

SOFTWARE TOOLS

- Several workstation packages for workstation cluster parallel programming: (multiprocessor versions are also available)
- . PVM (Parallel Virtual Machine) - Oak Ridge National Lab
 - homogeneous & heterogeneous workstations
 - C and FORTRAN support
- public-domain (http://www.netlib.org/pvm3/) MPI (Message Passing Interface) - MPI Forum
- provides a message-passing standard
- Several public-domain implementations (http://www.osc.edu/mpi/)
- LAM (Ohio Supercomputing Center) MPICH (Argonne National Lab and Mississippi State University)

Broadcast, Multicast, Scatter, Gather,

Reduce Routines

the group is formed.

 $pvm_mcast($) - is not a group operation, it sends the contents of send

EXAMPLE: A PVM program that adds a group of 1000 numbers together.

Chapter 2 - Parallel Processing

buffer to each process that is listed in a task ID array.

pvm_joingroup() - process joins a named group.

used with a group of processes after

pvm bcast()

• pvm_scatter()

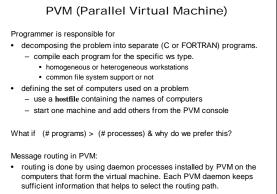
pvm gather()

• pvm_reduce()

Dr. Kivanc Dincer

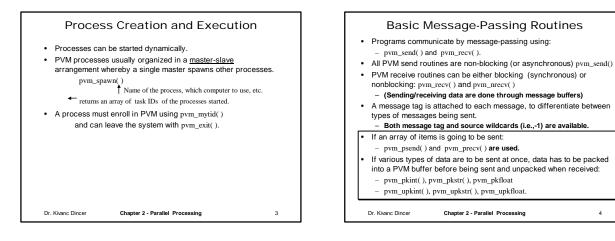
- CHIMP (Edinburgh Parallel Computing Center) UNIFY (Mississippi State University)
- Vendor-Specific proprietary message-passing packages: MPL (IBM) for SP-2, etc.

 - Dr. Kivanc Dincer Chapter 2 - Parallel Processing



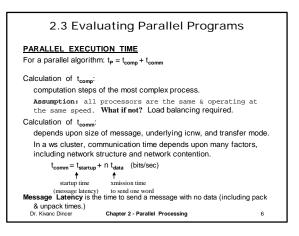
2

Dr. Kivanc Dincer Chapter 2 - Parallel Processing

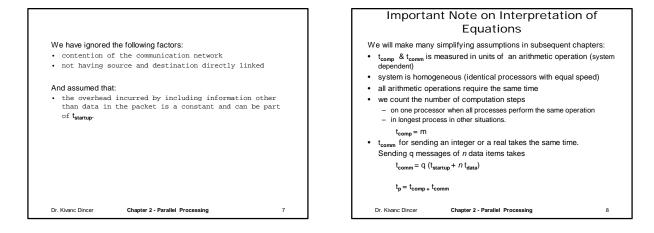


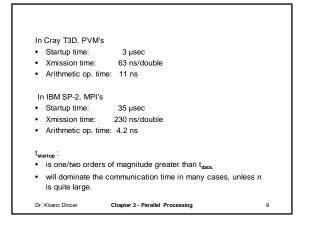
5

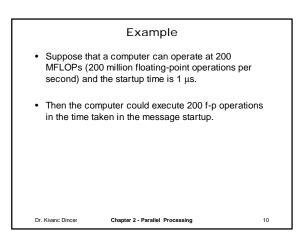
1



1







Latency Hiding

- The deleterious effect on the execution time as shown in previous example is known as the <u>Achilles' heel</u> of message-passing computers.
- Latency Hiding: One way to ameliorate the situation is to overlap the communication with subsequent computations (keeping processor busy with useful work while waiting for the communication to be completed)
 - nonblocking send routines & (locally) blocking send routines.
 - Using parallel slackness
 - mapping multiple processes on a processor (virtual processors) and using time-sharing facility.
 an m-process(or) algorithm implemented on an n-processor
 - an m-process(or) algorithm implemented on an n-processor machine is said to have a <u>parallel slackness of m/n</u> for that machine, where n<m.

11

Using threads

Dr. Kivanc Dincer Chapter 2 - Parallel Processing

Time Complexity of a Parallel Algorithm If we use time complexity analysis, which hides lower terms, t_{comm} will have a time complexity of O(n). Complexity of t_p will be the sum of the computation and communication. Example Problem: Dypose that we were to add n numbers on two computers, where each computer adds n/2 numbers together. • he numbers are initially held by the first computer for adding the two partial sums together.

Computation/Communication Ratio

Communication is very costly.

If both $t_{\text{comp}} \, \text{and} \, t_{\text{comm}} \, \text{has the same complexity} \Rightarrow ?$

Example: N-body problem $t_{\text{comm}} \text{ is } O(N) \text{ and } t_{\text{comp}} \text{ is } O(N^2)$ we can find an N where $t_{\mbox{\scriptsize comm}}$ will dominate the $t_{\mbox{\scriptsize comp}}$

Chapter 2 - Parallel Processing

13

15

Cost-Optimal Algorithms

The cost to solve a problem is proportional to the execution time on a single processor system (using the fastest known sequential algorithm.)

 $Cost = t_p x n = k x t_s$ where k is a constant.

A parallel algorithm is cost-optimal if (Parallel time complexity) x (# processors) = sequential time complexity

Example:

Suppose the best known sequential algorithm for a problem has time complexity of $O(n \log n)$. Are the following cost optimal?

- A parallel algorithm for the same problem that uses *n* processes and has a time complexity of $O(\log n)$.
- A parallel algorithm that uses n² processors and has time complexity of O(1).

14

Dr. Kivanc Dincer Chapter 2 - Parallel Processing

Comments on Asymptotic Analysis

Time complexity is

Dr. Kivanc Dincer

· often used for

- sequential program analysis
- theoretical analysis of parallel programs
- much less useful for evaluating the potential performance of parallel programs:
 - Big-Oh and other complexity notations use asymptotic methods (the variable under consideration to tend to infinity) however
 - · often the # processors are constrained and
 - · data sizes are finite and manageable.
 - Analysis often ignores lower terms that could be important (e.g., t_{startup} dominates overall communication time when n is large)
 - Analysis also ignores other factors that appear in real computers, such as communication contention.

Dr. Kivanc Dincer Chapter 2 - Parallel Processing