RESILIENCE EXTENSIONS FOR MPI: ULFM

USER LEVEL FAILURE MITIGATION

User Level Failure Mitigation is a set of new interfaces for MPI that enable Message Passing programs to restore MPI functionality affected by process failures. The MPI implementation is spared the expense of internally taking protective and corrective actions against failures. Instead, it reports operations whose completion is rendered impossible by failures. Using the constructs defined by ULFM, applications and libraries drive the recovery of the MPI state. Consistency issues resulting from failures are addressed according to application's needs and the recovery actions are limited to the necessary MPI communication objects. Therefore, the recovery scheme is more efficient than a generic, automatic recovery technique, and can achieve both goals of enabling applications to resume communication after failures and maintaining extreme communication performance outside recovery periods.

GOALS

FLEXIBILITY

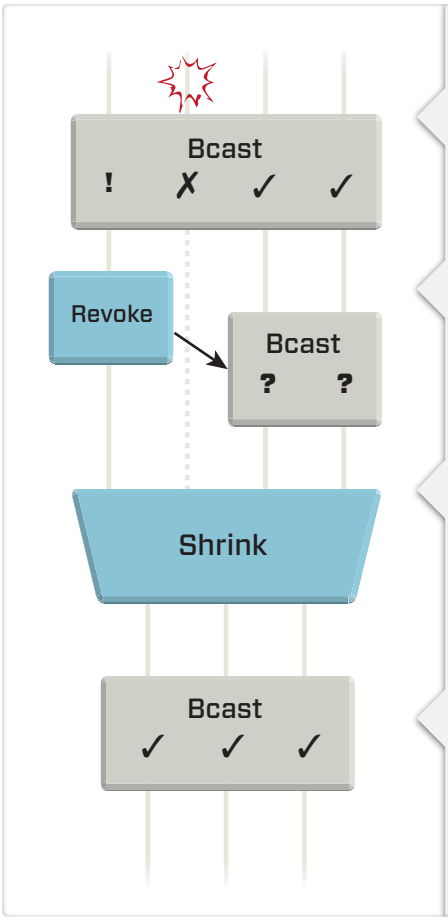
In application-centric recovery, applications pay for the necessary protection only. Varied and complex recovery strategies can be derived.

UNOBTRUSIVENESS

Protective actions are located outside of critical MPI routines. MPI implementations uphold optimized communication and collective algorithms, unmodified.

PRODUCTIVITY

Legacy fragile applications run out of the box, only five new functions are added to repair MPI. Inclusion into the MPI standard is under review, for portability.



Failure Notification

When an MPI communication cannot complete because a participating process has failed, the associated operation is interrupted locally and returns a specific error code.

Error Knowledge Propagation

If a collective recovery scheme is necessary, notified processes call MPI_COMM_REVOKE to interrupt operations at processes proceeding unaware of the failure.

Communication Repair

Applications can either stabilize existing communication objects or replace them with new objects excluding the failed processes. If necessary, replacement processes are spawn with MPI-2 routines.

Continued Execution

After recovery is finished, the application can continue as before. The varied methods of recovery allow highly efficient fault tolerance tailored for each application use case.

INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE UT

SPONSORED BY

E-MAIL ULFM
ulfm@googlegroups.com

FIND OUT MORE AT
www.fault-tolerance.org

RESILIENCE EXTENSIONS FOR MPI: ULFM

USER LEVEL FAILURE MITIGATION

RESILIENCY

Resiliency refers to the ability of the MPI application not only to survive failures, but also to recover into a consistent state from which the execution can be resumed. One of the most strenuous challenges is to ensure that no MPI operation stalls from the consequences of failures, for fault tolerance is impossible if the application cannot regain full control of the execution. In ULFM compliant implementations, an error is returned when a failure prevents a communication from completing. However, it indicates only the local status of the operation, and does not permit deducing if the associated failure has impacted MPI operations at other ranks. This design choice avoids expensive consensus synchronizations from obtruding into MPI routines, but leaves open the danger of some processes proceeding unaware of the failure. ULFM therefore defines the new construct `MPI_COMM_REVOKE` to let processes which have received an error resolve such divergence, but only when necessary.

FLEXIBILITY

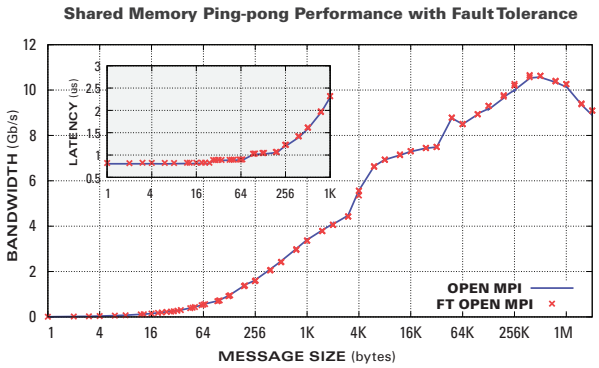
Flexibility in fault response is paramount: not all applications have identical requirements. ULFM puts the recovery actions under the control of the user. Therefore, master-slave applications that can continue their computation despite failures do not pay the cost of any recovery actions, while consistency restoration interfaces are available to applications that need to restore a global context (typical case for applications with collective communications). Aside from applications, ULFM can be used by high level abstractions, such as transactional fault tolerance, uncoordinated checkpoint-restart and programming languages, to support advanced fault tolerance models that are not tied to a particular MPI implementation.

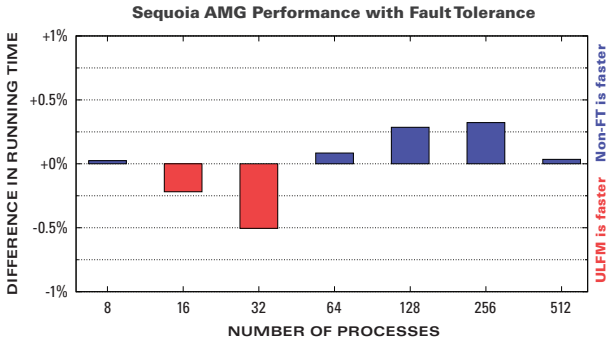
PRODUCTIVITY

Productivity and the ability to handle the large number of legacy codes already deployed in production is another key feature. A fault tolerant API should be easy to understand and use in common scenarios, as complex tools have a steep learning curve and a slow adoption rate by the targeted communities. To this end, the number of newly proposed constructs has been reduced to five (along with nonblocking variants). These five functions provide the minimal set of tools to resume MPI communications after failures. Furthermore, ULFM is backward compatible, and supports non fault tolerant applications without modification to enable incremental migration.

PERFORMANCE

Performance impact outside of recovery periods should be minimal. As illustrated by the Netpipe ping-pong (left) with the ULFM compliant version of Open MPI: even on the most demanding shared memory micro benchmarks, no overhead is observed. The result on Sequoia AMG (right), an unstructured physics mesh application, further emphasizes that failure-free performance is undisturbed with a complex communication pattern. Such outstanding performance is possible because the principles embraced by ULFM limit the number and extent of modifications to the MPI implementation, as failure protection actions within the implementation are minimal and recovery is delayed until requested by the application.





INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE

SPONSORED BY

E-MAIL ULFM
ulfm@googlegroups.com

FIND OUT MORE AT

www.fault-tolerance.org