THE UNIVERSITY OF WARWICK

High Performance Computing *Course Notes*

Cluster Technologies -I

Dr Ligang He

History and Evolution of HPC Systems (revisit)

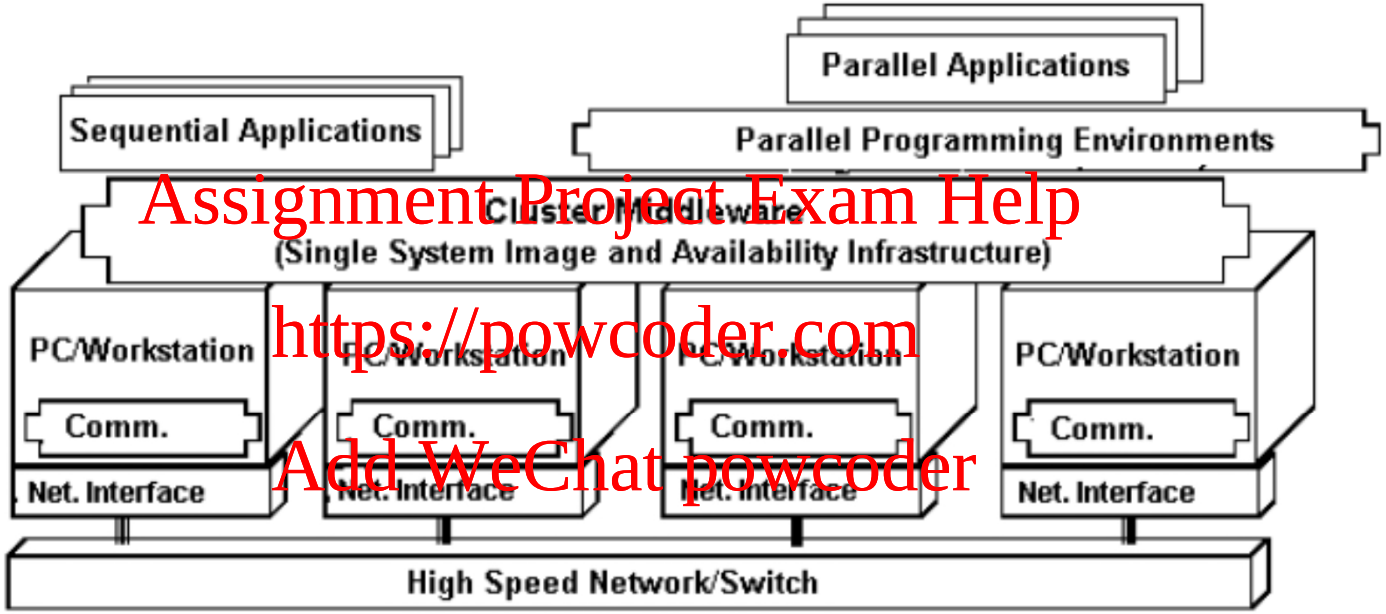
- 1960s: Scalar processor
- 1970s: Vector processor
- Later 1980s: Massively Parallel Processing (MPP)
- Later 1990s: Cluster
  - Connecting stand-alone computers with high-speed network (over-cable networks)
    - Commodity off the shelve computers
    - high-speed network: Gigabit Ethernet, infiniband
    - Over-cable network vs. on-board network
  - Not a new term itself, but renewed interests
    - Performance improvement in CPU and networking
    - Advantage over custom-designed mainframe computers: Good portability

Why did Clusters gain popularity then?

Clustering gained new wave of interests when 3 technologies converged:

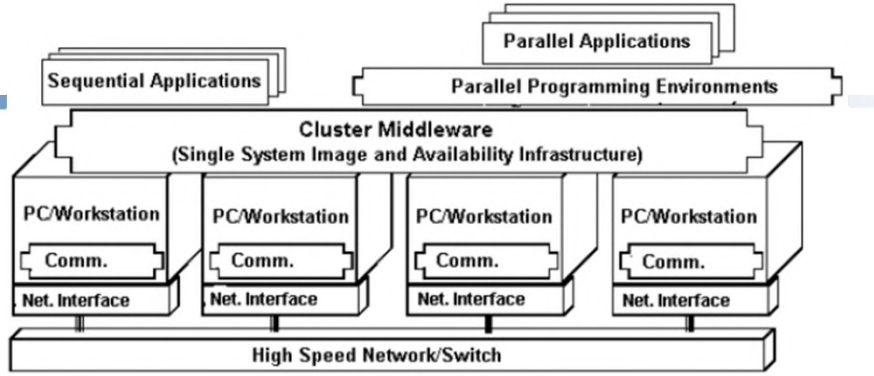
1. Very high performance Microprocessors
   - PC performance today = old time supercomputers
2. High speed communication
3. Standard tools for parallel/ distributed programming

Cluster Architectures



Cluster system architecture

Cluster Components...1
Nodes

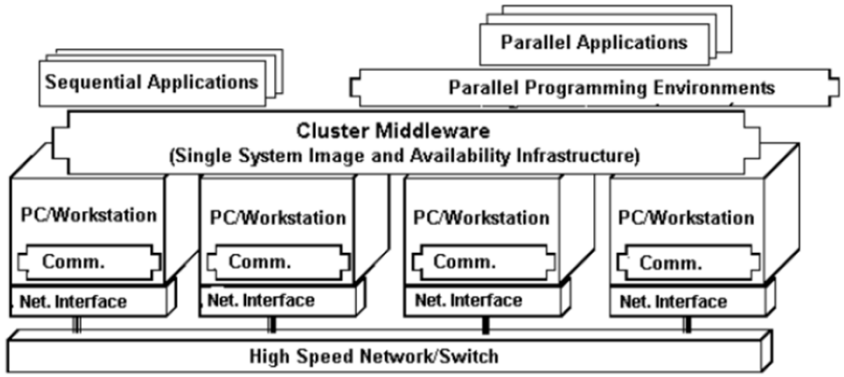


- Multiple High Performance Components:
  - PCs
  - Workstations
  - SMPs
- They can be based on different architectures and running different OS
- But usually, the nodes in a cluster are homogenous
  - They have same architecture and performance, and are installed with the same os

Cluster Components…2
OS

- **OS used in various cluster systems :**

Linux	(Beowulf)
Microsoft NT	(Illinois HPVM)
SUN Solaris	(Berkeley NOW)
IBM AIX	(IBM SP2)
HP UX	(Illinois - PANDA)
Mach	(Microkernel based OS)(CMU)



Cluster Components…3
High Performance Networks

Ethernet (10Mbps),

FDDI (100Mbps): Fibre Distributed Data Interface

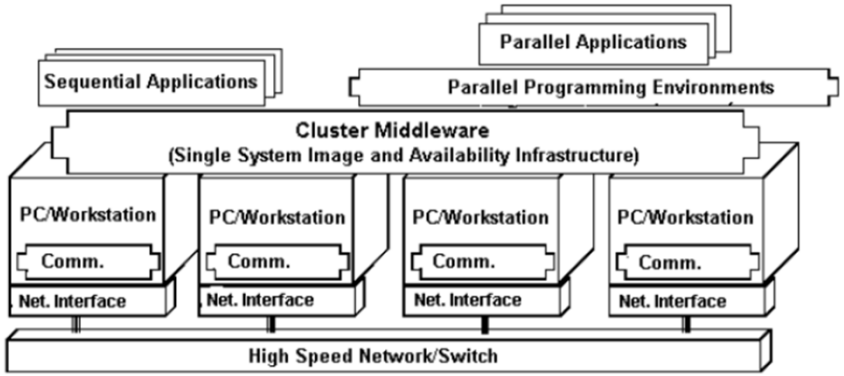
Fast Ethernet (100Mbps),

Gigabit Ethernet (1Gbps)

Myrinet (10Gbps)

10 Gigabit Ethernet (10Gbps)

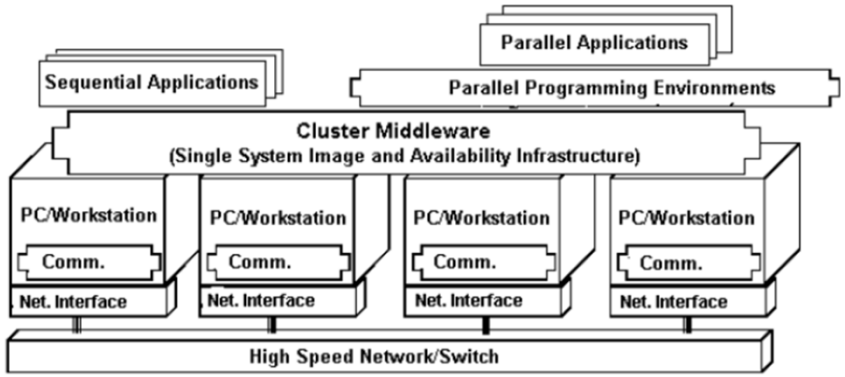
Infiniband (24-290Gbps)



Cluster Components…4
Network Interfaces

Network Interface Card

- Myrinet NIC
- Ethernet NIC
- Infiniband NIC
- …



Cluster Components…5 Communication Software

Traditional OS supported protocols (heavy weight due to protocol processing)..

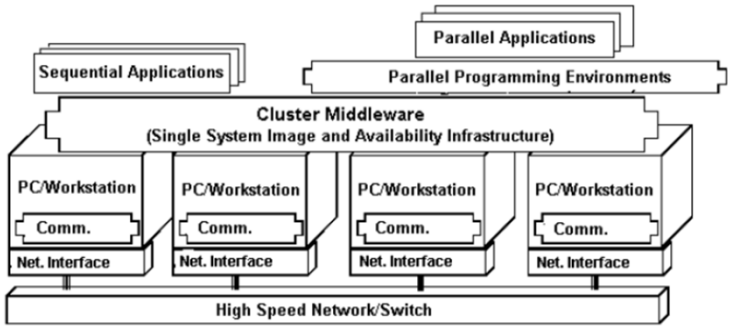
- 7 layer OSI reference model,
- Sockets (TCP/IP), pipes, etc.

Light weight protocols (User Level)

- Active Messages (Berkeley): used in NOW system
- Fast Messages (Illinois): used in HPVM
- U-net (Cornell)
- XTP (Virginia)

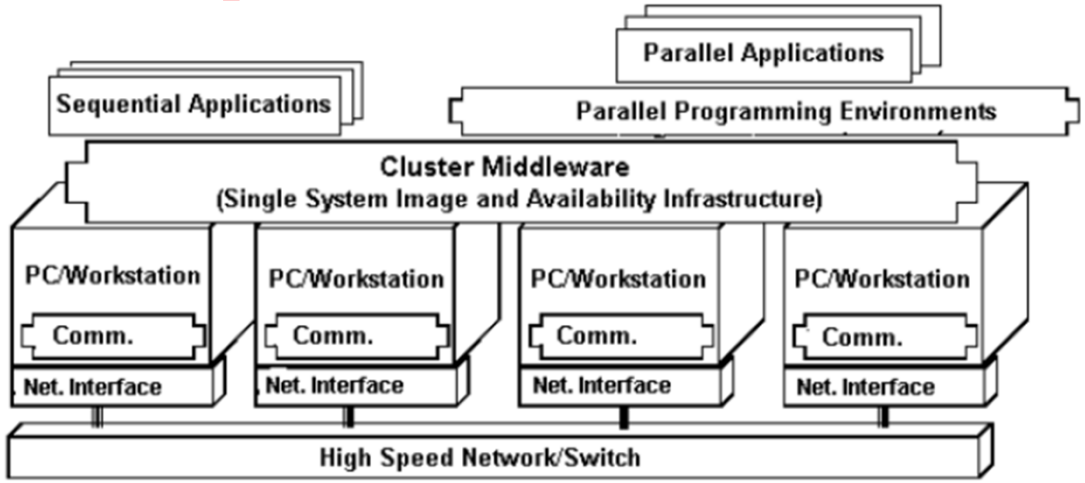
Active Message:
http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-92-675.pdf

Fast Messages:
https://courseware.ee.calpoly.edu/~jharris/3comproject/Reference/high%20performance%20messaging%20on%20workstations.pdf

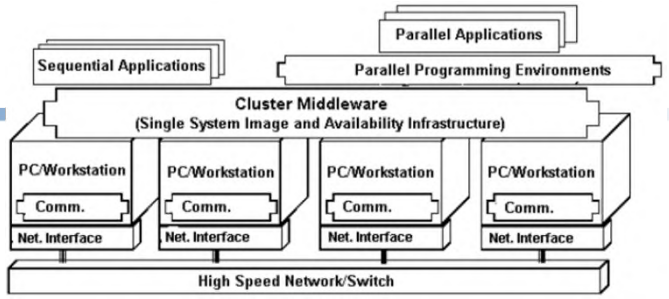


Cluster Components…6
Cluster Middleware

- Provide workload and resource management
- Present the single system image of the cluster
- Examples:
  - Moab
  - SLURM
  - PBS
  - Condor



Cluster Components…8 Development Tools



Processes: MPI, PVM, DSMs

Threads (Multicore computers)

OpenMP, POSIX Threads, Java Threads

Compilers

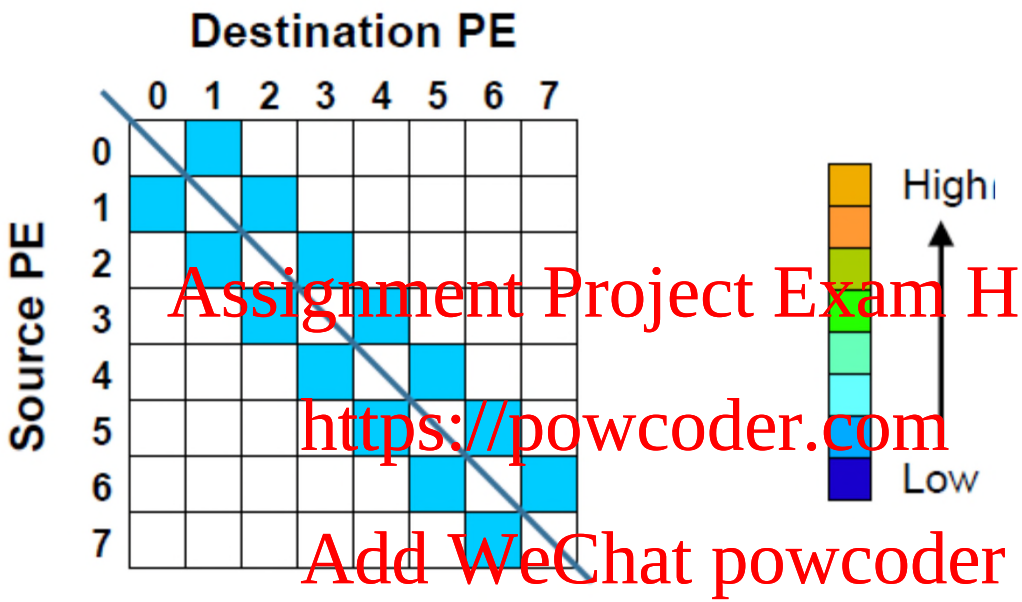
- **C/C++/Java:**
- **MPICC**

Debugger for sequential programs: gdb and dbx

Debuggers for parallel programs: Buster

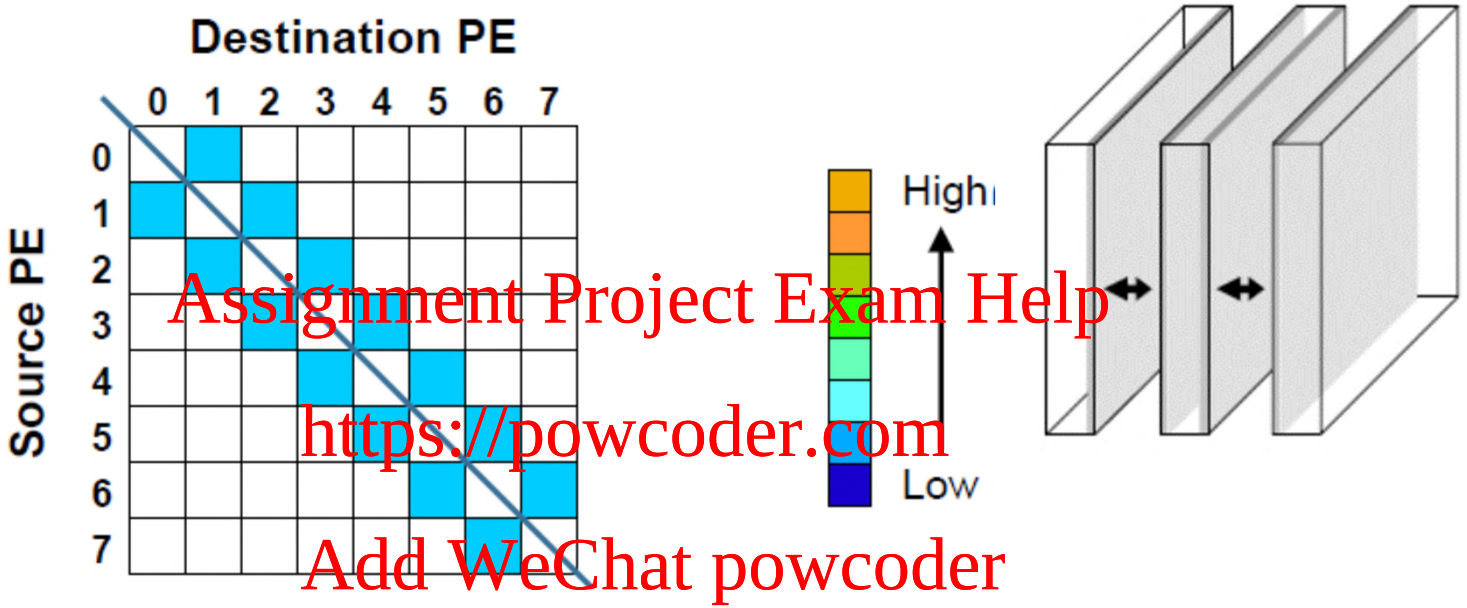
Performance Analysis Tools and Visualization Tools: e.g. Vampir Trace

Vampir Visualizes Communication Pattern



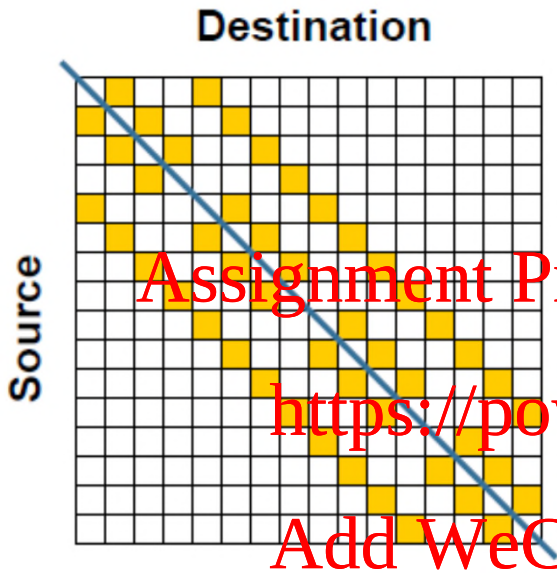
- Output from several parallel performance profilers – e.g. VampirTrace.
- Example above shows nearest neighbor communications for a 1-D data decomposition (Each PE sends to PE+1, and PE-1).
- Symmetrical iff equal data flow between sub-grids in both directions.
- Communication patterns can be identified from the matrix.

Vampir Visualizes Communication Pattern



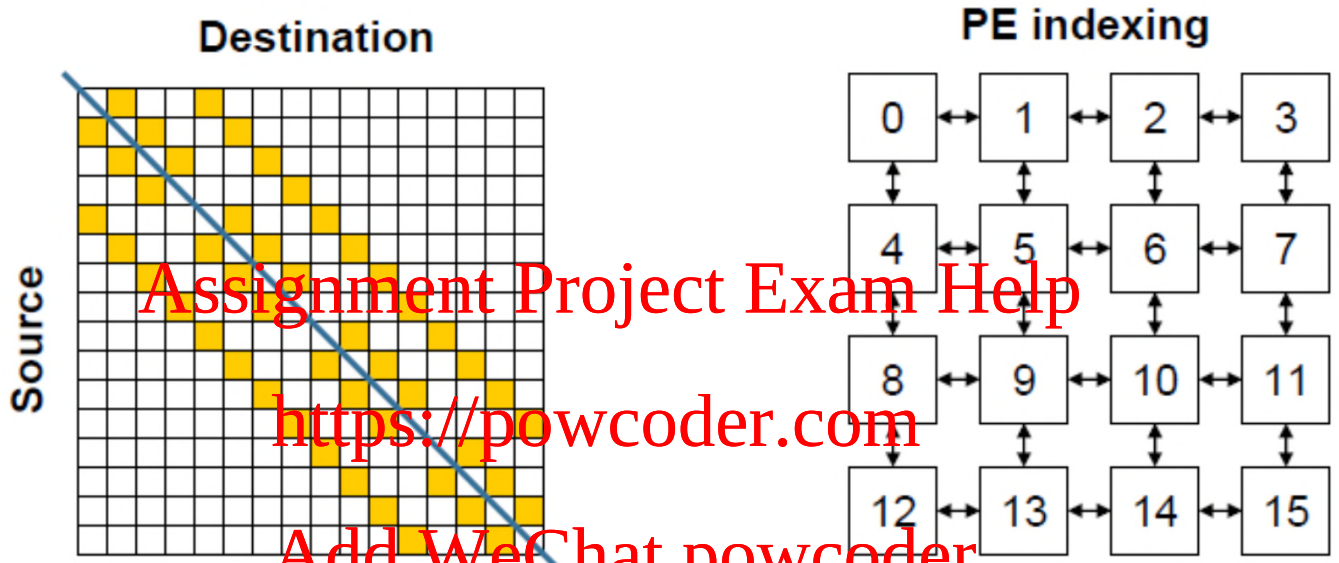
- Output from several parallel performance profilers – e.g. VampirTrace.
- Example above shows nearest neighbor communications for a 1-D data decomposition (Each PE sends to PE+1, and PE-1).
- Symmetrical iff equal data flow between sub-grids in both directions.
- Communication patterns can be identified from the matrix.

Vampir Visualizes Communication Pattern



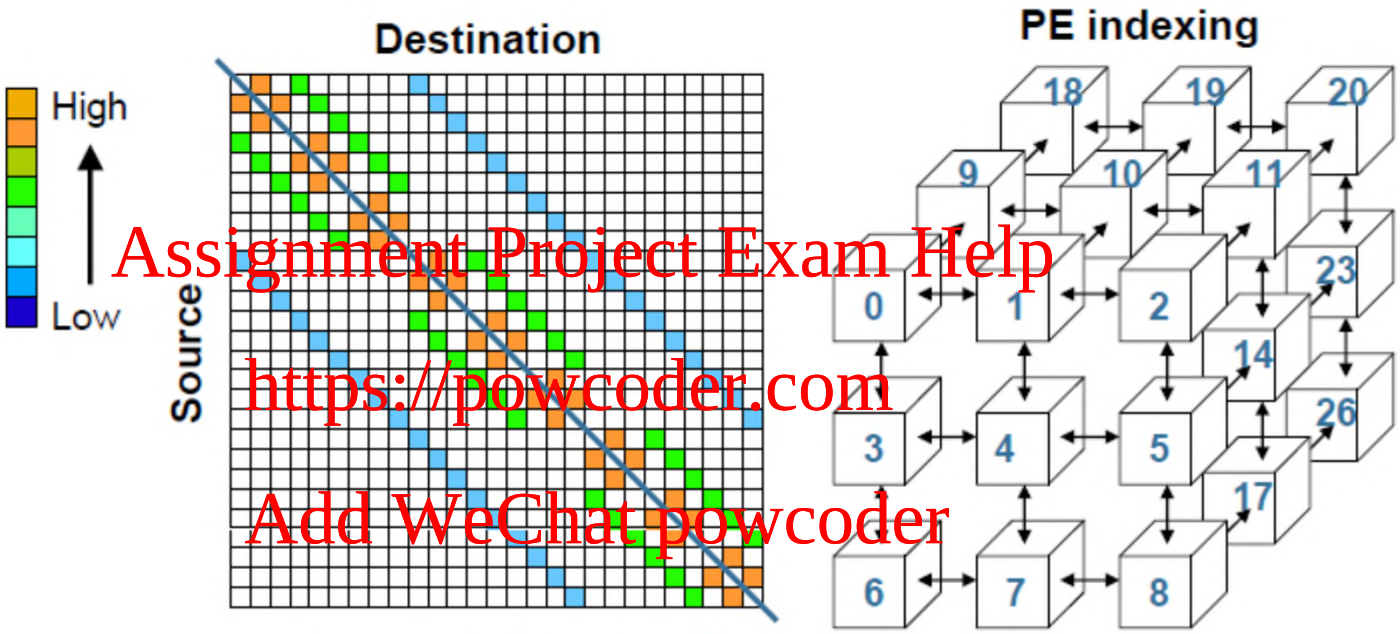
- Communication pattern typical of a 2-D decomposition.
- Equal amount of traffic (and messages) occur in shaded locations (in this example).

Vampir Visualizes Communication Pattern



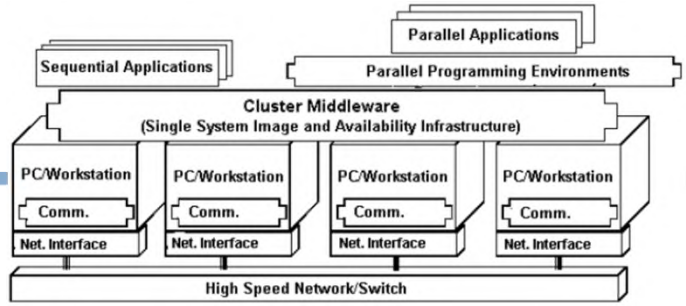
- Communication pattern typical of a 2-D decomposition.
- Equal amount of traffic (and messages) occur in shaded locations (in this example).

Vampir Visualizes Communication Pattern



- Communication pattern for a 3-D decomposition.
- Level of traffic in X > Y > Z (in this example).

Cluster Components…9 Applications



Sequential application

Parallel / Distributed application

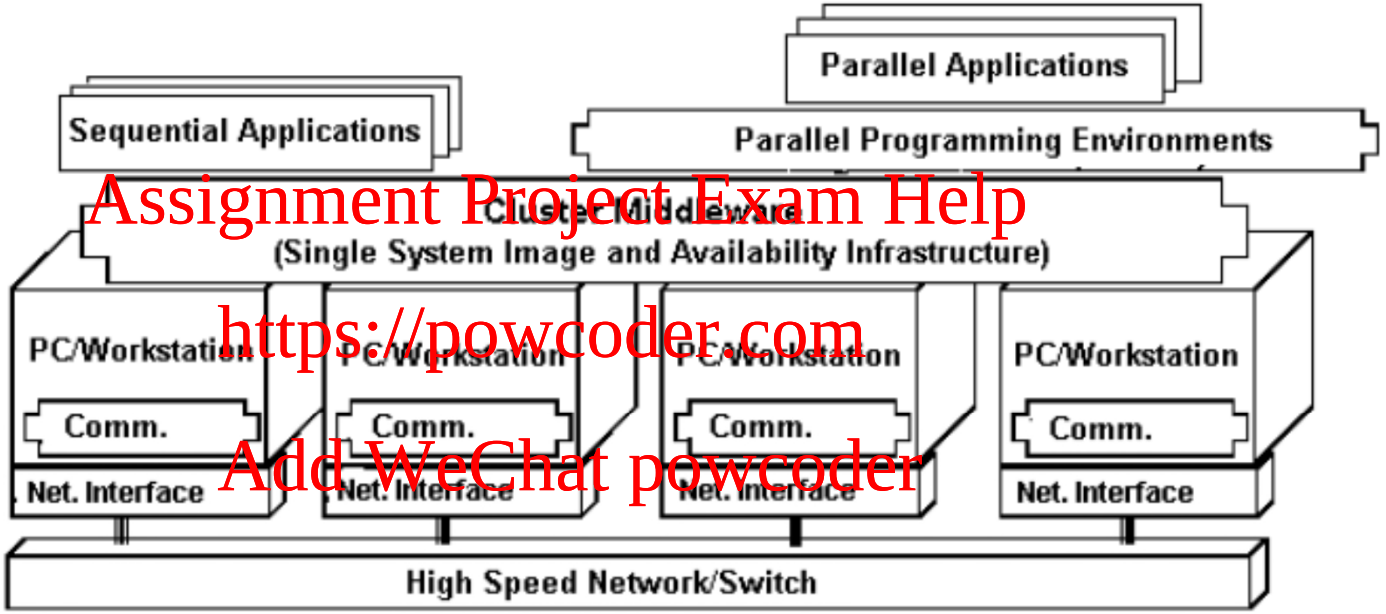
Scientific applications: each is computation-intensive

- Weather Forecasting
- Computational AI
- Molecular Biology Modeling
- Engineering Analysis (CAD/CAM)
- ……

Service applications: high arrival rate of service requests

- Google
- Ebay
- amazon

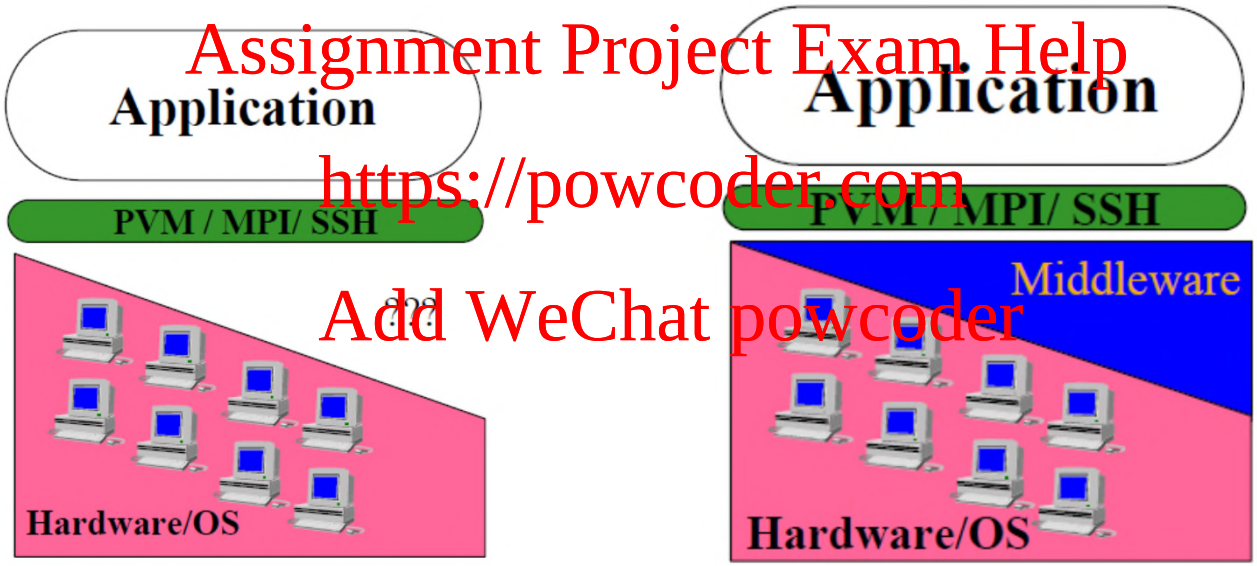
Cluster Architectures



Cluster system architecture

What is Single System Image (SSI) ?

A single system image is the illusion, created by Cluster management software (middleware), that presents a collection of resources as a single powerful resource.



Single Entry Point

ssh cluster.my_institute.edu √

ssh node1.cluster. institute.edu ×

Benefits of SSI or Middleware

- Simplified system management
- Use system resources transparently
  - Users need not be aware of the detailed resource information and underlying system architecture to use these machines effectively
- Transparent load balancing and process migration across nodes.
- Improved reliability and availability
- Improved system-oriented performance
  - Global view of middleware and local view of a user

Cluster Management Software

- **Goal: Help the allocation of resources to jobs, given jobs' resource requirements and local policy restrictions**
- **Three parties in a cluster environment**
  - **Users: supplying the job and job requirements**
  - **Administrators: describing local use policies**
  - **Cluster management software: monitoring the state of the cluster, scheduling the jobs and tracking the resource usage**
- **Typical activities performed by cluster management software**
  - **Queuing**
  - **Scheduling**
  - **Monitoring**
  - **Resource management**
  - **Accounting**

Queuing

- Job submission usually consists of two primary parts:
  - Job description (e.g. job name, the location of the required input files)
  - Resource requirements (e.g. the amount of memory, the number of CPUs needed)


```
#!/bin/bash
#MSUB -q etc
#MSUB -l nodes=2:ppn=16
#MSUB -l pmem=128mb
#MSUB -l walltime=00:5:00
srun ./helloMPI
```


- Once submitted, the jobs are held in the queue until the job is at the head of the queue and the matching resources are available

```
showq -u your_username
```

Scheduling

- Determining at what time a job should be put into execution on which resources
- There are a variety of metrics to measure scheduling performance
  - System-oriented metrics (e.g. throughput, utilisation, average response time of all jobs)
  - user-oriented metrics (e.g. response time of a job submitted by a user)
  - They can contradicts each other and balance needs to be made

Monitoring

- Providing information to administrators, users and the Cluster manager on the status of jobs and resources
- The method of collecting the info may differ between different cluster management systems, but the general purpose is the same

Resource management

- → Handling the details of
  - ❑ Starting the job execution on the resources
  - ❑ Stopping a job
  - ❑ Cleaning up the temporary files generated by the jobs after the jobs are completed or aborted
  - ❑ Removing or adding resources

- → For the batch system, the jobs are put into execution in such a way that the users don't have to be present during execution
- → For interactive systems, the users have to be present to supply arguments or information during the execution of the jobs.

Accounting

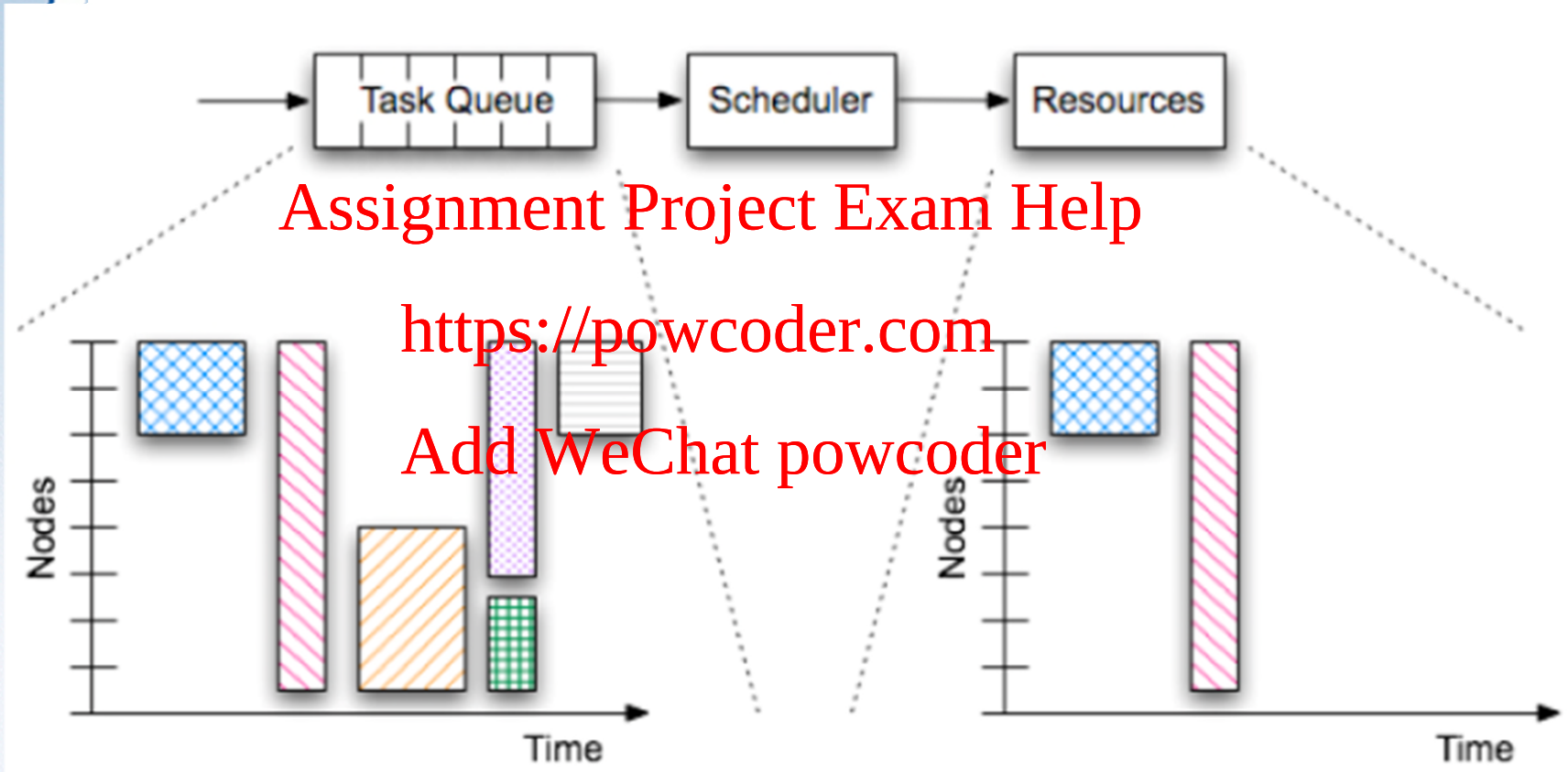
- → Accounting for which users are using what resources for how long
- → Collecting resource usage data (e.g. job owner, resources requested by the job, resource consumption by the job)
- → Accounting data can be used for:
  - ❑ Producing system usage and user usage reports
  - ❑ Tuning the scheduling policy
  - ❑ Anticipating future resource requirements by users
  - ❑ Calculating future resource allocations
  - ❑ Determining the area of improvement within the cluster

Schedule Polices

The simplest policy:

- First-Come First-Served
- Jobs are run in the same order as they are submitted.
- Does not require prior knowledge about jobs (e.g. runtime).
- Problems: jobs can block other jobs from starting, despite there being no performance benefit to either user.

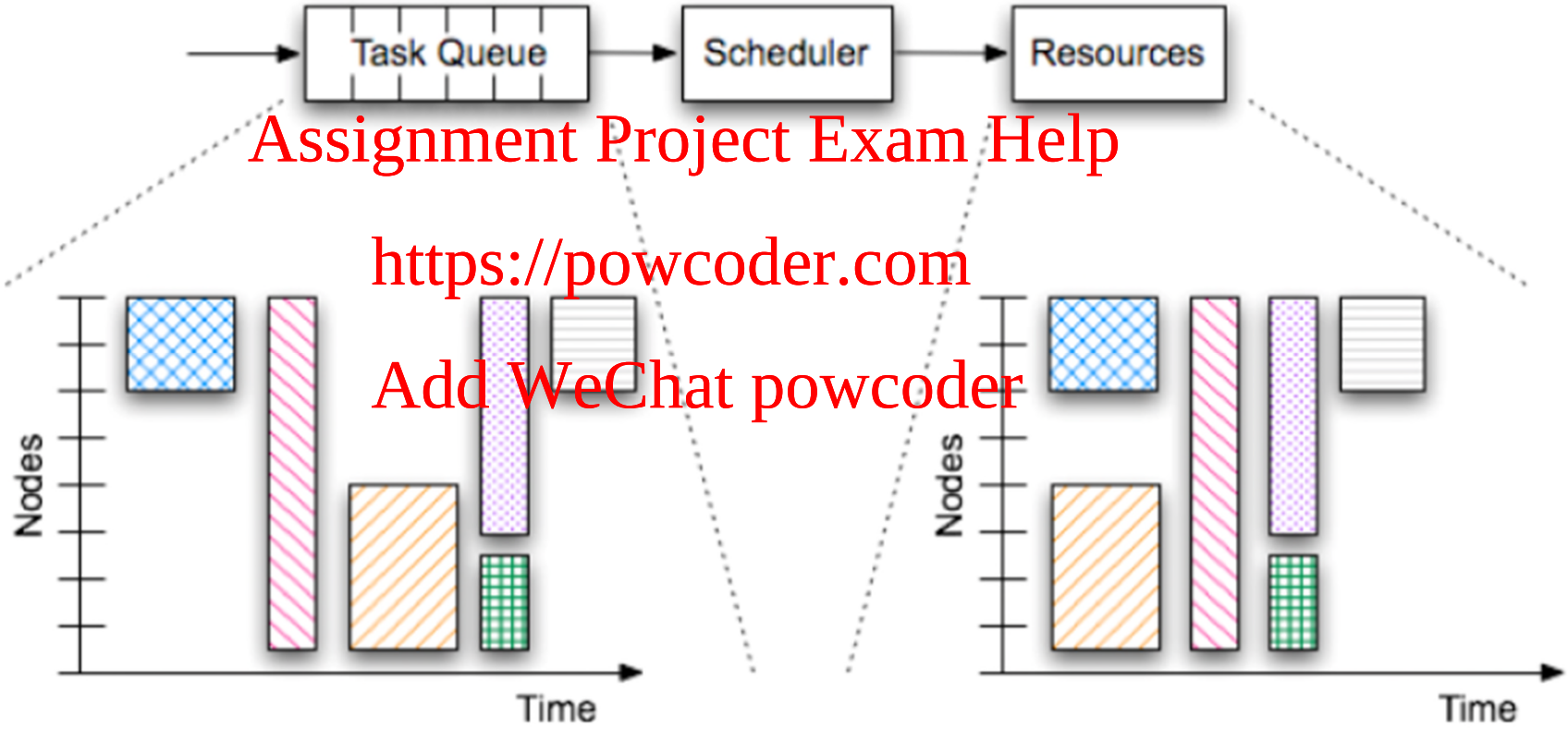
First-Come First-Served



Backfilling

- The problem with FCFS is that idle time (sum of unused processing intervals) can be significant.
- One improvement is to "backfill".
- Allows a job to start if it does not delay the execution of the first job in the queue.

Backfilling

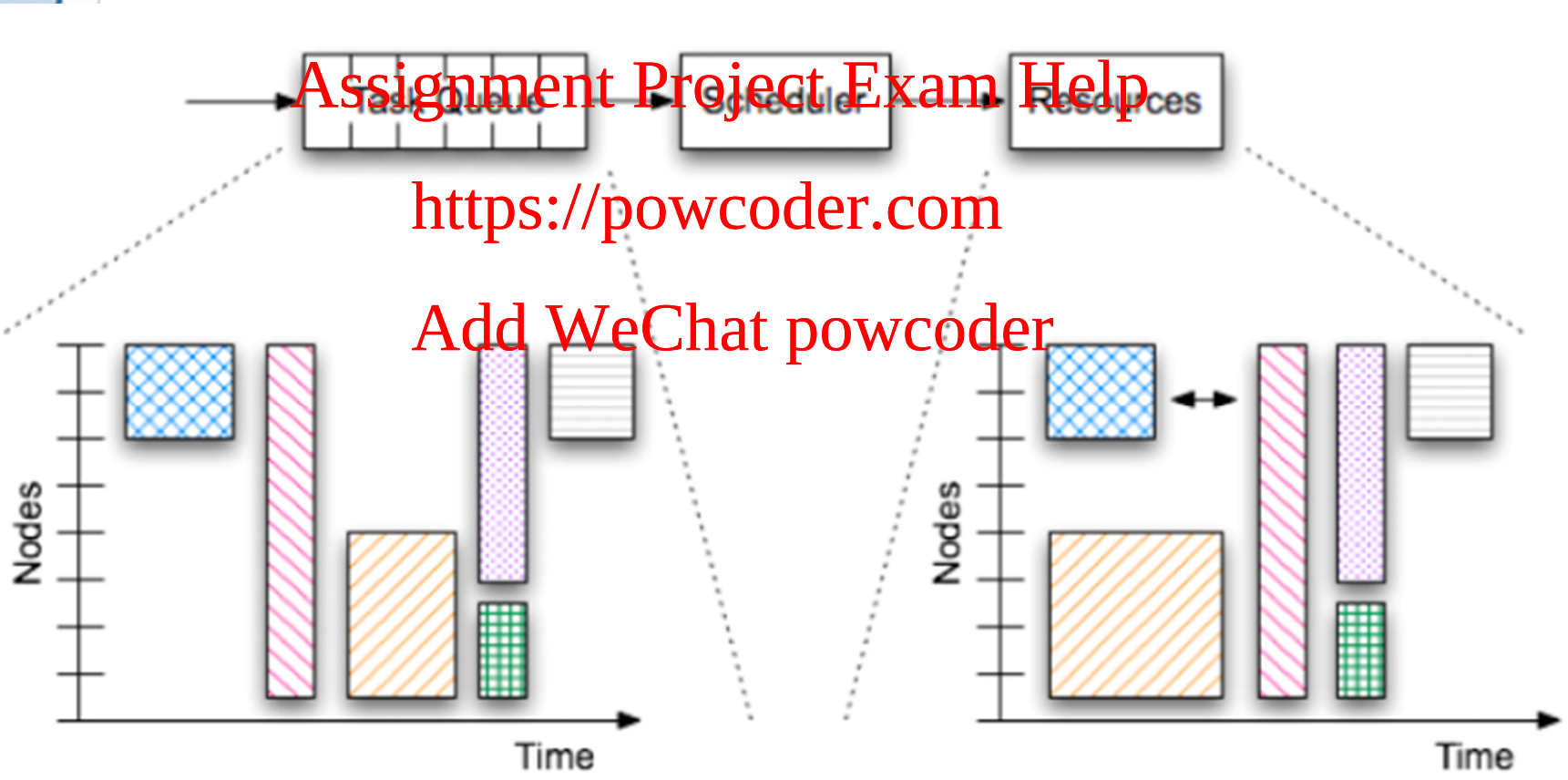


Backfilling

- **Advantages:**
  - **Utilisation is improved.**
- **Disadvantages:**
  - **Information about the job execution time is required.**
  - **User estimation are usually inaccurate.**
  - **It is a policy decision to decide what to do if a job overruns; many administrators choose to terminate a job if it exceeds its allocated execution time otherwise some users may deliberately underestimate the job length to get an earlier job start time.**

Backfilling

A problem if predicted runtime is wrong:



Schedule Polices

Reservation:

- Increasingly user-based quality of service (QoS) is an important scheduling metric.
- In addition to normal scheduling, reservation services can be used to plan resource allocation.
- Users are able to
  - set up a reserved block of processing capability that they are able to use at some point in the future.
  - reserve a part of resources in the cluster to be dedicated to a certain group of users