Portable and Scalable Parallel Applications with VCluster

J. Lee, H. Zhang, R. K. Guha
School of Computer Science
University of Central Florida

Acknowledgements:
ARO Grants: DAAA19-01-1-0502
& W911NF04110100
NSF Grant: EIA 0086251

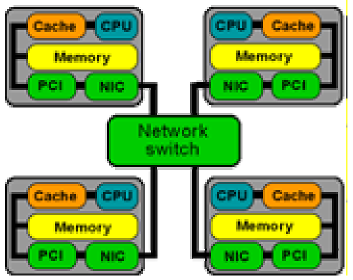
Outline of the presentation

- Introduction to cluster computing
  - Background
- Middleware and VCluster
- Experimentation with applications
  - Communication Overhead
  - Parallel Dirichlet Problem
  - Back Propagation Neural Network
- Conclusion and future work

Cluster Computing

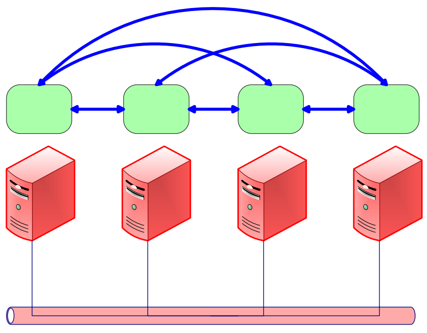
- Become popular with the availability of
  - High performance microprocessors
  - High speed networks
  - Distributed computing tools
- Provide performance comparable to supercomputers with a much lower price

4 node PC/workstation cluster

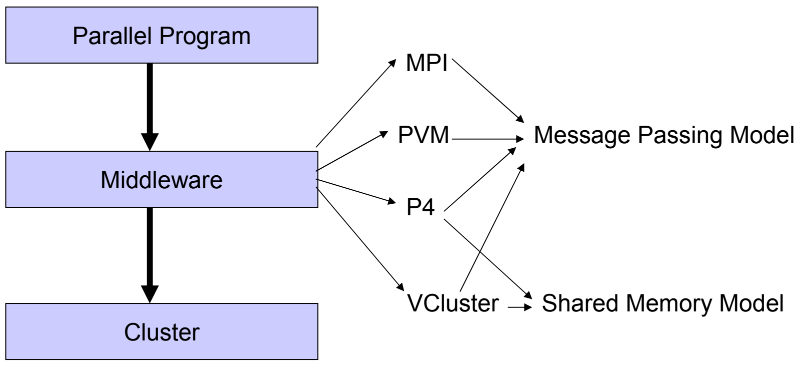


Cluster Computing

- To run a parallel program on a cluster
  - Processes must be created on every machine in the cluster
  - Processes must be able to communicate with each other



Middleware



Middleware

- MPI-Message Passing Interface standard
  - MPI is a library specification for message-passing, proposed as a standard by a broadly based committee of vendors, implementers, and users.
  - MPI 1.0 was finished in June, 1994.
  - High performance is the focus of the MPI design

Middleware

- PVM-Parallel Virtual Machine
  - Is a software package that permits a heterogeneous collection of Unix and/or Windows computers hooked together by a network to be used as a single large parallel computer.
  - Began in the summer of 1989
  - Sacrifice some performance to make heterogeneous machines to work together

UCF

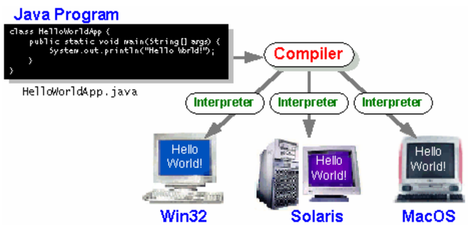
Middleware

- P4
  - p4 is a library of routines designed to express a wide variety of parallel algorithms portably, efficiently and simply
  - Has been used since 1984
  - Support both shared-memory model and message-passing model

UCF

Java Programming language

- Platform Neutral bytecode
  - Java program is compiled into bytecode
  - Bytecode file is carried out by the interpreter (JVM) by translating into machine code
  - Same bytecode file can be carried out on different machines



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

HelloWorldApp.java

Problems in Cluster Computing

- Communication latency
  - Faster networking technologies
    - ATM network
    - Myrinet
      - Particular network protocol
    - Gigabit Ethernet
      - Not fast enough
  - Symmetric Multiprocessor (SMP) machines

Problems in Cluster Computing

- New problems by SMP machines
  - Shared Memory Model
    - Must support shared memory model and message passing model at the same time
  - Multithreading
    - Must support multithreading
    - Must support communication between threads

UCF 11

Problems in Cluster Computing

- Current Systems for SMP machines
  - Shared Memory Model
    - MPI and PVM are focused on message passing model
  - Multithreading
    - None of MPI, PVM or p4 supports multithreading inherently, a separate thread library must be used
    - None of MPI, PVM or p4 supports communication between threads

UCF 12

Problems in Cluster Computing

- Portability and maintainability
  - Heterogeneous machines
    - Big endian and little endian
  - User need to handle these differences or even need to maintain a different copy of parallel program for each type of machine
  - Java is attractive because of its platform neutral byte codes and object oriented paradigm

UCF 13

Problems in Cluster Computing

- Portability and maintainability of current systems
  - MPI, PVM and p4 are primarily for C and Fortran, which have poor portability compared to Java
  - Some Java implementation of MPI and PVM have emerged, but a model dedicated to Java will use Java more efficiently.

UCF 14

VCluster Model

- Motivation

  A solution to the problems above

  - SMP Machines
    - Shared Memory Model
    - Multithreading
    - Thread Communication
  - Portability and maintainability
    - Clusters of heterogeneous machines

UCF 15

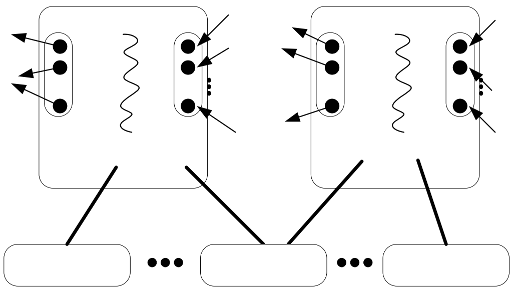
VCluster Model

- Features
  - Supports a cluster consisted of both uni- or multi-processor machines
    - Combines shared memory model and message passing model in a same framework
    - Multithreading and communication between threads
  - Implemented in 100% Java so has good portability and maintainability

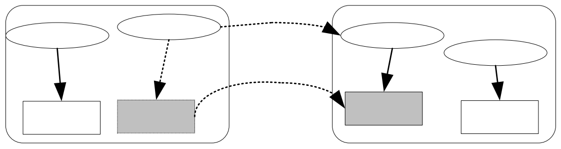
UCF 16

VCluster Model

- Thread as the basic computing unit instead of processes
- Unidirectional channels between threads for communication
- A thread associate itself with a set of states
- Two or more threads can share one state, another way of shared memory communication.
- Programmers write the parallel program by extending VCluster thread, state, channel, and channel set classes.



VCluster Model

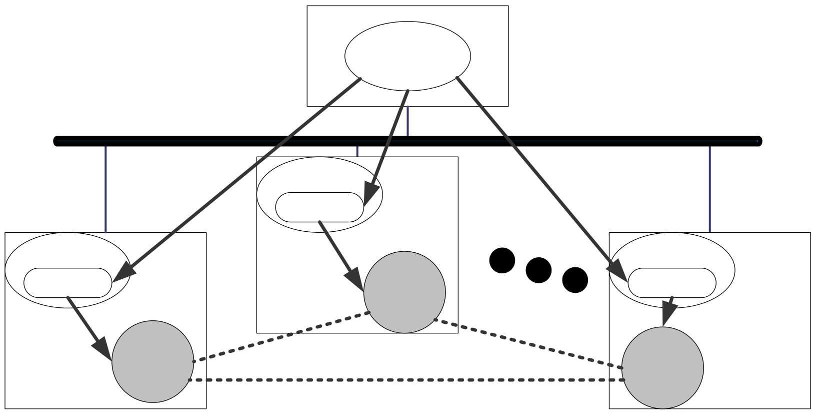


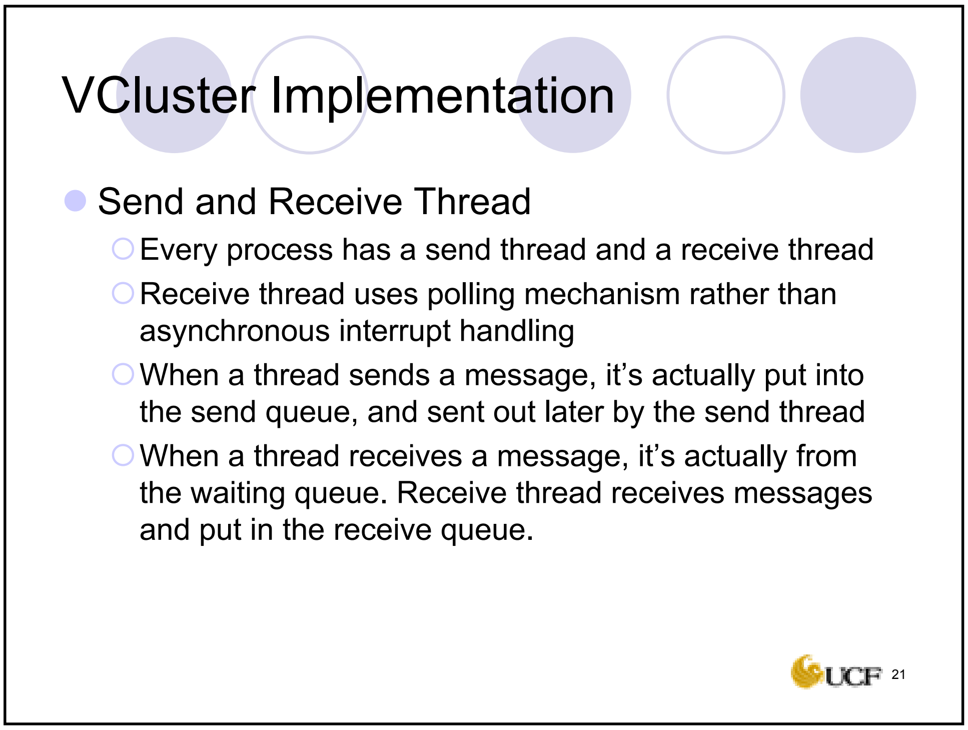
- Virtual Thread Migration
  - State is transferred to another machine and a new virtual thread is created based on the state.
  - A virtual thread is virtual in that it can migrate within the system from machine to machine.

VCluster Implementation

- A daemon is running on each node
  - Process creation and termination
  - Output replay
- Initialization Process
  - Daemon waiting for request
  - A service thread is created for each request
  - Service thread spawn the requested process
  - Processes establish connection with the help of Service thread
  - Service thread relay output to the root process

VCluster Implementation

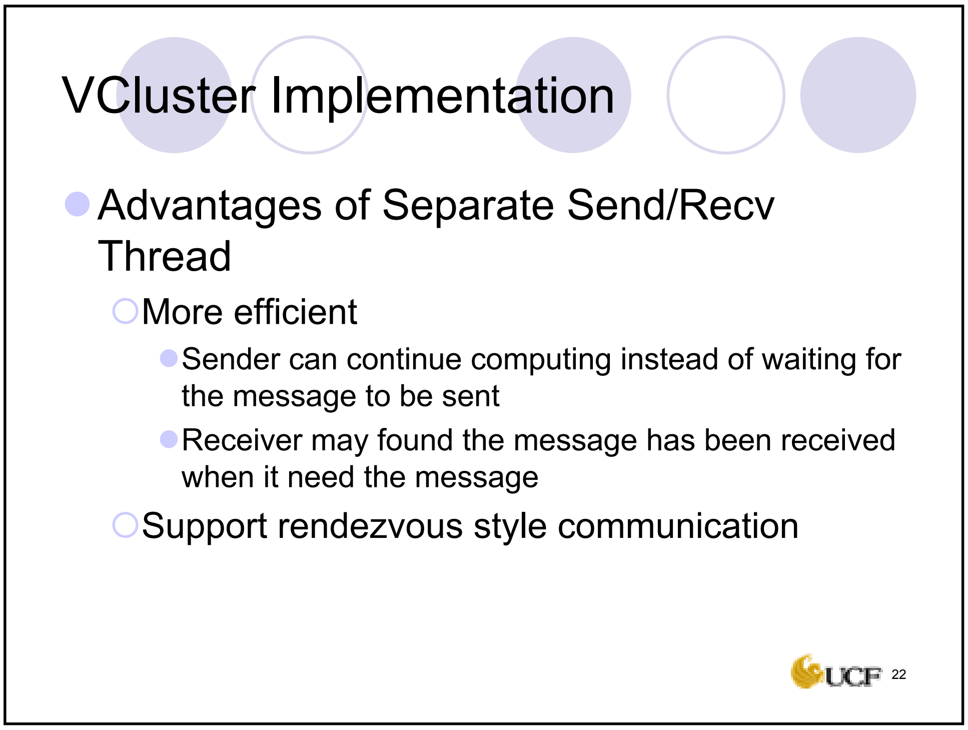




VCluster Implementation

- Send and Receive Thread
  - Every process has a send thread and a receive thread
  - Receive thread uses polling mechanism rather than asynchronous interrupt handling
  - When a thread sends a message, it's actually put into the send queue, and sent out later by the send thread
  - When a thread receives a message, it's actually from the waiting queue. Receive thread receives messages and put in the receive queue.

UCF 21

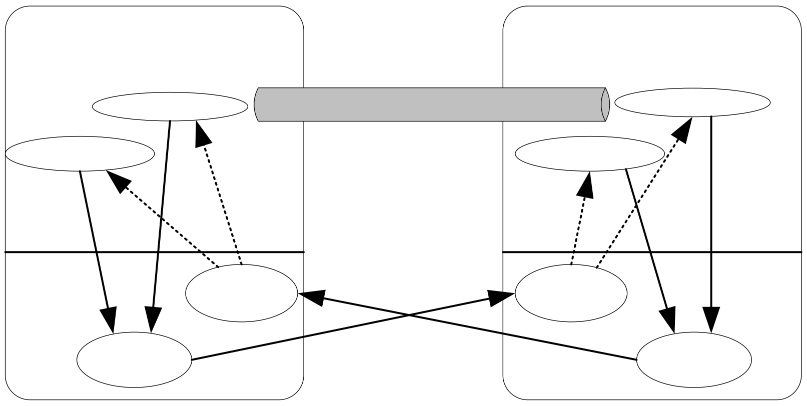


VCluster Implementation

- Advantages of Separate Send/Recv Thread
  - More efficient
    - Sender can continue computing instead of waiting for the message to be sent
    - Receiver may found the message has been received when it need the message
  - Support rendezvous style communication

UCF 22

VCluster Implementation



23

Experimentation with Applications

- Systems compared with VCluster
  - MPICH
    - A C implementation of MPI
  - mpiJava
    - A java wrapper of MPI
  - JPVM
    - A pure java implementation of PVM specification

24

P

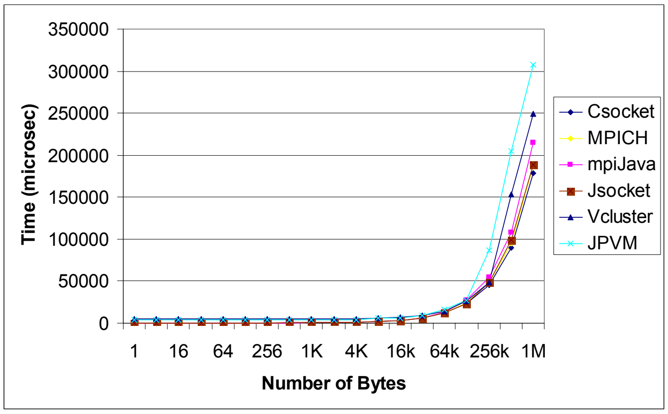
C_threa

Applications - 1

- Communication Overhead
  - 2 processes on two machines
  - Message is sent back and forth between these two processes
  - Roundtrip time for a message to go out and back is measured
  - Tested on messages with size between 1 byte and 1 Megabyte

Applications - 1

- Communication Overhead



Applications - 1

- Communication Overhead
  - VCluster outperforms JPVM for large message size
    - Separate send/recv thread and polling
  - Java based systems are slower, but
    - Coarse grain parallelism is mostly used in cluster computing so the communication overhead can be compensated by the larger computation part

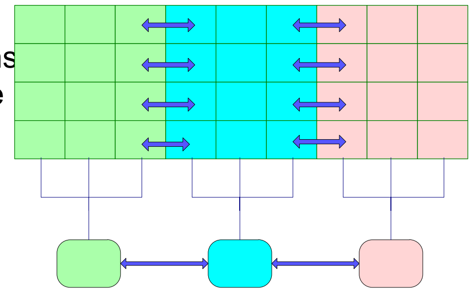
Applications - 2

- Dirichlet Problem
  - A simple numerical simulation problem on a two dimensional grid. Each point on the grid has a location (x, y) and a value T(x, y) representing temperature of some material.
  - At each time step, each point's temperature is averaged with its neighbor's temperatures to find the point's temperature at the end of the time step.

$$T(x, y, t+1) = \frac{T(x, y-1, t) + T(x+1, y, t) + T(x, y+1, t), + T(x-1, y, t) + T(x, y, t)}{5}$$

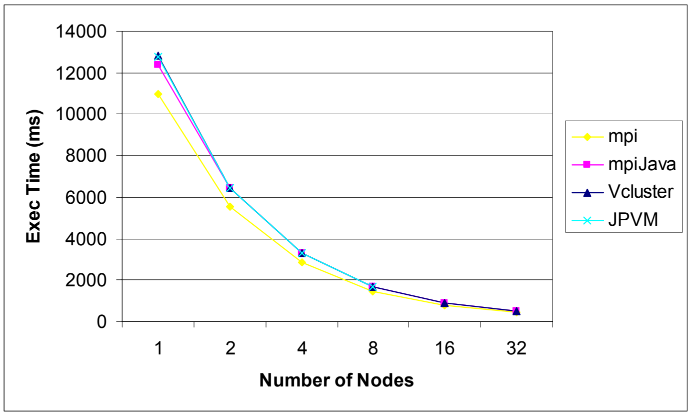
Applications - 2

- Parallel Dirichlet Problem
  - Divide grid into columns each process calculate cells in one column
  - Neighbor processes need to communicate



Applications - 2

- Parallel Dirichlet Problem



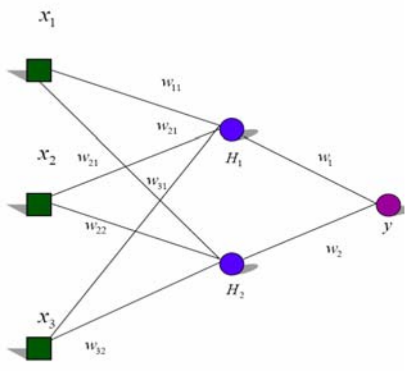
Applications - 2

- Parallel Dirichlet Problem
  - Java based systems have a comparable performance
  - With enough number of processors Java based message passing libraries can mitigate the associated overhead compared with C based libraries and be an attractive tool for a large scale parallel application development

Applications - 3

- Back Propagation Neural Network
  - Predict the value of y for a given input $(x_1,x_2,\ldots,x_n)$
  - Known $(x_1,x_2,\ldots,x_n,y)$ records are used to train the network and find optimized weights
  - Normally more than one iterations of training

$$g_0^{-1}(E(y)) = w_0 + w_1 H_1 + w_2 H_2$$
$$H_1 = g_1(w_{01} + w_{11}x_1 + w_{21}x_2 + w_{31}x_3)$$
$$H_2 = g_2(w_{02} + w_{12}x_1 + w_{22}x_2 + w_{32}x_3)$$



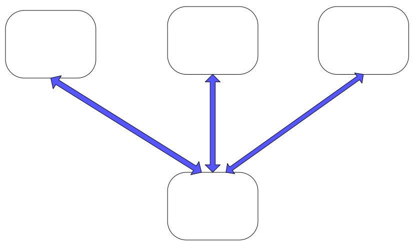
Applications - 3

- Back Propagation Neural Network
  - one of the most popular neural network training algorithms and has shown robust performance in many diverse applications
  - However, computational complexity of the BPNN makes its use challenging especially when the training data set size is huge.

UCF 33

Applications - 3

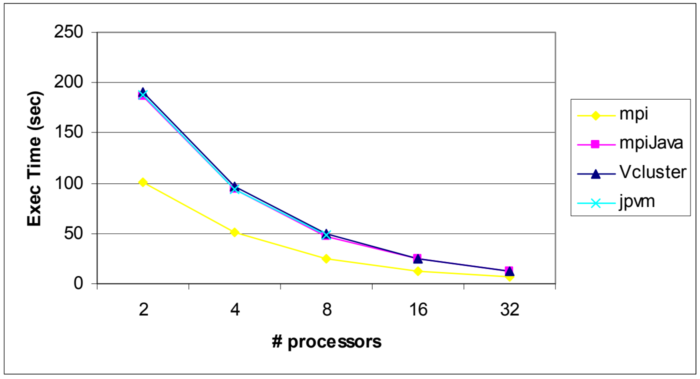
- Parallel Back Propagation Neural Network
  - Divide training data set into smaller sets
  - Each process trains the network use a smaller set
  - The master process collects and redistributes the new weights after each iteration of training, so the communication is between the master process and other worker processes



UCF 34

Applications - 3

- Parallel Back Propagation Neural Network



Applications - 3

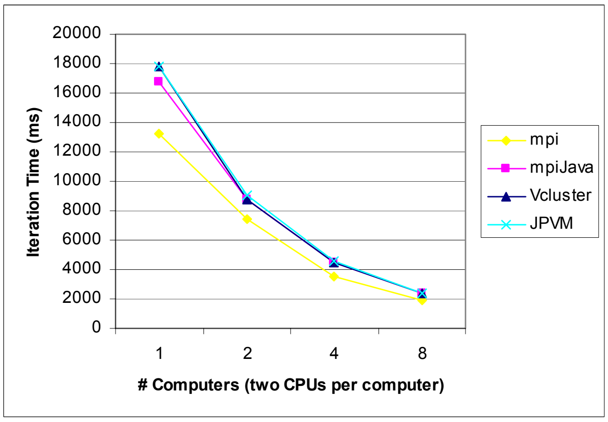
- Parallel Back Propagation Neural Network
  - Java based systems have a comparable performance
  - The gap between VCluster and MPICH converges as the number of processors increases again.

Applications - 4

- Cluster of Multiprocessors
  - VCluster
    - Slightly change to the program
  - MPICH
    - Use a thread library, like pthread
    - Handle thread communication
  - mpiJava and JPVM
    - Use Java thread
    - Handle thread communication

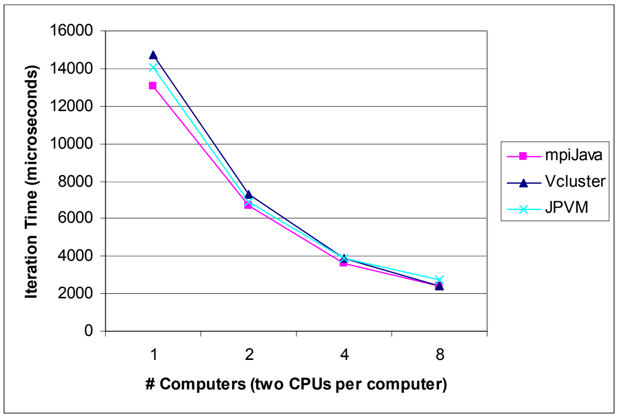
Applications - 4

- Parallel Dirichlet with one thread per node



Applications - 4

- Parallel Dirichlet with two threads per node



Applications - 4

- Dirichlet on Cluster of Multiprocessors
  - On SMP machines multithreading has better performance than single-threading.
  - VCluster outperform JPVM when eight nodes are used
  - Java based systems have a comparable performance, but remember that it takes much more work to write a multithreading program in mpiJava and JPVM, even more in MPICH

Conclusion

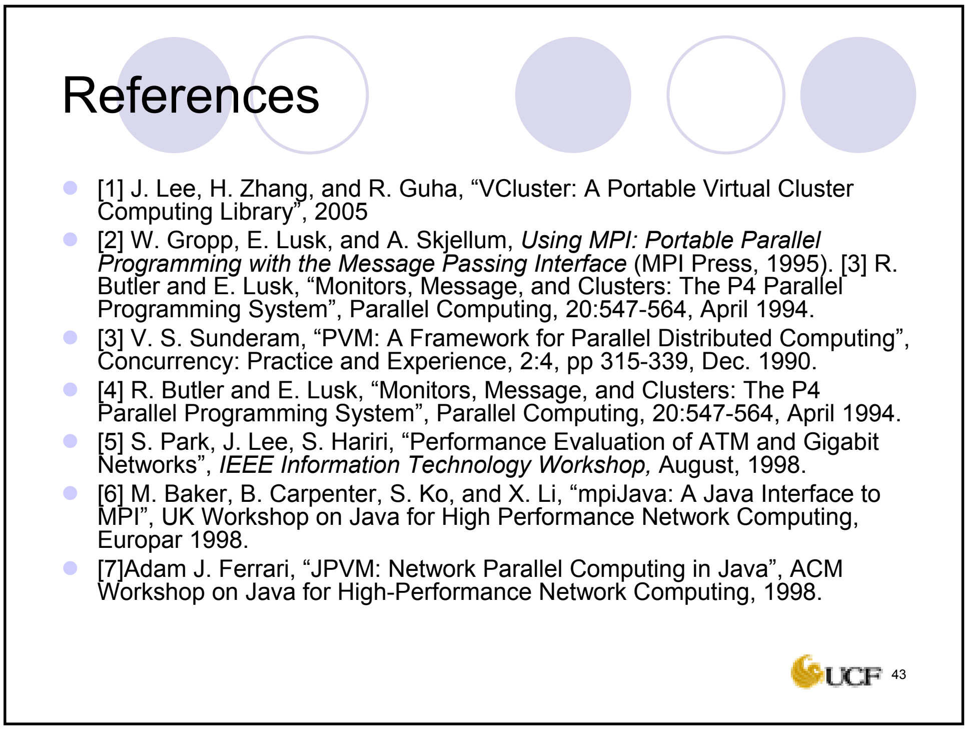
- Prototype of VCluster is comparable to the relevant Java based message passing libraries.
- VCluster provided a more convenient and efficient programming model for application development due to its unique architecture that combines multithreading with communication.
- When enough number of processors are available VCluster can have a performance close to C libraries.

UCF

Future work

- Thread Migration
  - Virtual state
- Dynamic Load Balancing
  - Thread migration
- Security and Fault Tolerance
  - Java cryptography and security packages
  - Thread migration
- Intra-cluster computing

UCF



References

- [1] J. Lee, H. Zhang, and R. Guha, “VCluster: A Portable Virtual Cluster Computing Library”, 2005
- [2] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message Passing Interface* (MPI Press, 1995). [3] R. Butler and E. Lusk, “Monitors, Message, and Clusters: The P4 Parallel Programming System”, Parallel Computing, 20:547-564, April 1994.
- [3] V. S. Sunderam, “PVM: A Framework for Parallel Distributed Computing”, Concurrency: Practice and Experience, 2:4, pp 315-339, Dec. 1990.
- [4] R. Butler and E. Lusk, “Monitors, Message, and Clusters: The P4 Parallel Programming System”, Parallel Computing, 20:547-564, April 1994.
- [5] S. Park, J. Lee, S. Hariri, “Performance Evaluation of ATM and Gigabit Networks”, *IEEE Information Technology Workshop,* August, 1998.
- [6] M. Baker, B. Carpenter, S. Ko, and X. Li, “mpiJava: A Java Interface to MPI”, UK Workshop on Java for High Performance Network Computing, Europar 1998.
- [7]Adam J. Ferrari, “JPVM: Network Parallel Computing in Java”, ACM Workshop on Java for High-Performance Network Computing, 1998.