MPI on HPX: Status Update
Overview

• Work focused on startup procedure
• Using VolpexMPI instead of Open MPI initially
  - Smaller code base makes debugging and transition smoother
  - Will return to Open MPI later in the process
VolpexMPI

• Experimental MPI library
  o Supports replicated execution of MPI processes
  o Receiver initiated communication model
  o Distributed sender logging
  o Sophisticated algorithms for choosing most appropriate processes to communicate with
  o Tools for dynamically adding processes and monitoring state of job/node/process available

• Prototype implementation supports ~40 MPI functions
• Support for ssh, BOINC and CONDOR execution environments
• TCP based messaging
NAS Parallel Benchmarks

- Normalized execution times of VolpexMPI on a dedicated cluster over Gigabit Ethernet
- Open MPI v1.4.1 reference times are 100
Default Startup Procedure

**mcfarun / mpirun**
- what resources are available
- how many processes to start
- mapping of processes to resources
- launches startup_d daemon
- wait for job to finish

**startup_d**
- sets environment variables required for the MPI jobs
- starts the requested number of MPI processes on that node
- waits for children to finish

---

**`MPI_Init()`**
- reads environment variables set by `startup_d`
- connects back to `mcfarun`
- optionally: requests a unique process ID
- receives complete list/contact information of all processes in the MPI job
- generates internal structures required e.g. for `MPI_COMM_WORLD` etc.
**Startup Procedure with -hpx**

**mcfarun / mpirun**
- what resources are available
- how many processes to start
- mapping of processes to resources
- launches startup_d daemon 
- wait for job to finish

**startup_d**
- writes environment variables to file
- forks
- exec ssh shark01 ...
- starts the requested number of MPI processes on that node
- waits for children to finish

**MPI_Init()**
- reads environment variables from file
- connects back to mcfarun
- optionally: requests a unique process ID
- receives complete list/contact information of all processes in the MPI job
- generates internal structures required e.g. for MPI_COMM_WORLD etc.
Running an MPI program in VolpexMPI:

```bash
hadi@shark:~> mcfarun -np 4 -hostfile hosts <mpi_hello_world>
```

Running an HPX-MPI program in VolpexMPI:

```bash
hadi@shark:~> mcfarun -np 4 -hostfile hosts -hpx <hpx_mpi_hello_world>
```
hadi@shark:~> mcfarun -np 4 -hostfile hosts -hpx <hpx_mpi_hello_world>

```
ssh shark hpx_mpi_hello_world -x shark:7912 -l 4 --hpx:console
ssh shark01 hpx_mpi_hello_world -x shark01:7912 -a shark:7912 --hpx:worker
ssh shark02 hpx_mpi_hello_world -x shark02:7912 -a shark:7912 --hpx:worker
ssh shark03 hpx_mpi_hello_world -x shark03:7912 -a shark:7912 --hpx:worker
```

hosts

- shark01
- shark02
- shark03
- shark04
Example hello_world output

```
hadi@shark:~> mcfarun -np 4 -hostfile hosts -hpx <hpx_mpi_hello_world>
```

<table>
<thead>
<tr>
<th>Process ID</th>
<th>Volpex_ID</th>
<th>Host Name</th>
<th>Port Num</th>
<th>Job ID</th>
<th>Socket</th>
<th>Status</th>
<th>Fullrank</th>
<th>Executable</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>192.168.1.173</td>
<td>45001</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>0,A</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>192.168.1.171</td>
<td>45002</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>1,A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>192.168.1.172</td>
<td>45003</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>2,A</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>192.168.1.170</td>
<td>45004</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>3,A</td>
<td></td>
</tr>
</tbody>
</table>

Hello world from process 3 of 4
hello world from OS-thread 0 on locality 0
hello world from OS-thread 0 on locality 2
hello world from OS-thread 0 on locality 3
hello world from OS-thread 0 on locality 1
Hello world from process 0 of 4
Hello world from process 2 of 4
Hello world from process 1 of 4
Next Steps

• Further integration by using HPX as the runtime environment
  • Global IDs of HPX instead of VolpexMPI IDs
  • Replacing VolpexMPI process management by HPX functionality
  • Communication through parcels
• Trying to move the HPX-C instead of C++

• Looking at utilizing XPI routine
  - Status of implementation?
  - Notion of processes in HPX a bit unclear
Influence of redundancy level

- Performance impact of executing one (x1), two (x2) and (x3) replicas of each process
- Normalized to the single redundancy VolpexMPI execution times
Influence of process failures

- Double redundancy
- Failing processes from both teams
- Normalized to the double redundancy execution times

8 processes

16 processes
Target Selection Algorithms (II)

- Double redundancy tests on a heterogeneous configuration
  - fast nodes: Gigabit Ethernet, 2.2 GHz
  - slow nodes: Fast Ethernet, 1.0 GHz
- Initially, both teams contain processes on fast and slow nodes
- Each MPI rank has one fast and one slow process
- Normalized towards double redundancy numbers on GE

---

8 processes

---

16 processes