ReMPI

MPI Record-and-Replay Tool for Debugging/Testing Non-deterministic MPI Applications

ECP 2nd annual meeting
February 5th

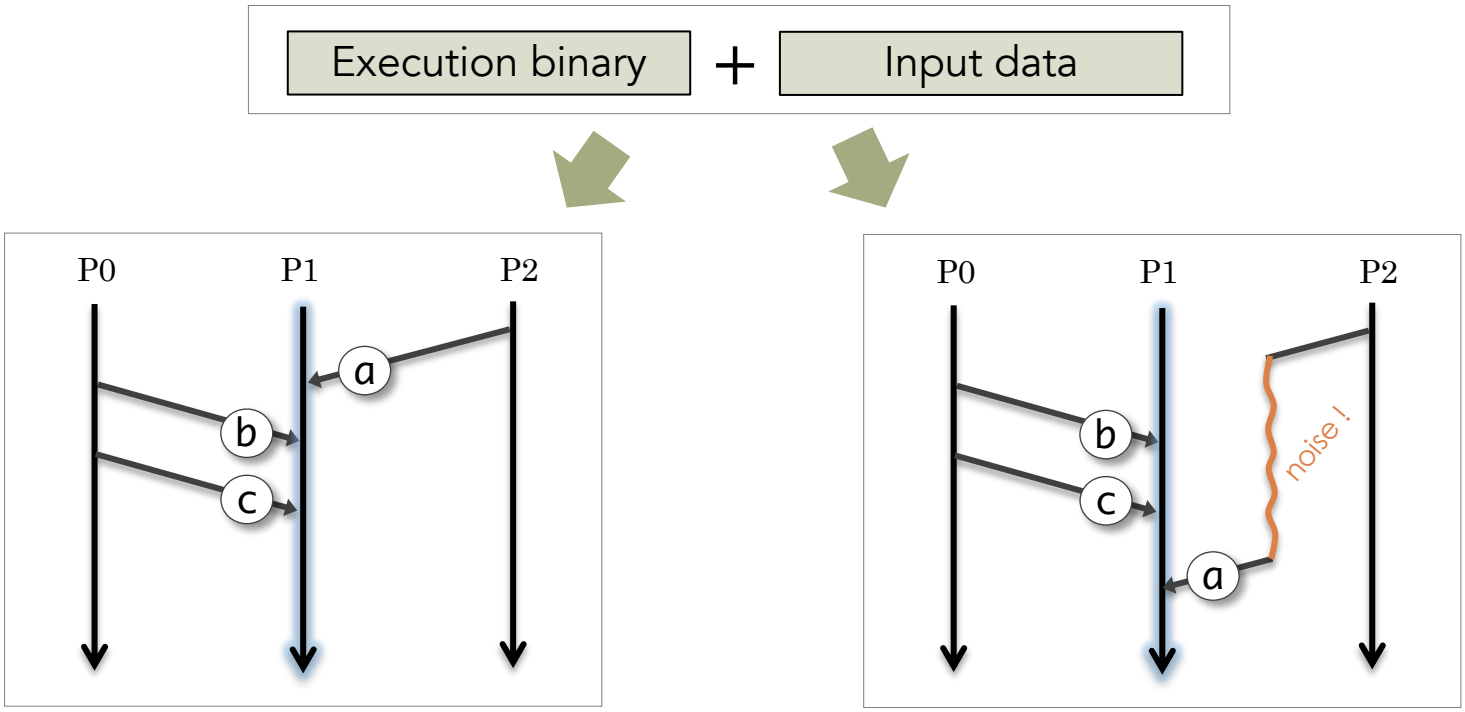
Kento Sato

Lawrence Livermore National Laboratory

LLNL-PRES-745265
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

What is MPI non-determinism ?

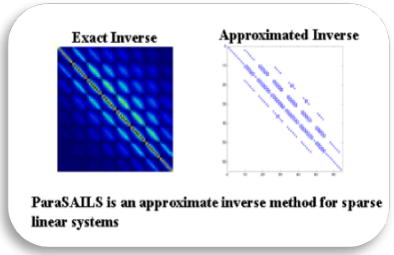
- Message receive orders change across executions
  - Unpredictable system noise (e.g. network, system daemon & OS jitter)
- Non-deterministic bug



If a bug manifests through a particular message receive order,
It's hard to reproduce the bug, thereby, hard to debug it

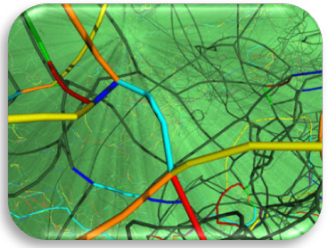
Non-deterministic bugs cost substantial amounts of time and efforts in MPI applications

Diablo/Hypre 2.10.1



- The bug manifested in particular clusters
- It hung only once every 30 runs after a few hours
- The scientists spent 2 months in the period of 18 months, and then gave up on debugging it

ParaDis



- The bug intermittently crashed the application at 100 to 200 iteration
- The scientists gave up debugging by themselves

and more ...

How MPI introduces non-determinism ?

- It's typically due to communication with MPI_ANY_SOURCE
- In non-deterministic applications, each MPI rank doesn't know which other MPI rank will send message and when

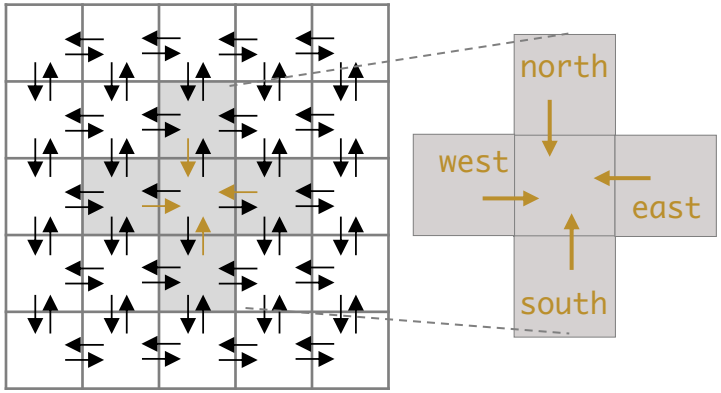
Non-deterministic code w/ MPI_ANY_SOURCE

```
MPI_Irecv(…, MPI_ANY_SOURCE, …);
while(1) {
  MPI_Test(flag);
  if (flag) {
    <computation>
    MPI_Irecv(…, MPI_ANY_SOURCE, …);
  }
}
```

CORAL benchmark: MCB (Monte carlo benchmark)

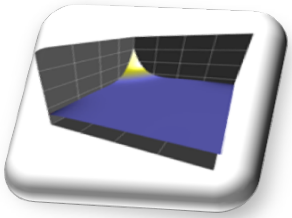
- Use of MPI_ANY_SOURCE is not only source of non-determinism
  - MPI_Waitany/Waitsome/Testany/Testsome also introduce non-determinism

Example: Communications with neighbors



Non-deterministic code w/o MPI_ANY_SOURCE

```
MPI_Irecv(…, north_rank, …, reqs[0]);
MPI_Irecv(…, south_rank, …, reqs[1]);
MPI_Irecv(…, west_rank , …, reqs[2]);
MPI_Irecv(…, east_rank , …, reqs[3]);
while(1) {
  MPI_Testsome(…, &reqs, &count, …, &status);
  if (count>0) {
    …
    for(…) MPI_Irecv(…, status[i].MPI_SOURCE, …);
    …
  }
}
```



MCB: Monte Carlo Benchmark

ReMPI deterministically reproduce order of message receives

https://github.com/PRUNERS/ReMPI

- ReMPI is an MPI record-and-replay tool
  - Record an order of MPI message receives
  - Replay the exactly same order of MPI message receives
- Even if a bug manifests in a particular order of message receives, ReMPI can consistently reproduce the target bug
- ReMPI is implemented as a PMPI wrapper
  - ReMPI can be used
    - On any MPI implementations
    - without recompiling your applications
- ReMPI can run with existing debugging tools
  - STAT,
  - Totalview, DDT

ReMPI replays matching/probing functions

- Message receive function
  - MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)
- Matching functions (Red variables are replayed)
  - MPI_Wait(MPI_Request *request, MPI_Status *status)
  - MPI_Waitany(int count, MPI_Request array_of_requests[], int *index, MPI_Status *status)
  - MPI_Waitsome(int incount, MPI_Request array_of_requests[], int *outcount, int array_of_indices[], MPI_Status array_of_statuses[])
  - MPI_Waitall(int count, MPI_Request array_of_requests[], MPI_Status *array_of_statuses)
  - MPI_Test(MPI_Request *request, int *flag, MPI_Status *status)
  - MPI_Testany(int count, MPI_Request array_of_requests[], int *index, int *flag, MPI_Status *status)
  - MPI_Testsome(int incount, MPI_Request array_of_requests[], int *outcount, int array_of_indices[], MPI_Status array_of_statuses[])
  - MPI_Testall(int count, MPI_Request array_of_requests[], int *flag, MPI_Status array_of_statuses[])
- Probing functins (Red variables are replayed)
  - MPI_Probe(int source, int tag, MPI_Comm comm, MPI_Status *status)
  - MPI_Iprobe(int source, int tag, MPI_Comm comm, int *flag, MPI_Status *status)

ReMPI provides several options for installation

https://github.com/PRUNERS/ReMPI

- Spack

```
$ git clone https://github.com/LLNL/spack
$ ./spack/bin/spack install rempi
```

- Tarball
  - https://github.com/PRUNERS/ReMPI -> [releases]

```
$ tar zxvf ./rempi_xxxxx.tar.bz
$ cd<rempi directory>
$ ./configure --prefix=<path to installation directory>
$ make
$ make install
```

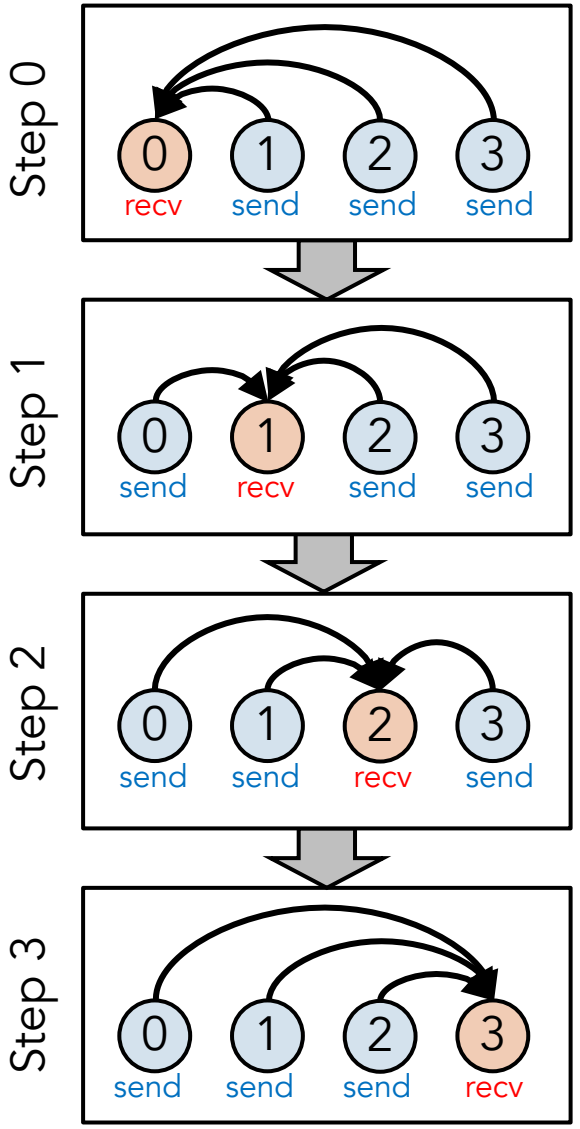
- Git repository

```
$ git clone git@github.com:PRUNERS/ReMPI.git
$ cd ReMPI
$ ./autogen.sh
$ ./configure --prefix=<path to installation directory>
$ make
$ make install
```

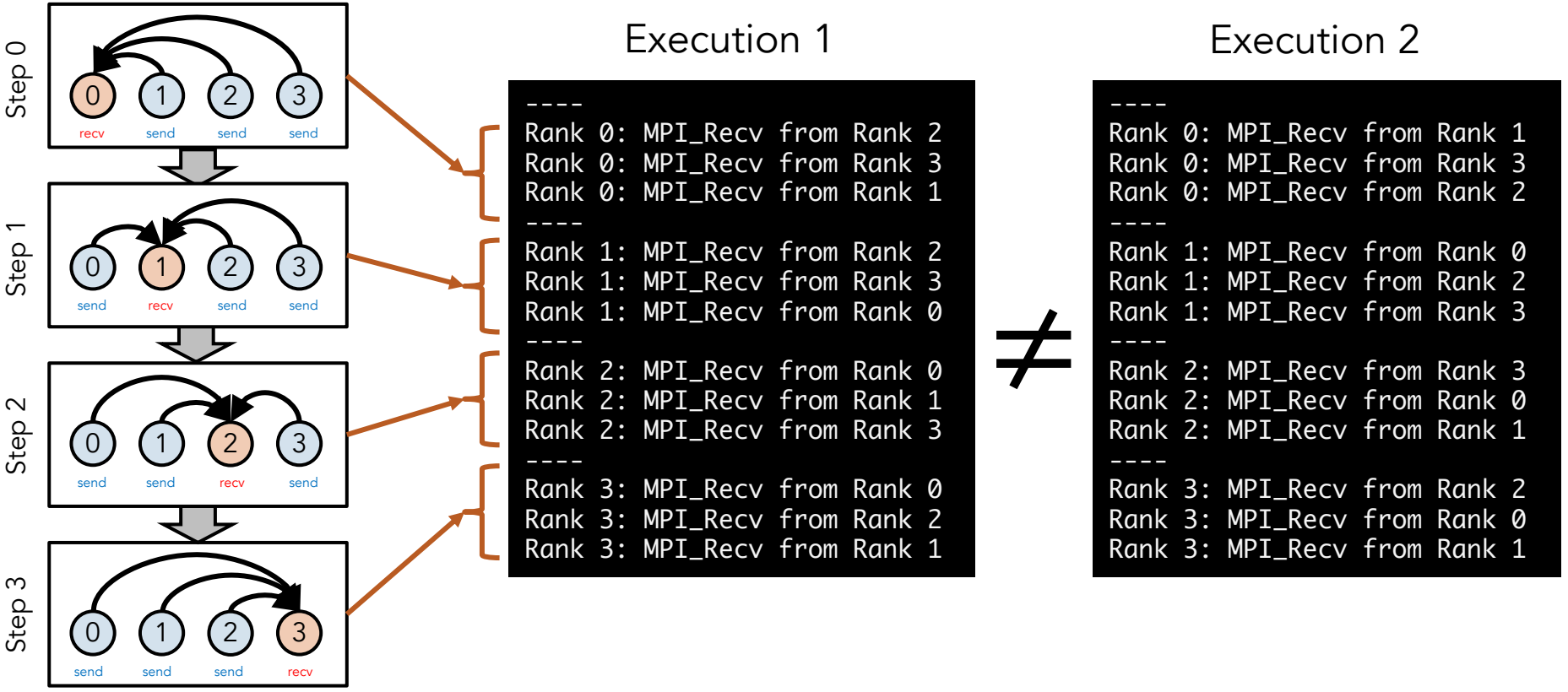
Example code

example.c

```
MPI_Comm_rank(MPI_COMM_WORLD,&my_rank);
MPI_Comm_size(MPI_COMM_WORLD,&size);
for(int dest = 0; dest<size; dest++) {
   if(my_rank == dest) {
     for(i = 0; i<size-1; i++) {
       MPI_Recv(…, MPI_ANY_SOURCE, …);
     }
   } else {
     MPI_Send(…, dest,…);
   }
   MPI_Barrier(MPI_COMM_WORLD);
}
```



Example code (cont'd)



ReMPI record-and-replay

- Record

```
$ rempi_record srun -n 4 example
```

OR

```
$ export REMPI_MODE=record
$ export LD_PRELOAD=/path/to/librempi.so
$ srun -n 4 example
```

- Replay

```
$ rempi_replay srun -n 4 example
```

OR

```
$ export REMPI_MODE=replay
$ export LD_PRELOAD=/path/to/librempi.so
$ srun -n 4 example
```

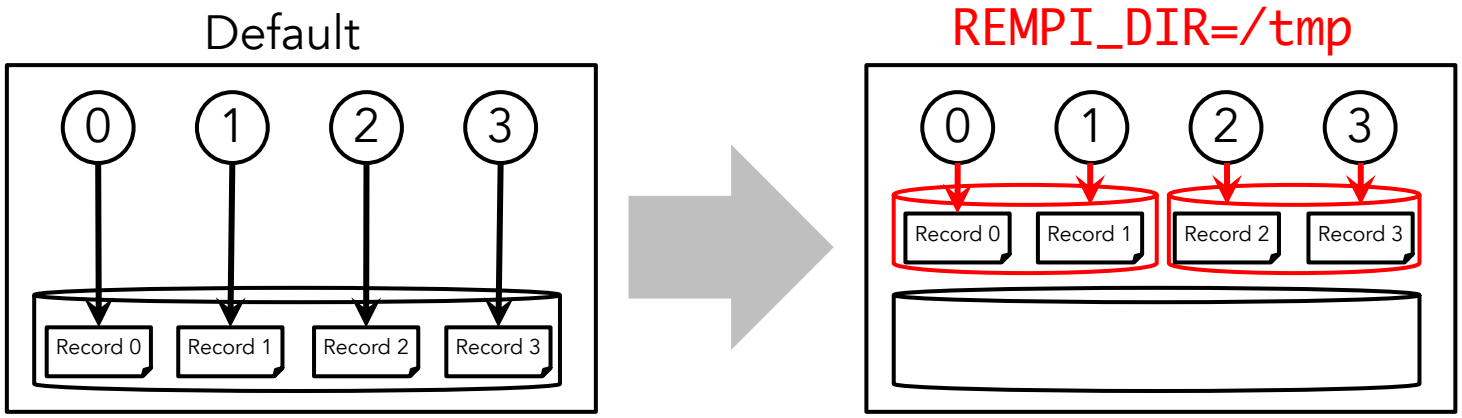
REMPI_DIR: Specifying record directory

- By default, ReMPI stores record files to current working directory
  - You can record file directory via "REMPI_DIR"
- Example
  - Record

```
$ rempi_record REMPI_DIR=/tmp srun -n 4 example
```

  - Replay

```
$ rempi_replay REMPI_DIR=/tmp srun -n 4 example
```



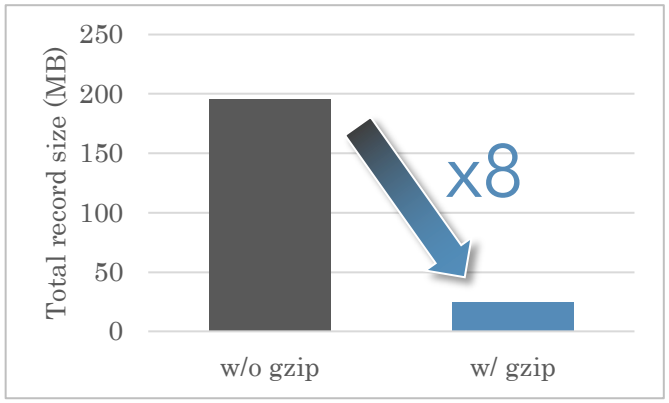
REMPI_GZIP: Compressing record

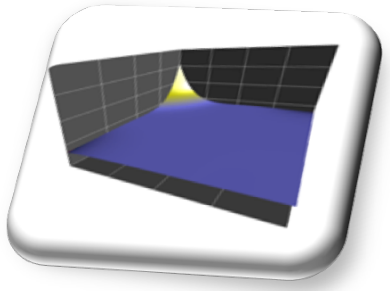
- ReMPI apply gzip the record data to reduce record size
- Example
  - Record

```
$ rempi_record REMPI_DIR=/tmp REMPI_GZIP=1 srun –n 4 example
```

  - Replay

```
$ rempi_replay REMPI_DIR=/tmp REMPI_GZIP=1 srun –n 4 example
```





MCB: Monte Carlo Benchmark

Total record size in MCB at 3,072 procs (Runtime: 12.3 sec)

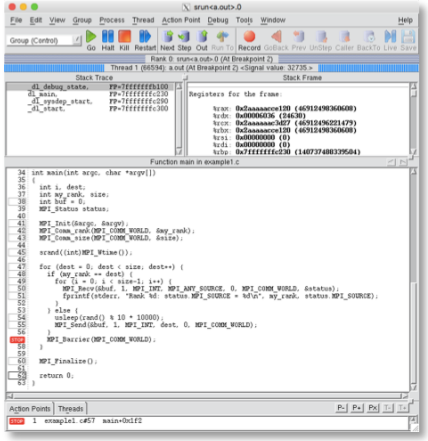
ReMPI replay under Totalview control

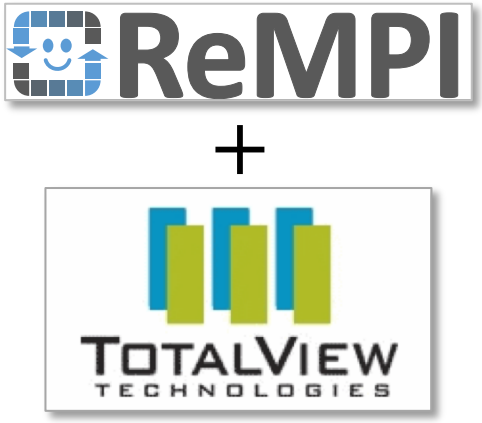
- ReMPI can also work with existing parallel debuggers
  - E.g.) Totalview
- Example
  - Record

```
$ rempi_record srun –n 4 example
```

  - Replay

```
$ rempi_replay totalview -args srun –n 4 example
```





Q&A

ReMPI

PRUNERS ReMPI 

OR https://github.com/PRUNERS/ReMPI


Lawrence Livermore
National Laboratory