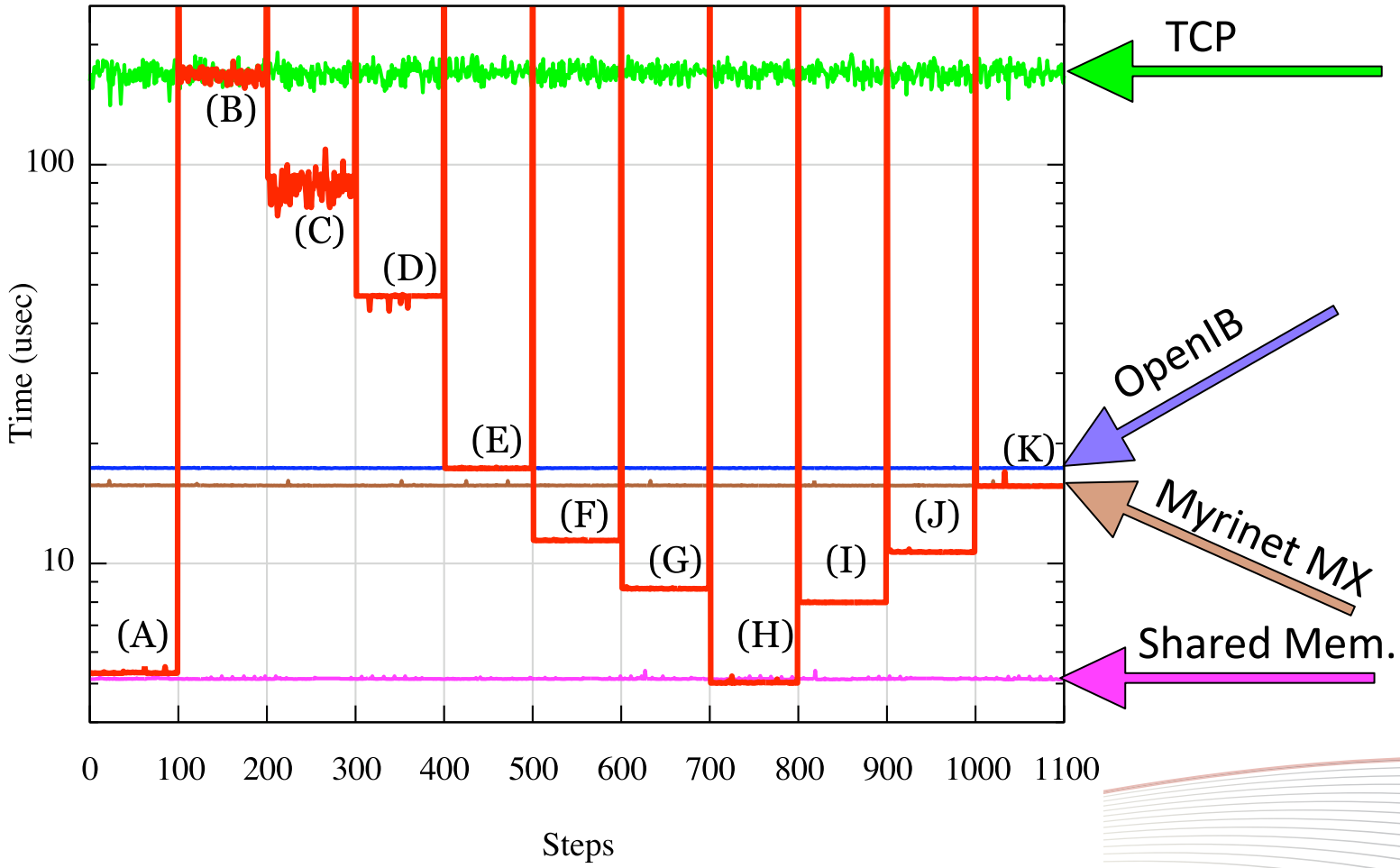
Checkpoint/Restart in Open MPI

- Application transparent, fully coordinated checkpoint/restart
- Bookmark exchange coordination protocol
- C/R service support
  - BLCR
  - *SELF*
  - Others in development…
- Interconnect support
  - Ethernet
  - Myrinet
  - InfiniBand
  - Shared Memory
- Dynamically adapt to network availability
- **Available in v1.3 release**

Hursey, J., et. al., *The design and implementation of checkpoint/restart process fault tolerance for Open MPI*. IEEE IPDPS, 2007.

Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

Network Reconfiguration



Building with Checkpoint/Restart (Default: Compiled Out)

- Enable checkpoint/restart (cr)

```
$ ./configure --with-ft=cr
```

```
WARNING: ************************************************
WARNING: *** Fault Tolerance Integration into Open MPI is *
WARNING: *** a research quality implementation, and care  *
WARNING: *** should be used when choosing to enable it.   *
WARNING: ************************************************
```

- Enable checkpoint helper thread

```
$ ./configure --enable-mpi-threads \
              --enable-ft-thread
```

```
WARNING: ************************************************
WARNING: *** Fault Tolerance with a thread in Open MPI    *
WARNING: *** is an experimental, research quality option. *
WARNING: *** It requires progress or MPI threads, and     *
WARNING: *** care should be used when enabling these      *
WARNING: *** options.                                     *
WARNING: ************************************************
```

Using Checkpoint/Restart (Default: Disabled)

Terminal 1

```
shell$ mpirun –np 16 –am ft-enable-cr my-app
```

Terminal 2

```
shell$ ompi-checkpoint 1234
Snapshot Ref.: 0 ompi_global_snapshot_1234.ckpt
shell$ ompi-checkpoint 1234
Snapshot Ref.: 1 ompi_global_snapshot_1234.ckpt
```

Sequence Numbers

Global Snapshot Reference

```
shell$ ompi-restart ompi_global_snapshot_1234.ckpt
```

Hursey, J., et. al., *The design and implementation of checkpoint/restart process fault tolerance for Open MPI*. IEEE IPDPS, 2007.

Using Checkpoint/Restart: MPI Suspend/Resume

Terminal 1

```
shell$ mpirun –np 16 –am ft-enable-cr my-app
```

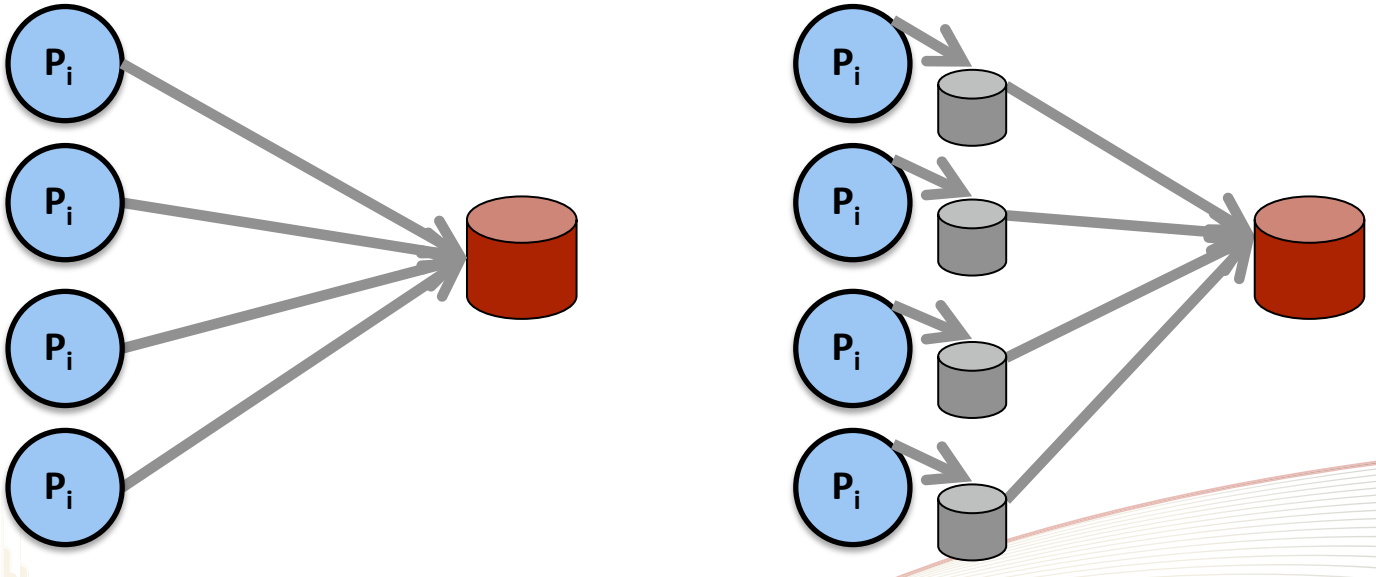
Terminal 2

```
shell$ ompi-checkpoint --stop -v 1234
[localhost:001300] [  0.00 /   0.20]  Requested - ...
[localhost:001300] [  0.00 /   0.20]  Pending   - ...
[localhost:001300] [  0.01 /   0.21]  Running   - ...
[localhost:001300] [  1.01 /   1.22]  Stopped   - ...
Snapshot Ref.: 0 ompi_global_snapshot_1234.ckpt

shell$ kill –CONT 1234
```

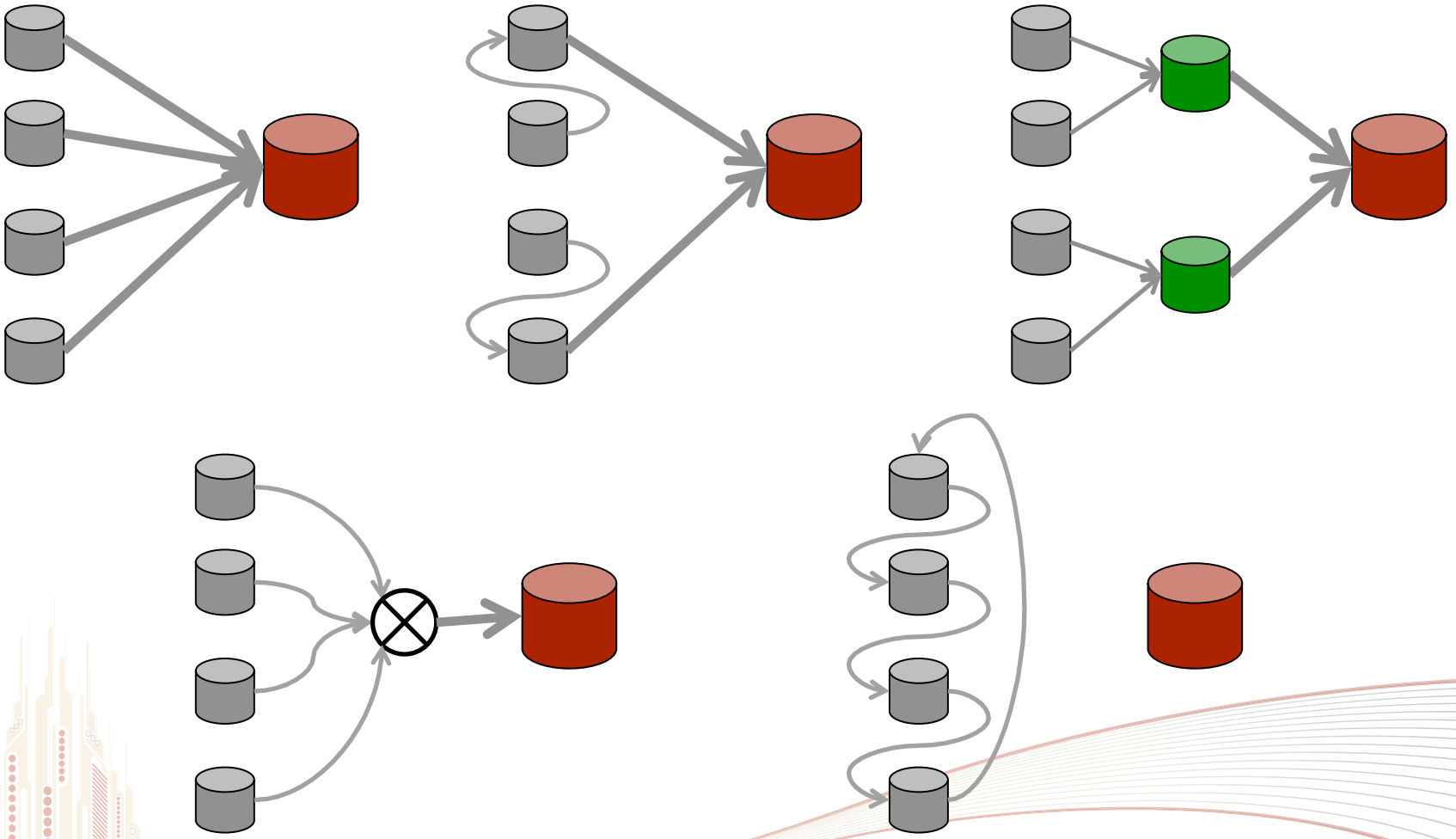
Hursey, J., et. al., *The design and implementation of checkpoint/restart process fault tolerance for Open MPI*. IEEE IPDPS, 2007.

Stable Storage

Any storage device that survives the maximum number of expected faults in the system



Stable Storage Opportunities



Customization: Stable Storage Location

Terminal 1

```
shell$ cat $HOME/.openmpi/mca-params.conf
# Remote snapshot directory (Globally mounted)
snapc_base_global_snapshot_dir=/home/me/checkpoints
```

Terminal 2

```
shell$ mpirun -np 16 -am ft-enable-cr my-app
...
```

Customization:
Local Staging to Stable Storage

Terminal 1

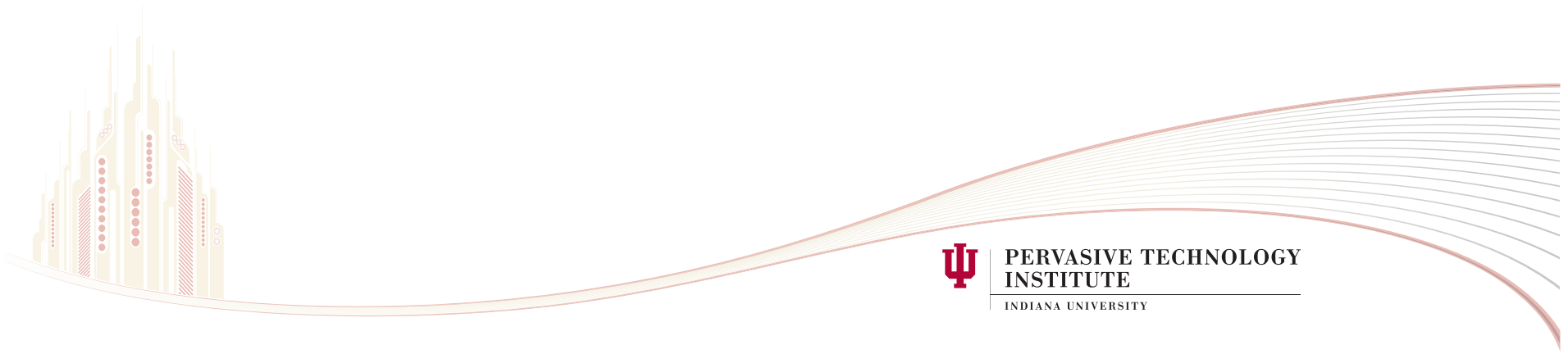
```
shell$ cat $HOME/.openmpi/mca-params.conf
# Remote snapshot directory (Locally mounted)
snapc_base_global_snapshot_dir=/tmp/me/global

# Local snapshot directory (Locally mounted)
crs_base_snapshot_dir=/tmp/me/local

# Stage locally then transfer in the background
snapc_base_store_in_place=0
```

Terminal 2

```
shell$ mpirun -np 16 -am ft-enable-cr my-app
...
```

Customization:
SELF Checkpoint/Restart Service

http://osl.iu.edu/research/ft/ompi-cr/examples.php



Failure Free Performance Impact

Latency

Interconnect	No C/R	With C/R	% Overhead
Ethernet (TCP)	49.92 µs	50.01 µs	0.2 %
InfiniBand	8.25 µs	8.78 µs	6.4 %
Myrinet MX	4.23 µs	4.81 µs	13.7 %
Shared Memory	1.84 µs	2.15 µs	16.8 %

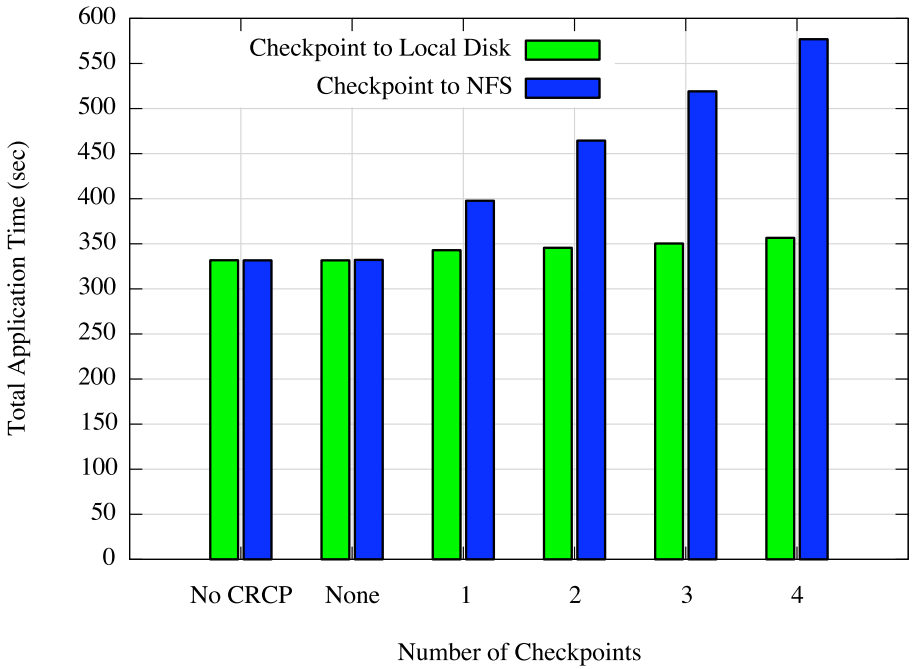
Bandwidth

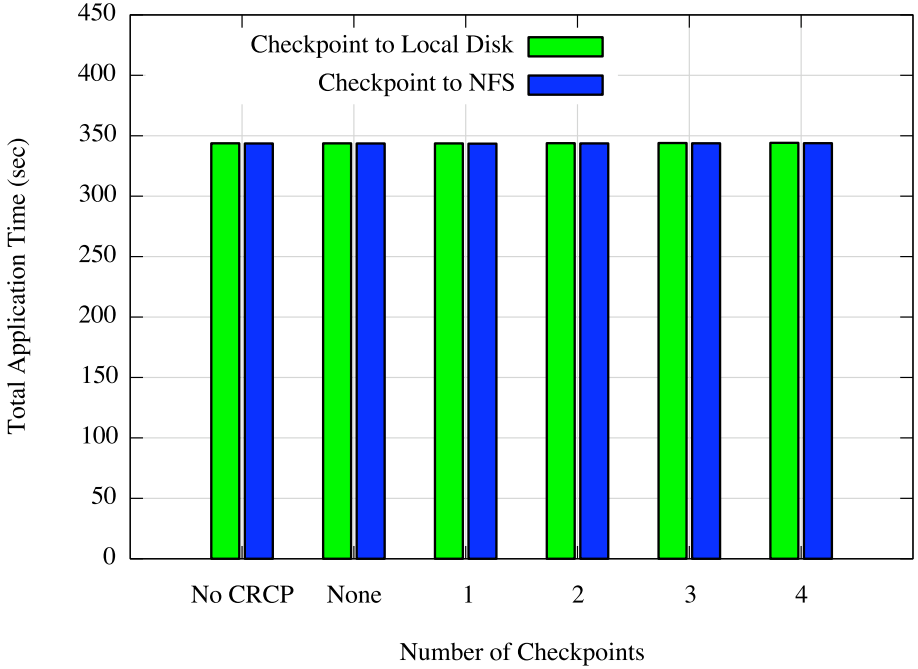
Interconnect	No C/R	With C/R	% Overhead
Ethernet (TCP)	738 Mbps	738 Mbps	0.0 %
InfiniBand	4703 Mbps	4703 Mbps	0.0 %
Myrinet MX	8000 Mbps	7985 Mbps	0.2 %
Shared Memory	5266 Mbps	5258 Mbps	0.2 %

NASA Parallel Benchmarks: 0 – 0.6 %

Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

Checkpoint Overhead



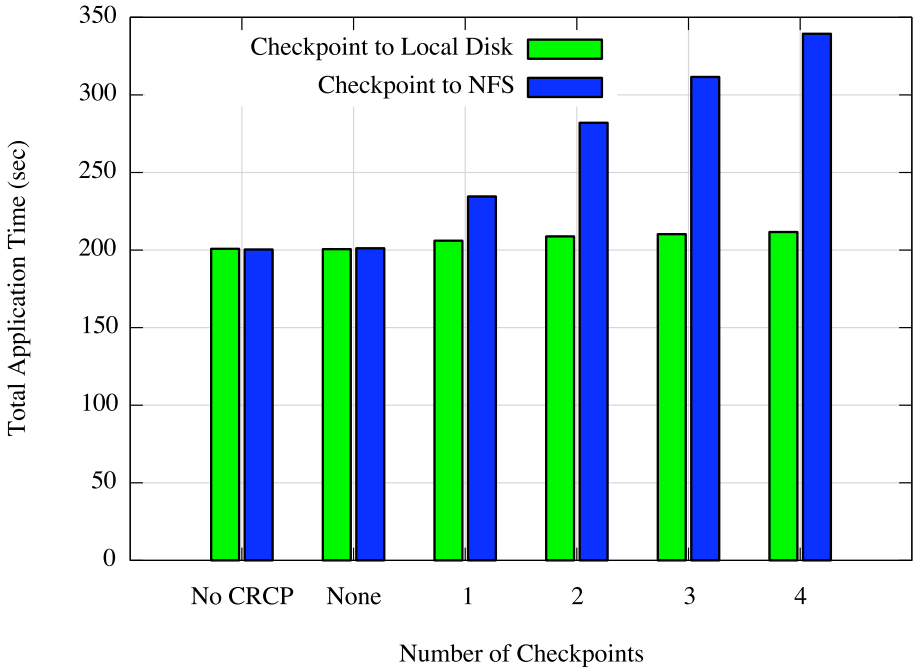


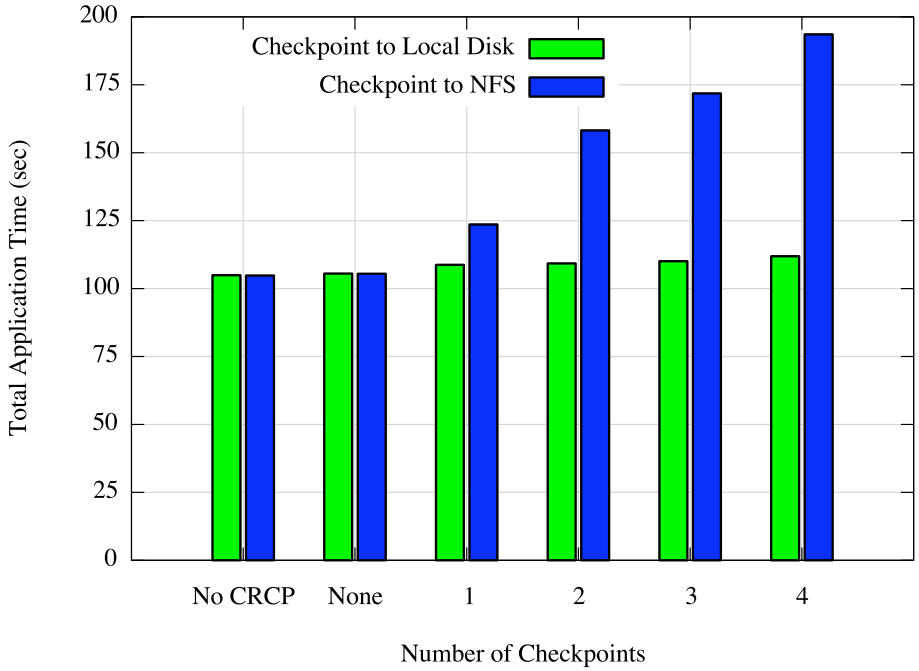
BT Class C 36 Procs
4.2 GB/120 MB

EP Class D 32 Procs
102 MB/3.2 MB

Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

Checkpoint Overhead



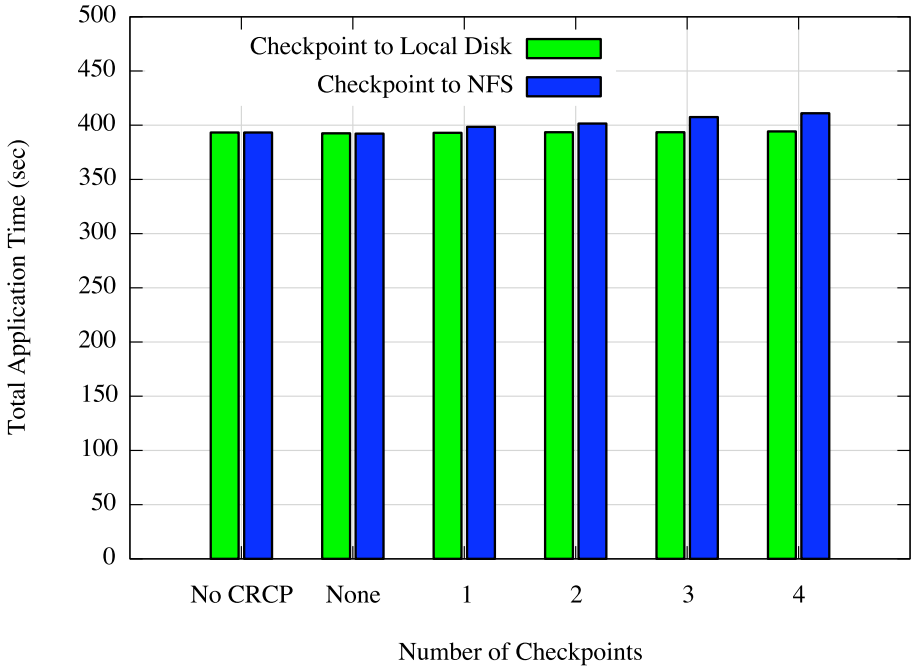


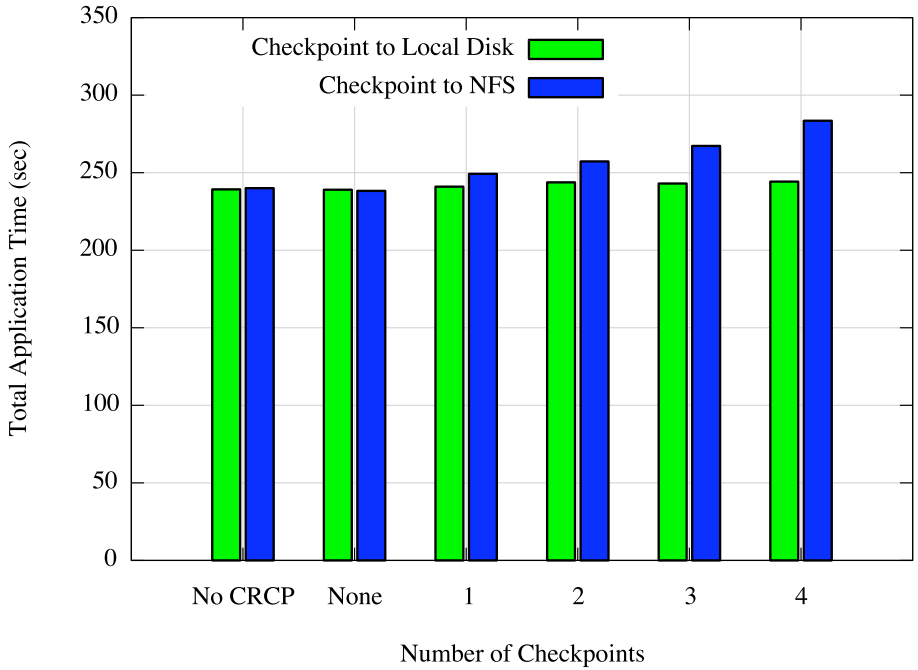
SP Class C 36 Procs
1.9 GB/54 MB

LU Class C 32 Procs
1 GB/32 MB

Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

Checkpoint Overhead





Gromacs (DPPC) 8 Procs
267 MB/33 MB

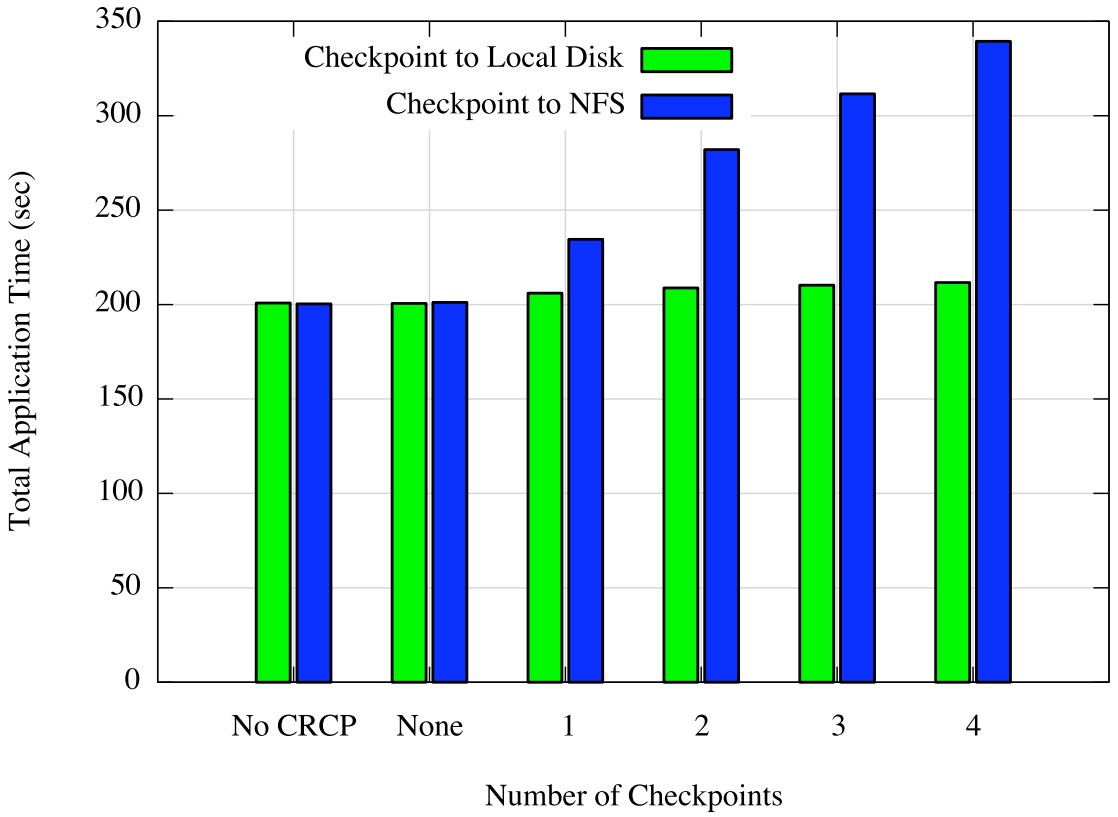
Gromacs (DPPC) 16 Procs
473 MB/30 MB

Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

Checkpoint Bottlenecks

98.8% File I/O
0.7% Modex
0.3% Coord. Protocol
0.2% Internal Coord.



Hursey, J., et. al., *Interconnect Agnostic Checkpoint/Restart in Open MPI*. ACM HPDC, 2009.

Future Directions

- Checkpoint/Restart Systems
  - User level, and improved application level support
  - Memory inclusion/exclusion
  - Incremental checkpointing
- Advanced storage techniques
  - Compression
  - Peer-based storage
  - Staggering and staging

Proactive Process Migration

Joshua Hursey
Indiana University
jjhursey@osl.iu.edu

http://osl.iu.edu/research/ft

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

Process Migration

The movement of a set of processes from one machine to another without residual dependencies

- Proactive Migration
  - Move processes when asked by predictor (e.g., CIFTS FTB, RAS, …)
- Cluster Management
  - Move processes when asked by an end user
- Load Balancing
  - Migrate when a load imbalance is detected

Process Migration Implementation

- Builds upon a checkpoint/restart infrastructure
  - State saved on one machine,
  - Transferred to another machine,
  - Restarted and rejoined to the computation
- Many types of process copy techniques available
  - **Eager**
  - Pre-copy
  - Lazy
  - Post-copy

Hursey, J., et. al., *The design and implementation of checkpoint/restart process fault tolerance for Open MPI*. IEEE IPDPS, 2007.

Recovery Service Framework

Policy enforcement for runtime fault recovery and preventative actions

- Policy variations:
  - **Abort:**
    Terminate job
  - **Ignore:**
    Stabilize and run without the failed process
  - **Migrate:**
    Preventatively move processes between resources
  - **Restart:** (*Automatic Recovery*)
    Automatically restart from the last checkpoint

Building with Process Migration Support
(No change from C/R Support)

- Enable checkpoint/restart (cr)

```
$ ./configure --with-ft=cr
```

```
WARNING: ************************************************
WARNING: *** Fault Tolerance Integration into Open MPI is *
WARNING: *** a research quality implementation, and care  *
WARNING: *** should be used when choosing to enable it.   *
WARNING: ************************************************
```

- Enable checkpoint helper thread

```
$ ./configure --enable-mpi-threads \
              --enable-ft-thread
```

```
WARNING: ************************************************
WARNING: *** Fault Tolerance with a thread in Open MPI    *
WARNING: *** is an experimental, research quality option. *
WARNING: *** It requires progress or MPI threads, and     *
WARNING: *** care should be used when enabling these      *
WARNING: *** options.                                     *
WARNING: ************************************************
```

Using Process Migration (Default: Disabled)

Terminal 1

```
shell$ mpirun –np 16 –am ft-enable-cr my-app
```

Terminal 2

```
shell$ ompi-migrate --off node01 123

shell$ ompi-migrate -v -x node01 --onto node02,node03 123
[localhost:01300] [  0.00 /   0.00]  Requested      - ...
[localhost:01300] [  0.00 /   0.00]  Running        - ...
[localhost:01300] [  0.00 /   0.00]  Checkpointing - ...
[localhost:01300] [  1.10 /   1.10]  Restarting     - ...
[localhost:01300] [  1.08 /   2.18]  Finished       - ...
```

Availability & Future Directions

- Availability:
  - Checkpoint/Restart: Available in the current v1.3
  - Process Migration - In development
    - Public release - Spring 2010 (v1.5 series)
- Future work:
  - Automatic recovery
  - Improved checkpoint file handling
  - Alternative copy techniques (e.g., pre-copy)
  - MPI application fault tolerance policies

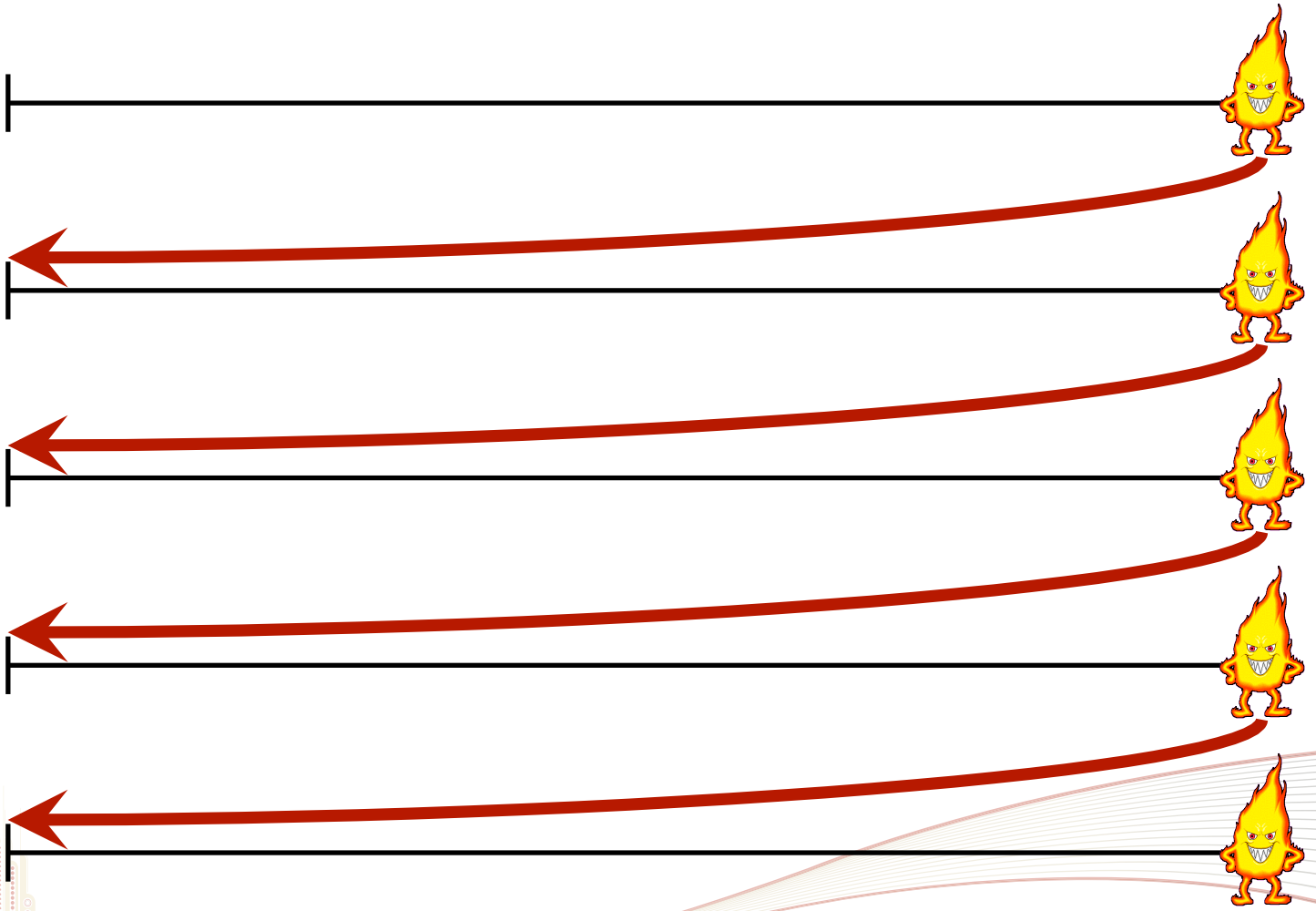
Checkpoint/Restart Enabled Parallel Debugging

Joshua Hursey
Indiana University
jjhursey@osl.iu.edu

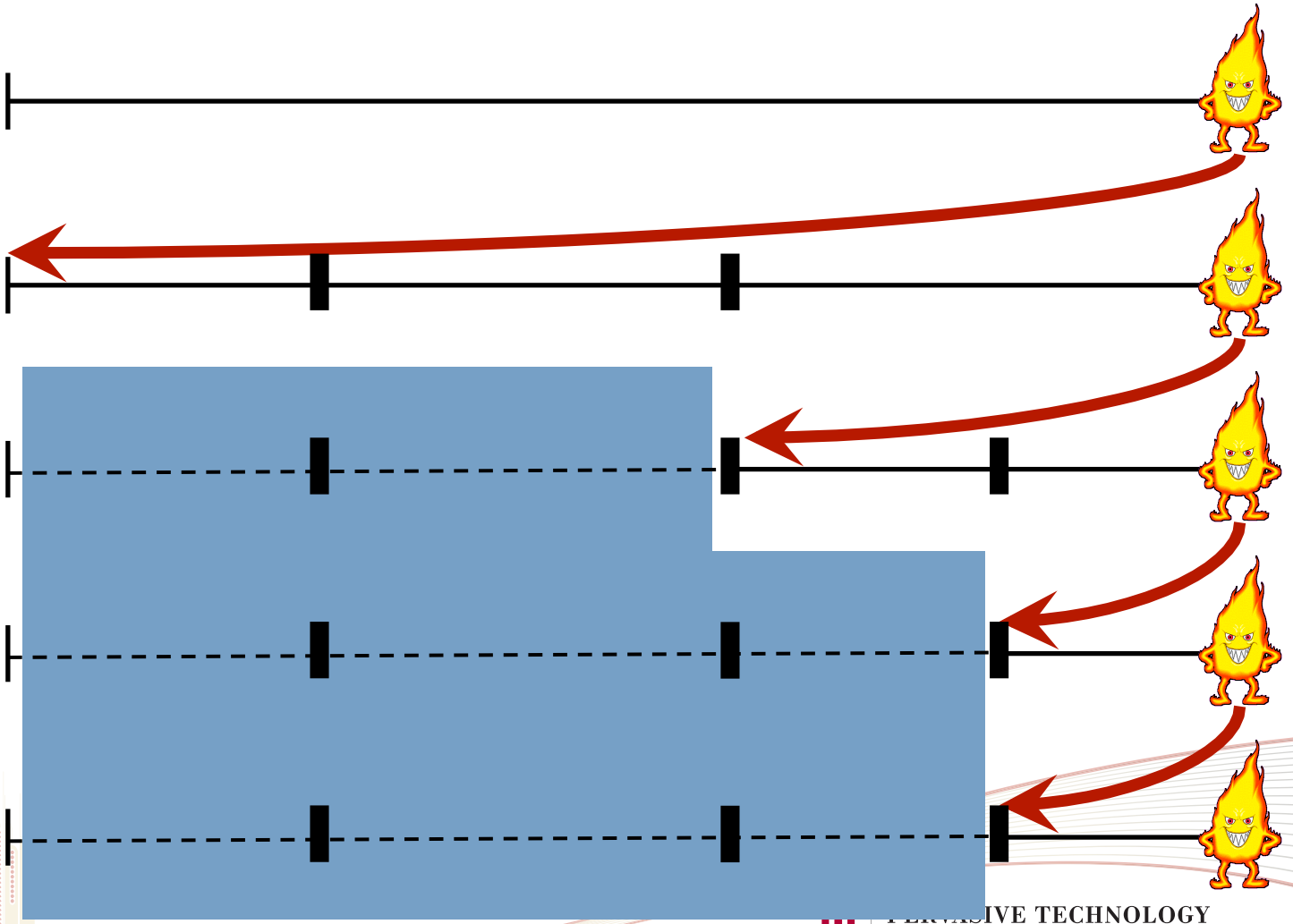
http://osl.iu.edu/research/ft

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

Cyclic Debugging



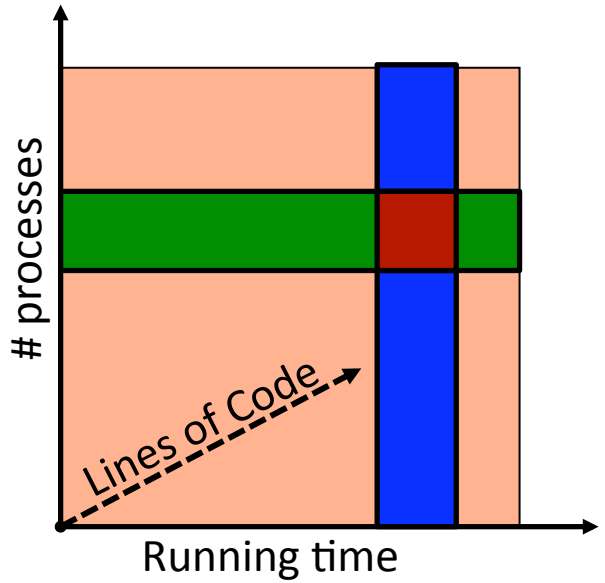
Checkpoint/Restart Enabled Debugging



Checkpoint/Restart Enabled Parallel Debugging

"My program only fails after ***4 hours*** when running with ***>512 processes***."

- Step-backward (a.k.a. reverse execution)
  - Combination of checkpoint/restart and message logging
- Specified a C/R interface for:
  - Parallel debugger,
  - C/R enabled MPI implementation,
  - Checkpoint/restart service



Hursey, J., et. al., *Checkpoint/Restart Enabled Parallel Debugging*. (under submission), 2009.

Building with Process Migration Support
(Little change from C/R Support)

- Enable checkpoint/restart (cr)

```
$ ./configure --with-ft=cr
```

- Enable c/r enabled parallel debugging

```
$ ./configure --with-ft=cr --enable-crdebug
```

- Enable checkpoint helper thread (*Optional*)

```
$ ./configure --enable-mpi-threads \
              --enable-ft-thread
```

Hursey, J., et. al., *Checkpoint/Restart Enabled Parallel Debugging*. (under submission), 2009.

Using Checkpoint/Restart Enabled Parallel Debugging

Terminal 1

```
shell$ mpirun –np 16 –am ft-enable-cr my-app
```

Watch the following site for more details:
http://osl.iu.edu/research/ft

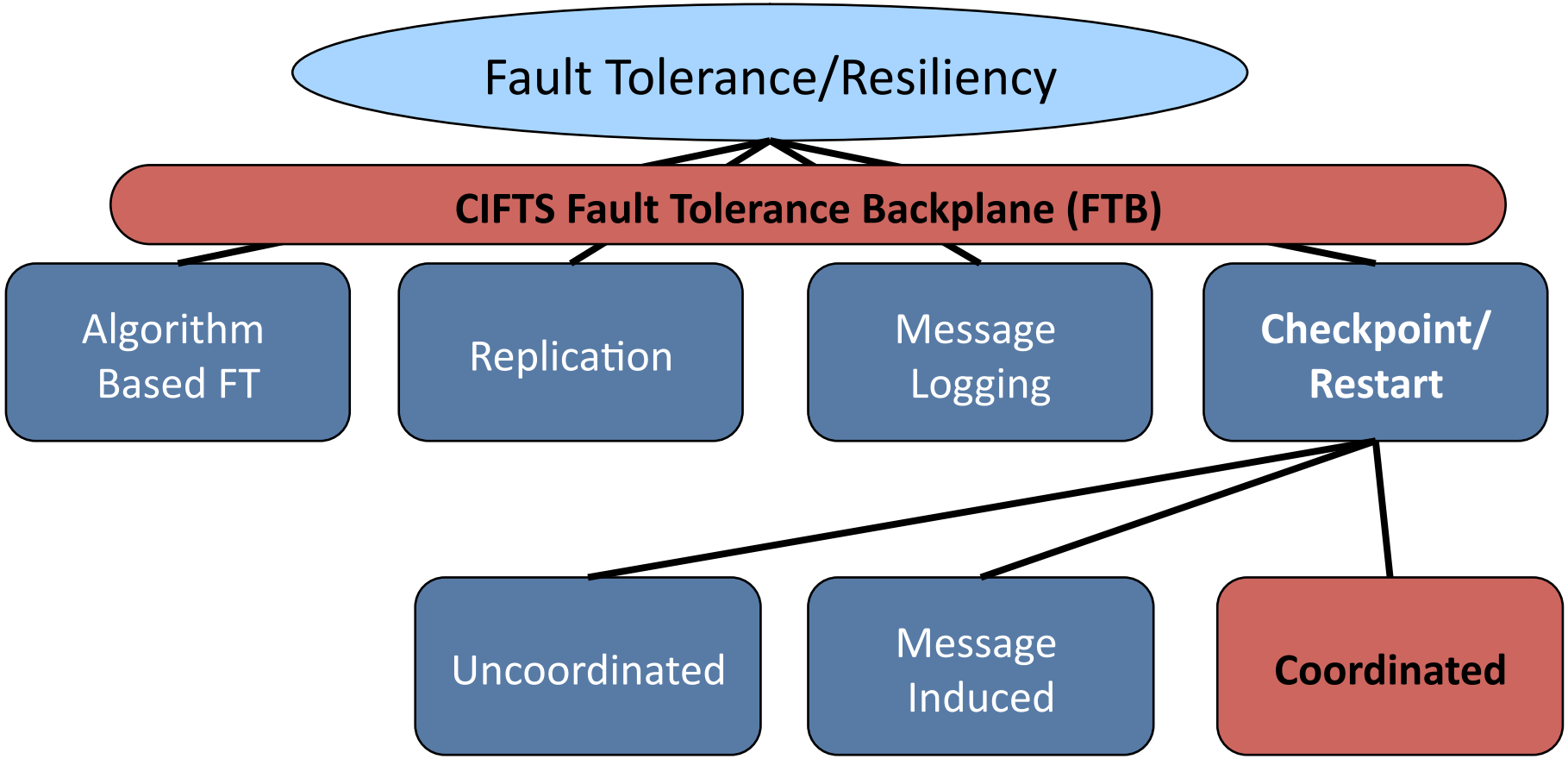
Hursey, J., et. al., *Checkpoint/Restart Enabled Parallel Debugging*. (under submission), 2009.

Availability

- Availability
  - Implementation finished, specification complete
  - Public release – Soon/Early 2010

- Watch the following site for more details:
  - http://osl.iu.edu/research/ft

Hursey, J., et. al., *Checkpoint/Restart Enabled Parallel Debugging*. (under submission), 2009.

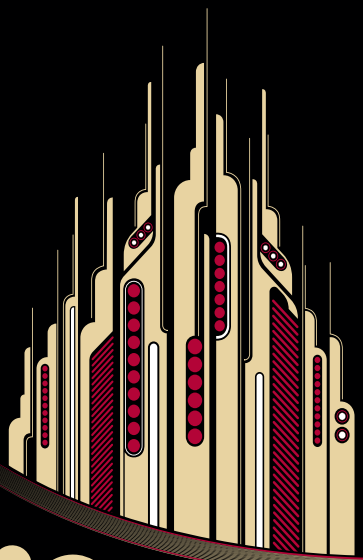


Fault Tolerance, Process Migration, Parallel Debugging

Open MPI Tutorial: Hacking Open MPI

Joshua Hursey
Open Systems Lab.
Indiana University
jjhursey@osl.iu.edu

http://osl.iu.edu/research/ft

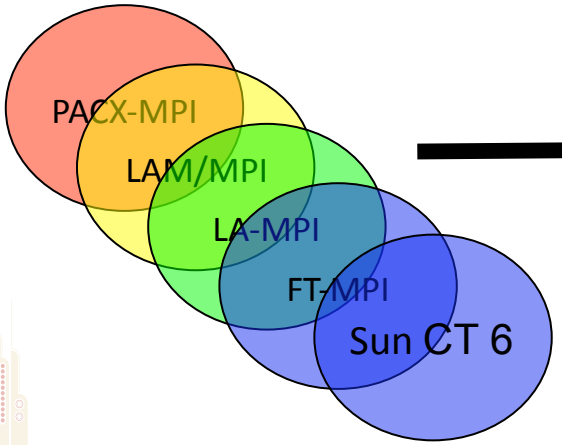


Look to the future of high-performance computing.

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

Open MPI

Combine **best practices** from previous MPI implementations into a single **open source**, **production quality**, **MPI-2 compliant** MPI implementation.



OPEN MPI

PERVASIVE TECHNOLOGY INSTITUTE
INDIANA UNIVERSITY

Open Source Project

- The Open MPI code base is open source
  - Anyone *can* fork, but we discourage that
  - There are too many MPI's already
- Does not exclude closed source
  - Can distribute closed-source plugins
  - Do not need to distribute Open MPI itself

Community

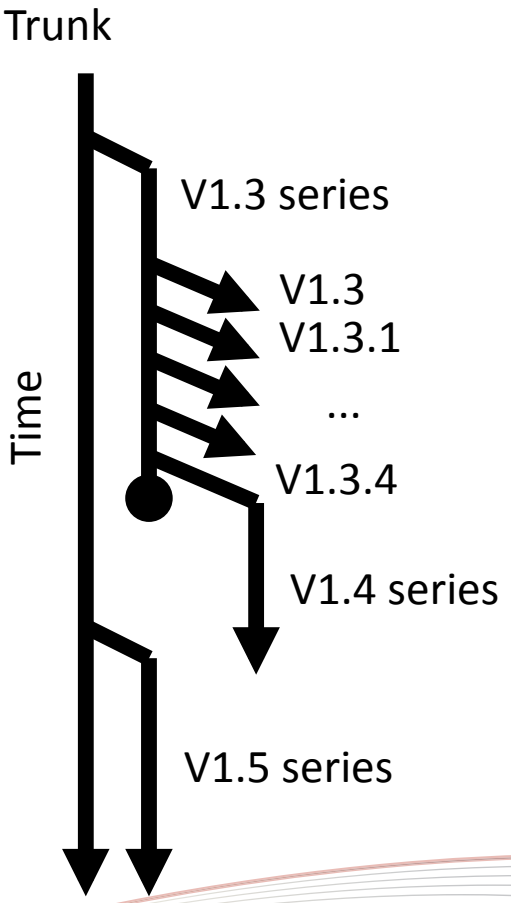
- Strong relationship with open source community
  - Open repository
  - Open mailing lists
  - Responsive to questions, problems
- Work with and for the HPC community

License

- Open MPI is licensed under the BSD
- All contributed code must be compatible with BSD
  - GPL is not compatible with BSD
  - One, top-level LICENSE file
- Always include all relevant notices when importing external source code

Subversion Repository

- /trunk open development
  - Head of development
- Development release series
  - Odd minor version numbers
  - /branches/v1.3, /branches/v1.5, …
- Stable release series
  - Even minor version numbers
  - /branches/v1.2, /branches/v1.4, …
- Tagged Releases
  - /tags/v1.2, /tags/v1.2.1, …



Nightly Regression Testing

- MPI Testing Tool (MTT)

  Infrastructure for automated, distributed testing

  http://www.open-mpi.org/projects/mtt

- Institutions volunteer testing resources
- Combine all testing results into a database
- Provide tools for testing and analysis
- Currently between 250K-500K tests per night

Hursey, J., et. al., *An Extensible Framework for Distributed Testing of MPI Implementations*. Euro PVM/MPI, 2007.

MPI Testing Tool

- Test any N MPI implementations
  - Each installed M different ways
    - Against T different test suites
      - Run each R different ways
- Multiplicative effect: N x M x T x R
- Fully automated (run via cron)
- Results go into a centralized database
  - Correctness and performance results
  - Available for historical data mining

MPI Testing Tool

- Supports "disconnected" scenarios
  - Download on one node
  - Compile / install on another
  - Run tests on another [cluster]
- Not yet released
  - Hope to be usable in near future
  - Will first make available to Open MPI members for distributed testing
  - Then open to community

MTT Reporter:
Relevance & Stability

MTT Reporter

All phases | MPI install | Test build | Test run

Date range:	past 24 hours		Hardware:	all	Show
Org:	all	Show	OS:	all	Show
Local username:	all	Hide	MPI name:	all	Hide
Platform name:	all	Hide	MPI version:	all	Hide

[Reset form] [Start over] Summary | Detail | Performance [Preferences] [Advanced]

Current time (GMT): 2008-05-14 13:10:09
Date range (GMT): 2008-05-13 13:10:09 - 2008-05-14 13:10:09
Phase(s): MPI install, Test build, and Test run
Number of rows: 10

Absolute date range: Create permalink
Relative date range: Create permalink

#	▲Org▼	▲Hardware▼	▲OS▼	MPI install		Test build		Test run				
				▲Pass▼	▲Fail▼	▲Pass▼	▲Fail▼	▲Pass▼	▲Fail▼	▲Skip▼	▲Timed▼	▲Perf▼
1	absoft	ia32	Linux	2	0	2	0	48	0	0	0	0
2	absoft	ppc	Darwin	2	0	2	0	24	0	0	0	0
3	absoft	undef	undef	0	0	3	0	78	0	0	0	0
4	cisco	x86_64	Linux	8	5	56	0	84246	725	648	292	118
5	ibm	ia32	Linux	4	0	20	0	742	2	36	4	52
6	iu	ppc64	Linux	8	0	23	0	5014	617	36	375	0
7	iu	x86_64	Linux	15	12	71	0	19704	179	51	10	0
8	mellanox	x86_64	Linux	5	0	30	0	4120	0	0	0	104
9	sun	i86pc	SunOS	2	0	12	0	2426	6	260	0	52
10	sun	sun4u	SunOS	1	1	6	0	1873	18	147	8	36
Totals				**47**	**18**	**225**	**0**	**118275**	**1547**	**1178**	**689**	**362**

Total script execution time: 8 second(s)

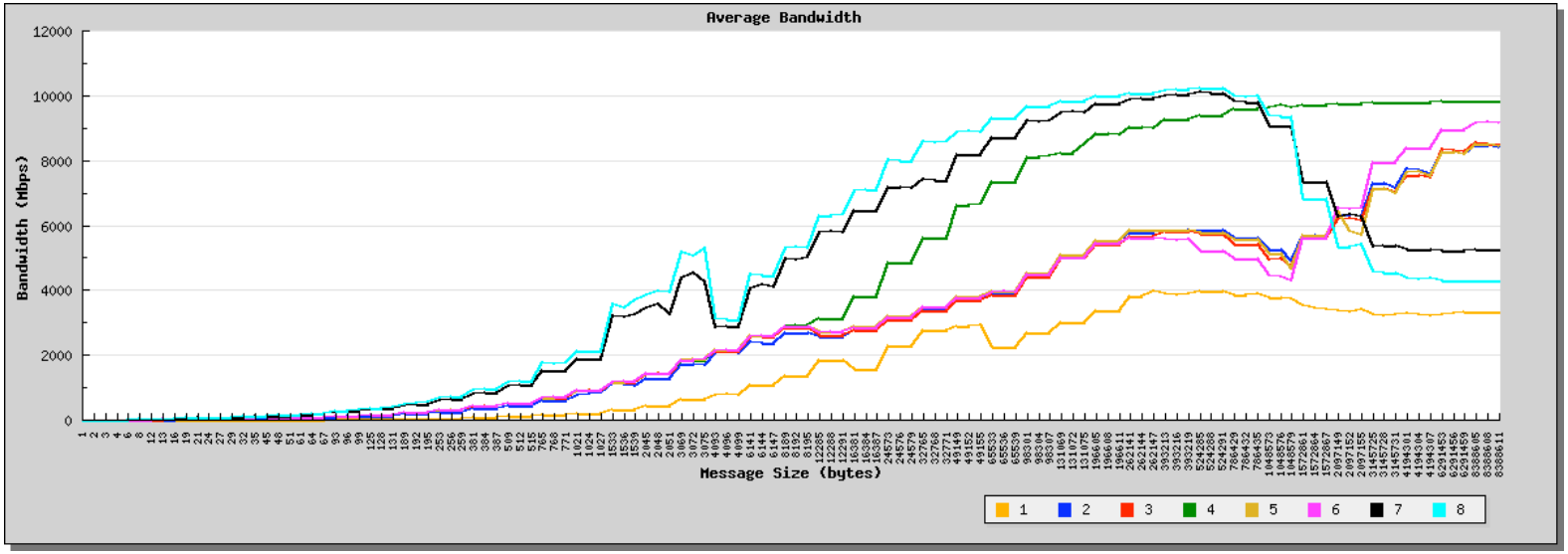
Total SQL execution time: 2 second(s)

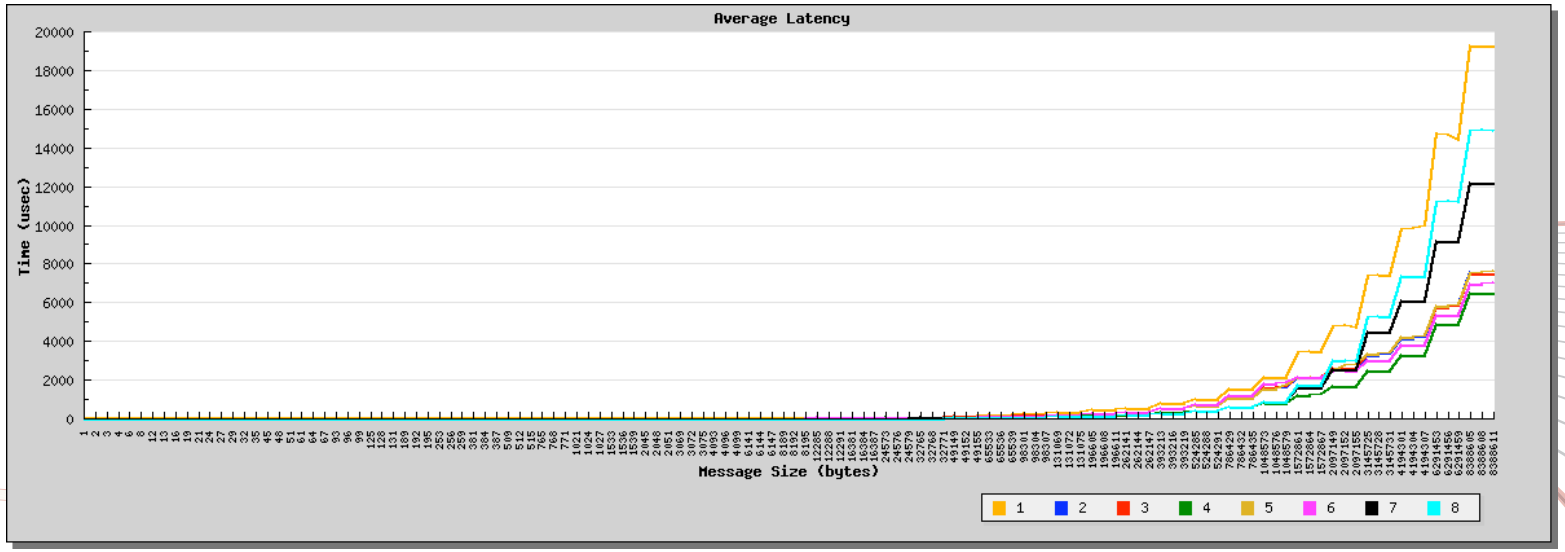
Overall MTT contribution graph (updated nightly): mtt-contrib.pdf

MTT Reporter: Debugging

#	1
Date range	2008-05-14 05:12:05
Org	iu
Platform name	IU_Odin
Hardware	x86_64
OS	Linux
Compiler	gnu
Vpath mode	unknown
Compiler version	3.4.6
Configure arguments	FCFLAGS=-m64 FFLAGS=-m64 CFLAGS=-m64 CXXFLAGS=-m64 --with-openib=/usr/local/ofed --with-openib-libdir=/usr/local/ofed/lib64 --with-wrapper-cflags=-m64 --with-wrapper-cxxflags=-m64 --with-wrapper-fflags=-m64 --with-wrapper-fcflags=-m64 --without-memory-manager --disable-debug --enable-binaries --with-devel-headers --disable-mpi-io --disable-mpi-f90 --disable-ipv6 --with-ft=cr --with-blcr=/san/blcr-0.6.5 --enable-mpi-threads --enable-ft-thread
Description	
Exit value	2
Signal	-1
Duration	00:06:33
Client serial	45755
Result message	Failed to build: make -j 8 all
Stdout	--- "make all result_stdout/result_stderr --- mv -f $depbase.Tpo $depbase.Plo depbase=`echo coll_sm_reduce.lo \| sed 's\|[^/]*$|.deps/&|;s|\.lo$||'`;\ /bin/sh ../../../../libtool --tag=CC --mode=compile gcc -DHAVE_CONFIG_H -I. -I../../../../opal/include -I../../../../orte/include -I../../../../ompi/include -I../../../../opal/mca/paffinity/linux/plpa/src/libplpa -I../../../.. -O3 -DNDEBUG -m64 -finline-functions -fno-strict-aliasing -pthread -fvisibility=hidden -MT coll_sm_reduce.lo -MD -MP -MF $depbase.Tpo -c -o coll_sm_reduce.lo coll_sm_reduce.c &&\ mv -f $depbase.Tpo $depbase.Plo libtool: compile: gcc -DHAVE_CONFIG_H -I. -I../../../../opal/include -I../../../../orte/include -I../../../../ompi/include -I../../../../opal/mca/paffinity/linux/plpa/src/libplpa -I../../../.. -O3 -DNDEBUG -m64 -finline-functions -fno-strict-aliasing -pthread -fvisibility=hidden -MT coll_sm_barrier.lo -MD -MP -MF .deps/coll_sm_barrier.Tpo -c coll_sm_barrier.c -fPIC -DPIC -o .libs/coll_sm_barrier.o libtool: compile: gcc -DHAVE_CONFIG_H -I. -I../../../../opal/include -I../../../../orte/include -I../../../../ompi/include -I../../../../opal/mca/paffinity/linux/plpa/src/libplpa -I../../../.. -O3 -DNDEBUG -m64 -finline-functions -fno-strict-aliasing -pthread -fvisibility=hidden -MT coll_sm_reduce.lo -MD -MP -MF .deps/coll_sm_reduce.Tpo -c coll_sm_reduce.c -fPIC -DPIC -o .libs/coll_sm_reduce.o libtool: compile: gcc -DHAVE_CONFIG_H -I. -I../../../../opal/include -I../../../../orte/include

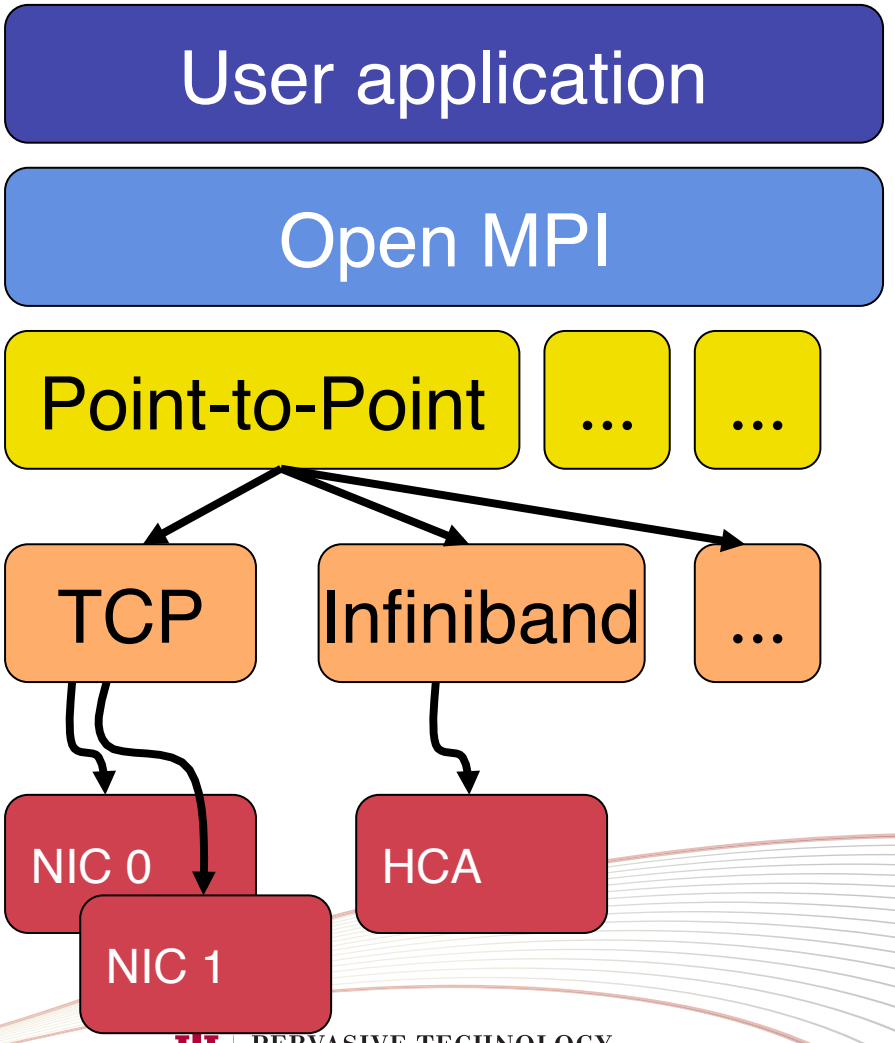
MTT Reporter:
Performance





Modular Component Arch. (MCA)

- **Framework**:
  - API targeted to a specific task
- **Component**:
  - An implementation of a framework's API
- **Module**:
  - An instance of a component



Why Components?

- Core set included in Open MPI distribution
- 3rd parties can develop / distribute
  - Open MPI development to the community
  - As source or binary (open vs. closed source)
- Can be added to existing Open MPI install
  - Reduce the need for multiple MPI installations
  - Can even be added on a per-user basis
- Run-time decisions (vs. compile-time)

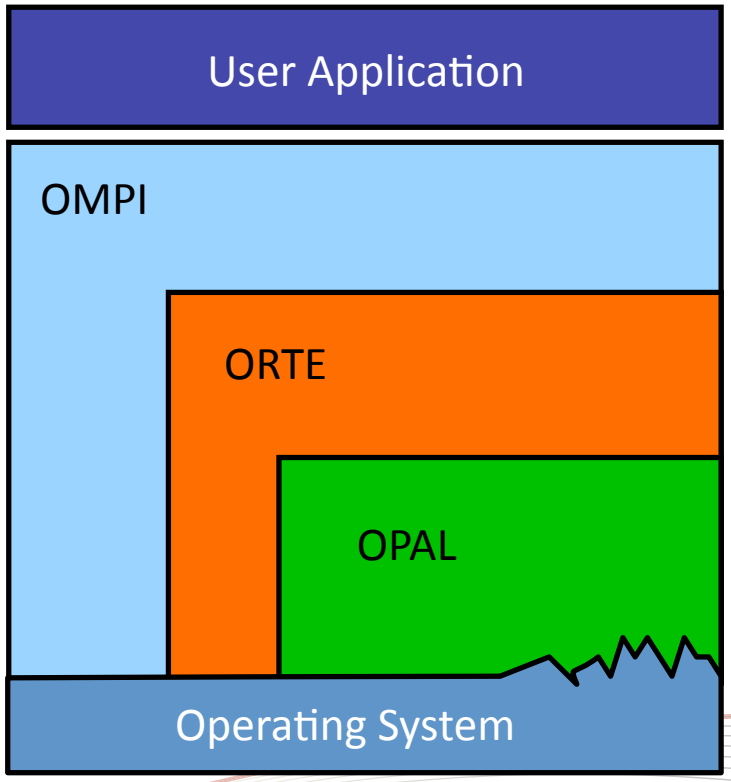
Why Components?

- Better software engineering
  - Enforce strict abstraction barriers
- Small, discrete chunks of code
  - Good for learning / new developers
  - Easier to maintain and extend
- Separate user apps from back-end libraries
  - E.g., user MPI apps not compiled against libibverbs.so / libgm.so / libpbs.a

Layers of Open MPI

Three distinct layers:

- **OMPI**:
  Open Message Passing Interface
- **ORTE**:
  Open Runtime Environment
- **OPAL**:
  Open Portable Access Layer



Code Layout

- <section>/mca/<framework>/<component>
  - Section = opal, orte, ompi
  - Framework = framework name, or “base”
  - Component = component name, or “base”
- Examples
  - ompi/mca/btl/tcp
  - ompi/mca/btl/openib

OPAL Framework Types

- opal/mca/*
  - backtrace — Backtrace accessors
  - carto — Host structure information
  - crs — Checkpoint/restart service
  - installdirs — Installation path accessors
  - maffinity — Memory affinity
  - memchecker — Memory checker
  - memcpy — Optimized memcpy implementations
  - memory — Memory hooks
  - paffinity — Processor affinity
  - pstat — Process statistics (ompi-top)
  - timer — High-resolution timers

ORTE Framework Types

- orte/mca/*
  - errmgr — Error manager
  - ess — Environment specific service
  - fddp — Fault predication
  - filem — File management
  - iof — I/O forwarding
  - notifier — Notification
  - odls — Daemon local launch subsystem
  - oob, rml, grpcomm, routed — Communication
  - plm — Process launch / control
  - ras, rmaps — Resource allocation and mapping
  - rmcast — Reliable multicast
  - sensor — Hardware sensor interface
  - snapc — Snapshot Coordinator

OMPI Framework Types

- ompi/mca/*
  - allocator — Memory allocation
  - coll — Collective operations
  - crcp — Checkpoint/restart coordination protocol
  - dpm — Dynamic process management
  - io — Parallel I/O
  - mpool — Memory pooling
  - op — MPI_Op back-end operations
  - osc — One-sided operations
  - pml,mtl,bml,btl — Point-to-point
  - pubsub — Publish/Subscribe
  - rcache — Registration cache
  - topo — Topology management

Tunable Parameters

- Philosophy: do not use constants
  - Use run-time parameters instead
- Referred to as “MCA parameters”
- Make everything a run-time decision
  - Give every param a “sensible” default
- Parameters usually indicate:
  - Values (e.g., short/long message size)
  - Behavior (e.g., selection of algorithm)

MCA Parameters

- Run-time tunable values
  - Per layer
  - Per framework
  - Per component ("plugin")
- Change behaviors of code at run-time
  - Does *not* require recompiling / re-linking
- Simple example
  - Choose which network to use for MPI communications

Intrinsic MCA Params

- Each framework name is an MCA param
  - Specifies which components to open
- MCA base automatically registers it
  - Value is a comma-delimited list of component names
  - Default value is often empty (meaning “all”)
- Inclusionary or exclusionary behavior
  - btl=tcp,self,sm
  - btl=^tcp

MCA Parameter Lookup Order

1. mpirun command line

```
mpirun --mca <name> <value>
```

2. Environment variable

```
export OMPI_MCA_<name>=<value>
```

3. File
   - $HOME/.openmpi/mca-params.conf
   - $prefix/etc/openmpi-mca-params.conf

   (these locations are themselves tunable)

4. Default value

ompi_info Command

- Tells everything about OMPI installation
  - Finds all components and all params
  - Great for debugging
- Can look up specific component
  - ompi_info --param <framework> <component>
  - Shows params and current values
  - Can also use keyword “all”
- “--parsable” option

Open MPI Repository

- https://svn.open-mpi.org/svn/ompi
  - The Open MPI source code
  - Directory structure:
    - /trunk development head
    - /branches/v1.2 Open MPI 1.2 release branch
    - /branches/v1.3 Open MPI 1.3 release branch
    - /tags/v1.3 Open MPI 1.3
    - /tags/v1.3.1 Open MPI 1.3.1
    - /tmp/…. Volatile development area

User Documentation

- Current main source: web FAQ
  - Easily extensible PHP code
  - Every time we see a question twice, put it on the FAQ
  - Google-able
- Heavily use of mailing lists
  - Web-archives, so also Google-able

MPI Extended Interfaces

- Developers can expose not-yet standard MPI interfaces to users that request them
  - Supports research and standardization
- User must opt-in in order to use the new interfaces.

```
$ ./configure --enable-mpi-ext=magic,nbc
```

See the following site for more details:
https://svn.open-mpi.org/trac/ompi/wiki/MPIExtensions

How to get involved

- Where to get the code:

  svn co http://svn.open-mpi.org/svn/ompi/trunk

- Send questions & patches to ompi-devel

  http://www.open-mpi.org/community/lists/ompi.php

- If you wish to become a contributing member

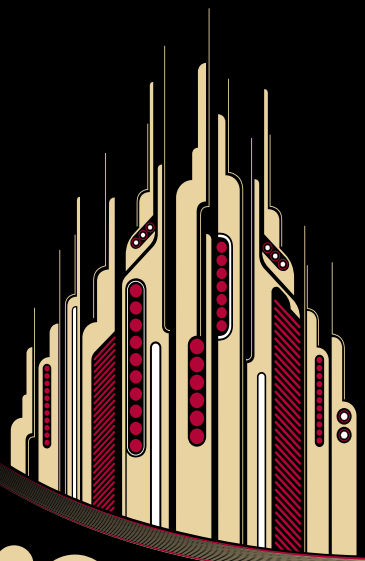
  http://www.open-mpi.org/faq/?category=contributing

Open MPI Tutorial

Andrew Lumsdaine
Joshua Hursey
Jeffrey M. Squyres
Abhishek Kulkarni

Thurs., Nov. 19, 2009
10:00 a.m. – 12:00 p.m.

http://osl.iu.edu/research/ft



Look to the future of high-performance computing.

http://www.open-mpi.org

http://osl.iu.edu/research/ft

Open MPI Tutorial Overview

- **Introduction** . . . . . *10:00 am*
  Andrew Lumsdaine
- **Open MPI** . . . . . *10:15 am*
  Jeffrey M. Squyres
  - Project Overview (≈*10:15 am*)
  - Building & Installing (≈10:30 am)
  - Application MPI Tuning (≈10:45 am)
- **Fault Tolerance** . . . . . *11:00 am*
  Joshua Hursey & Abhishek Kulkarni
  - CIFTS Fault Tolerance Backplane (FTB) (≈11:00 am)
  - Transparent Checkpoint/Restart (≈11:15 am)
  - Process Migration (≈11:30 am)
  - Parallel Debugging (≈11:40 am)
- **Hacking Open MPI** . . . . . *11:45 am*
  Joshua Hursey
  - Modular Component Architecture
  - Developing for Open MPI