Running MPI-based HPC on Unstable Processors

Edson Tavares de Camargo **Elias P. Duarte Jr.**

Federal University of Parana (UFPR) Curitiba, BRAZIL

{etcamargo, **elias**}@inf.ufpr.br

66th Meeting of the IFIP WG 10.4, **June 2014**,
Amicalola Falls, USA

HPC - High Performance Computing

- Computing-intensive computation
- Programs that take hours to complete execution
- Scientific applications, very-large scale problems
- Run on a large set of processors/cores

HPC - Massively Parallel Computers (MPC)

Tianhe-2:currently the #1 computer in the www.top500.org list - developed by China's National University of Defense Technology - 33.86 petaflops per second, 16,000 processors, a total of 3,120,000 cores

MPI-based Fault-Tolerant HPC

- Besides MPC: you can use clusters and grids of networked processors running MPI
- It is well-known that the largest the system size, the smallest the MTBF
- There have been several proposals for FT-MPI, the most recent proposed standard by MPI-Forum is ULFM (User-Level Failure Mitigation)

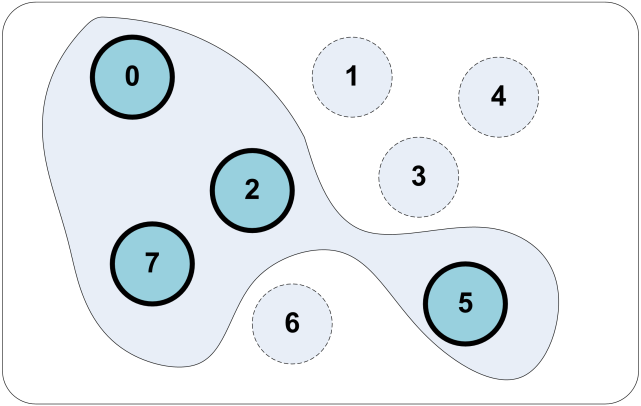
MPI ULFM: User Level Failure Mitigation

- Assumes the fail-stop model
- Assumes reliable communication channels
- Failures are detected as processors communicate
- As processor i determines that processor j is faulty, a consensus primitive can be called to inform the remaining processors
  - processor j is forever removed from the system

Shared Processors Can Present Unstable Behavior

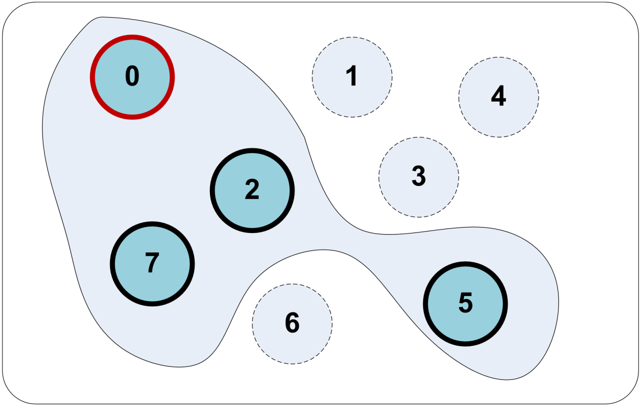
- In real systems based on shared processors the observed behavior of processors vary widely
    - due to a varying load and also network conditions
- In particular: a fault-free processor may fail to send a response within an expected time frame
- This condition can be transient, i.e. the processor later returns to predictable, stable behavior
- Using ULFM as soon as the processor goes through a unstable phase it is eliminated from the system forever

Proposed Solution: Maintaing a Dynamic Core of Stable Processors



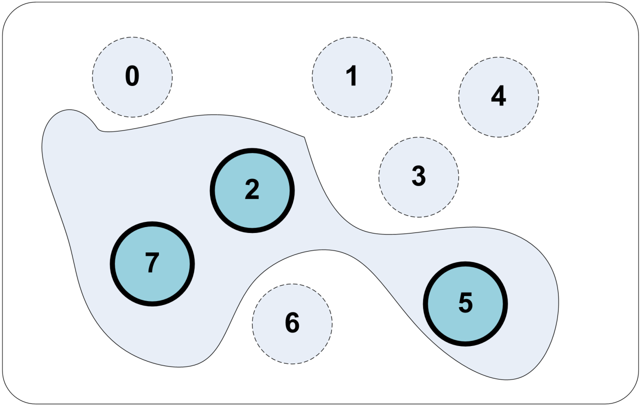
Stable core formed by processes 0, 2, 5 and 7

Proposed Solution: Maintaing a Dynamic Core of Stable Processors



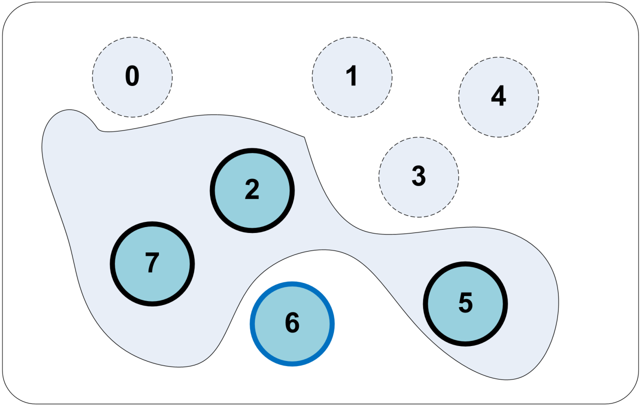
Process 0 becomes unstable

Proposed Solution: Maintaing a Dynamic Core of Stable Processors



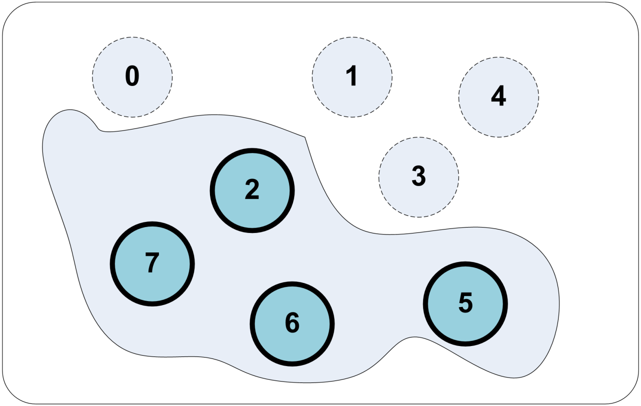
Stable core formed by processes 2, 5 and 7

Proposed Solution: Maintaing a Dynamic Core of Stable Processors



Process 6 considered stable

Proposed Solution: Maintaing a Dynamic Core of Stable Processors



Stable core formed by processes 2, 5, 6 and 7

Maintaining a Dynamic Core of Stable Processors

- We propose a strategy to diagnose system stability
- Monitoring is based on tests - pull-based failure detector
- Problem: if a processor is going through a unstable phase, this may be detected by some (not all) processors
- Test outcomes are thus ambiguous in this case: diagnosis with imperfect tests
- A Dynamic Stable Core of Processors (DSCP) consists of all processors considered to be stable by all stable processors

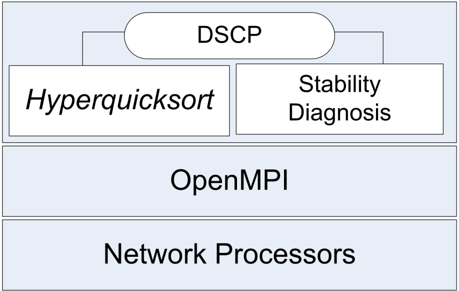
Maintaing a Dynamic Core of Stable Processors

- Processors execute tests on each other, and receive information from those processors tested as stable, if process j is considered to be unstable by any stable processor, it is removed from the DSCP

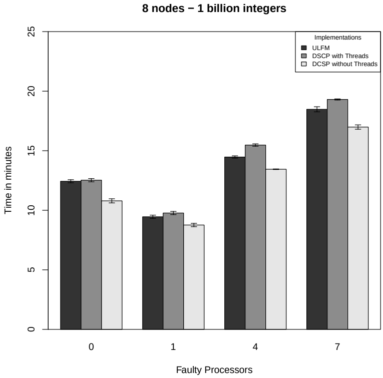
- After processor i tests unstable processor j as stable for ζ consecutive testing rounds, it runs a consensus algorithm within the DSCP to reincorporate j to the core

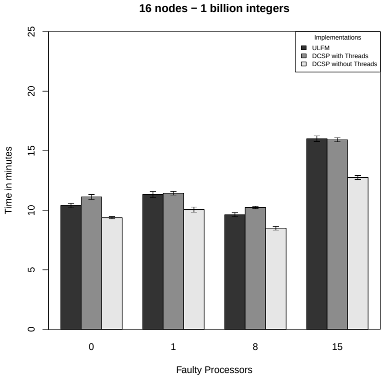
DSCP Implemented

- We implemented DSCP using OpenMPI with an added primitive for consensus
- Paxos was used for consensus: executed only by nodes in the stable core
- Results for HyperQuickSort, a parallel algorithm used for sorting a billion integers



Sample Experimental Results


8 nodes − 1 billion integers


16 nodes − 1 billion integers

Currently Working On...

- Strategies (mainly checkpointing) to optimize the execution flow as the DSCP composition changes
- Extending the model to allow a dynamic system size: new processors are added during run-time
- Extending the model to deal with network partitions: multiple nodes get disconnected/connected at once
- Implementation on PlanetLab: Stable Wormholes
- Implementation in OpenMP (shared memory)
- ...