Scheduling Message Passing

(what Meta-Computing systems need to do)

CS-594

Dr Graham E. Fagg

Spring 2001

Overview

– Covers details you may have missed in David Walkers MPI / Message Passing class.

– Assumes that you know and understand MPI and PVM.

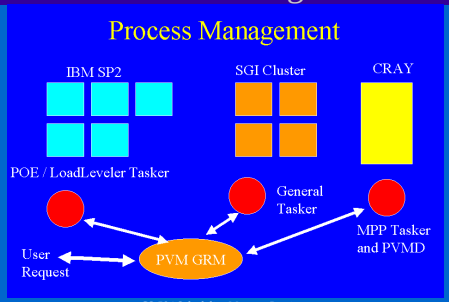
Closely coupled vs Cluster Computing

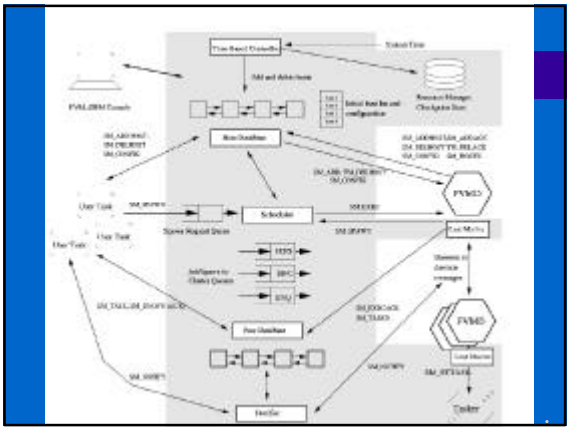
- Bottom line
  - MPI is better at message passing than PVM
  - More complex
  - Less flexible at anything else
    - I.e. it's a message passing system not a distributed environment

Scheduling

- Not built into MPI as it has no process control
  - But maybe an option under MPIRUN
  - Yep PVM has it all
    - user controllable pvm_spawn()
    - pvm_rm interface
      - also
        - pvm_tasker interface
        - pvm_hoster interface

Scheduling





Scheduling

- Two types to worry about
  - At spawn time
    - static allocation based on the environment
  - At run time
    - I.e. migration of tasks
      - system level migration
        - Special support needed (Condor)
      - User level
        - check points / restarts
    - Change work load allocated (bag of tasks)

Task allocation in PVM I.e. pvm_spawn()

- Before improving on it, had to figure out how it worked as it wasn't random but round-robin
- Aimed at using spare capacity
  - what spare capacity??

What is spare (what is even machine load?)

- Condor people claimed 10% utilisation for their systems
  - At Reading was more like 40-60% all the time.
- Load
  - machine average is not a good metric but without more specific help from the kernel it would have to do.
- Defined user classes and loading based on observations on the RDG system over a year...

Typical loading

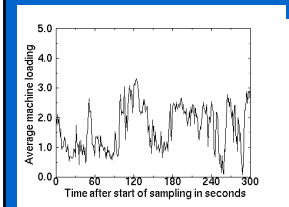


Figure: 4.14 Loading on a Meteorology Department machine sssvcm2.

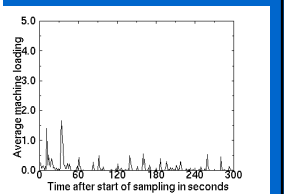
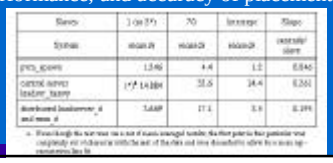


Figure: 4.15 Loading on a Psychology Department machine sssvcm1.

Better Spawn

- Added checks for load before starting.
  - Based on two methods, central RPC and distributed monitor daemons
  - Checked effects of this system on startup performance, and accuracy of placement.



Application Performance

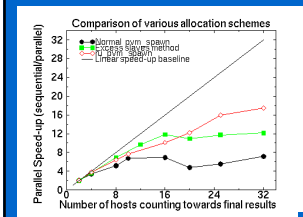
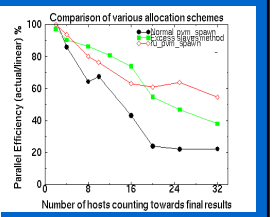
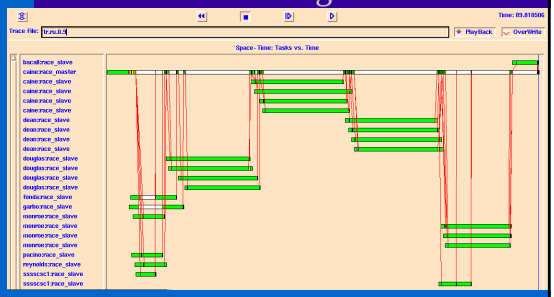


Figure: 4.21 Parallel speed-up of PNG Monte-Carlo simulations.



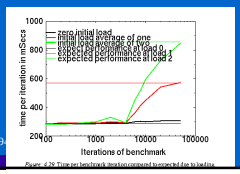
User level migration

User level migration



Back to load or not

- The load was not always a good measure and soon the network was very heterogeneous…
- Could you use a benchmark to find the real load?



Modified spawn using effective speed-up

- Takes into account benchmark and loading
  - Helps on heterogeneous networks

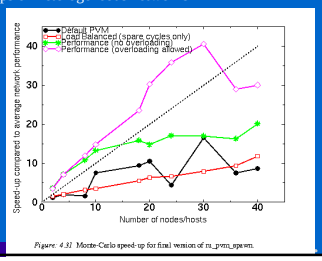


Figure: 4.31 Monte-Carlo speed-up for final version of rs_pvm_spawn.

Very advanced Schedulers

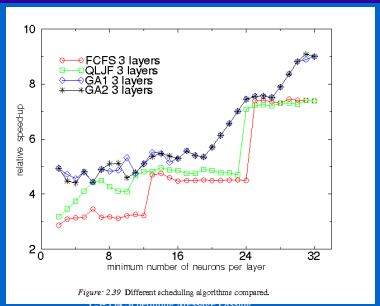


Figure: 2.39 Different scheduling algorithms compared.

Fault Tolerance

- Multiple methods
  - System level
    - from checkpoint file
  - User level
    - from data check point
  - How do we detect the failure?
    - Notify Message and/or time out