Cluster computers are made up of a group of computers that are linked together using a dedicated network. Access to the cluster is managed either by some of the nodes, or as shown on the left, managed by a "head node." Parallel programs can be written for clusters that execute on multiple nodes to solve problems faster.

Benefits of clustering
1) Aggregation of computing power.
2) Aggregation of memory.
3) Can be "grown" by adding nodes.

In the message passing parallel programming paradigm, processes executing in parallel use messages to communicate data and synchronize activities. This paradigm is a good fit for the underlying cluster computer architecture because a process can only access memory at the cluster node on which it is executing.

As programmers are required to explicitly detail all inter-process communication, a lot of programmer effort may be required when communication patterns are irregular.

Distributed shared memory (DSM) environments simplify cluster programming by automatically generating the necessary network communications. Working out the needed communication requires the DSM to monitor memory accesses, resulting in overheads that are in addition to the communication. Thus, there exists a tradeoff between ease-of-use and performance.

Yet, not all parallel programming scenarios are difficult to solve using message passing. In such cases, no tradeoff between ease-of-use and performance would exist!

The Best of Both Worlds
Integrating Message Passing into a Distributed Shared Memory Environment
H'sien Jin Wong

Integrating Shared Memory and Message Passing

Objective
The objective of integrating message passing into the distributed shared memory programming environment is so that overheads of the shared memory runtime can be avoided when a message passing solution is easy to implement.

Challenge
Message passing operations result in memory accesses that need to be handled by the DSM to maintain consistency of the shared memory. However, using pre-existing DSM solutions will only result in the same DSM overheads we wish to avoid, defeating the objective of integration.

Solution
Take advantage of the information provided by message passing operations. These are like "bulk memory access" on the provided message buffers. This allows a process to make deductions about memory accesses at other processes unilaterally. E.g. a send operation on buffer X implies that no other process should be writing to X. This presents an opportunity for the DSM to optimize its consistency bookkeeping work.

Results
Bandwidth is the amount of data that can be transferred in a given time. This test compares the bandwidth for transferring a region of memory using: 1) the distributed shared memory, 2) the integrated environment, and 3) pure message passing. Two different memory access patterns are used to create data dependencies, representing Good and Bad cases. The integrated implementation is able to match the performance of pure message passing in the Good case, and is only marginally worst off in the Bad case.

Supervisor: Dr Alistair Rendell