Metode şi Algoritmi de Planificare (MAP)

2009-2010

Curs 3

Taxonomia metodelor si algoritmilor de planificare

Outline

- General presentation
- Distributed systems layer
- Scheduling Taxonomy for Distributed Systems (Grid)
- Scheduling models
- Taxonomy of Systems
- Taxonomy of Resources
- Taxonomy of Applications

Scheduling in distributed systems

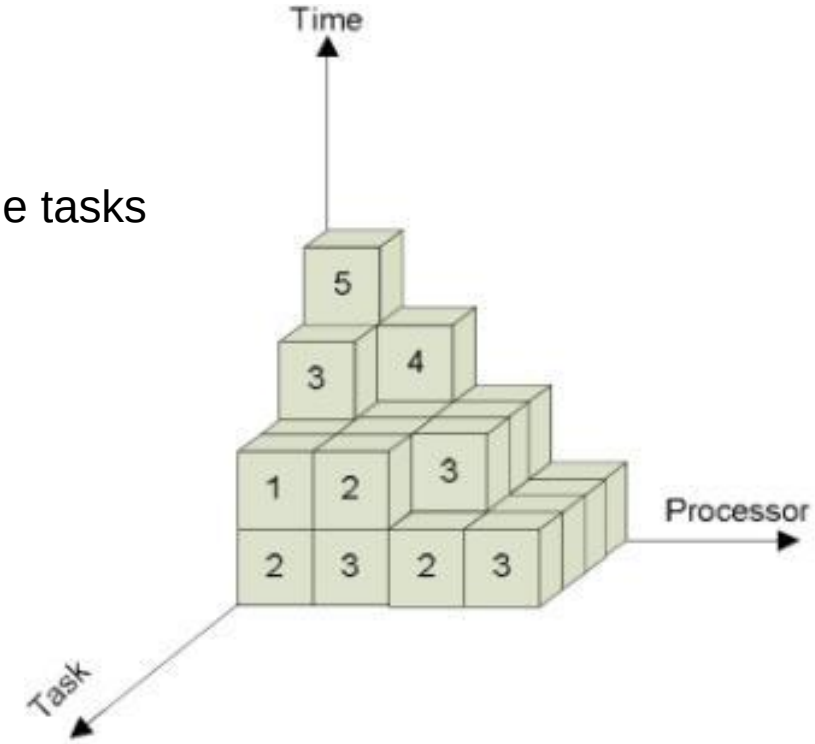
- **The scheduling problem**
  - Allocate a group of different tasks to available resources
  - NP-Complete problem
  - Optimizations of approximate algorithms are required
  - Dynamic (available) resource allocation

- **Tasks**
  - With dependences | Without dependencies
  - With preemption | Without preemption
  - With specific constrains

- **Environment: Computational and data GRID**
  - *Heterogeneous* – resources with different processing capacities
  - *Shared* – resources can be used by multiple users

Scheduling in distributed systems

- **Objectives**
  - Optimize the application execution
    - Minimize the total execution time of the tasks
    - Efficient use of computing resources
  - Successful completion of tasks
    - Deadline restrictions
    - Resource requirements
  - Optimize the scheduling process

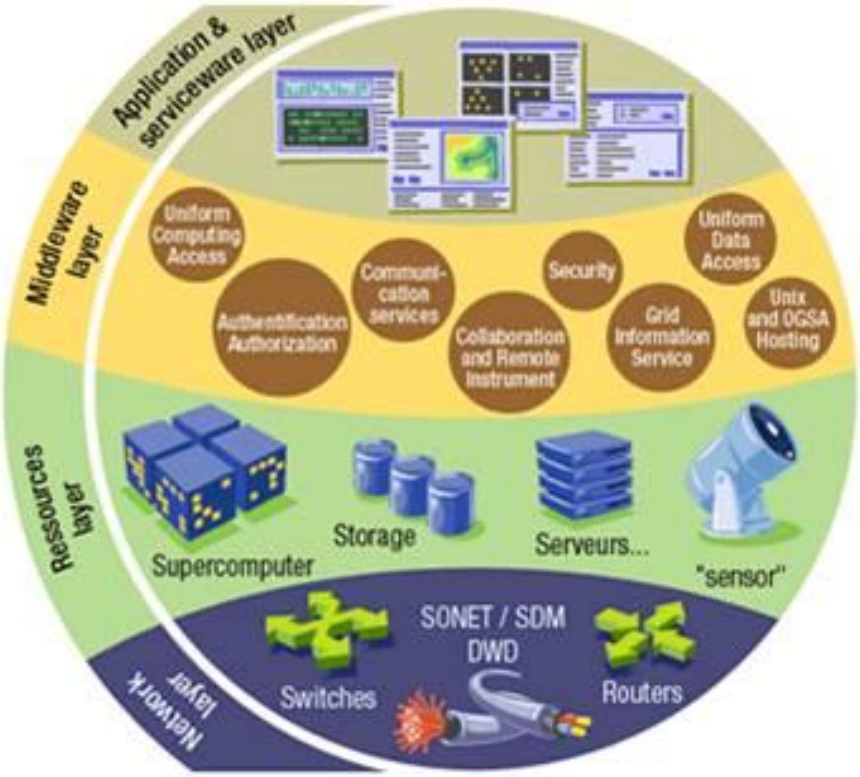


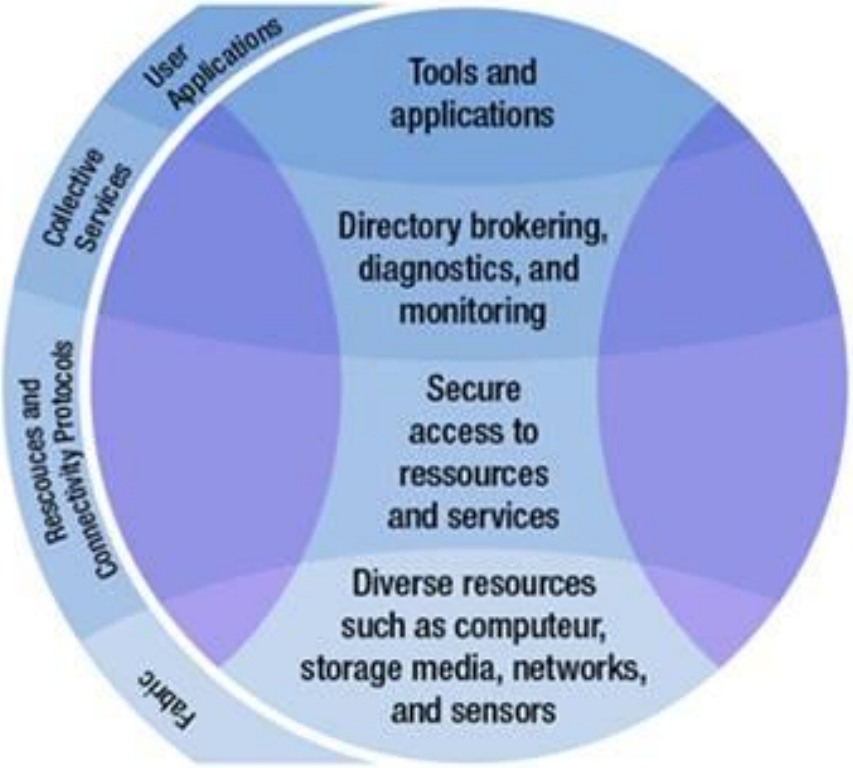
- **R** (P, Q, J, F, O) **| r_j; d_j; p_j; pmtn; prec |** C_{max} $(\Sigma \omega_j C_j)$

Scheduling in distributed systems

- In distributed computing, *sites can participate in by offering resources*, provided that certain conditions such as security requirements are met.
- The interaction between distributed resources during the execution of a job *requires a scheduling* layer that uses a different scheduling paradigm.
- *Scheduling process requires interaction to remote sites* and their local scheduling systems
  - => the use of *several scheduling layers for the Grid*.
- Different level for scheduling process:
  - lower-level scheduling instances, used for the *local scheduling of resources*, and
  - *higher-level scheduling instances*, used for interaction in coordinated scheduling in a distributed system.
- *Solution:*
  - *Decentralized scheduling*
  - *Use of monitoring as feedback*

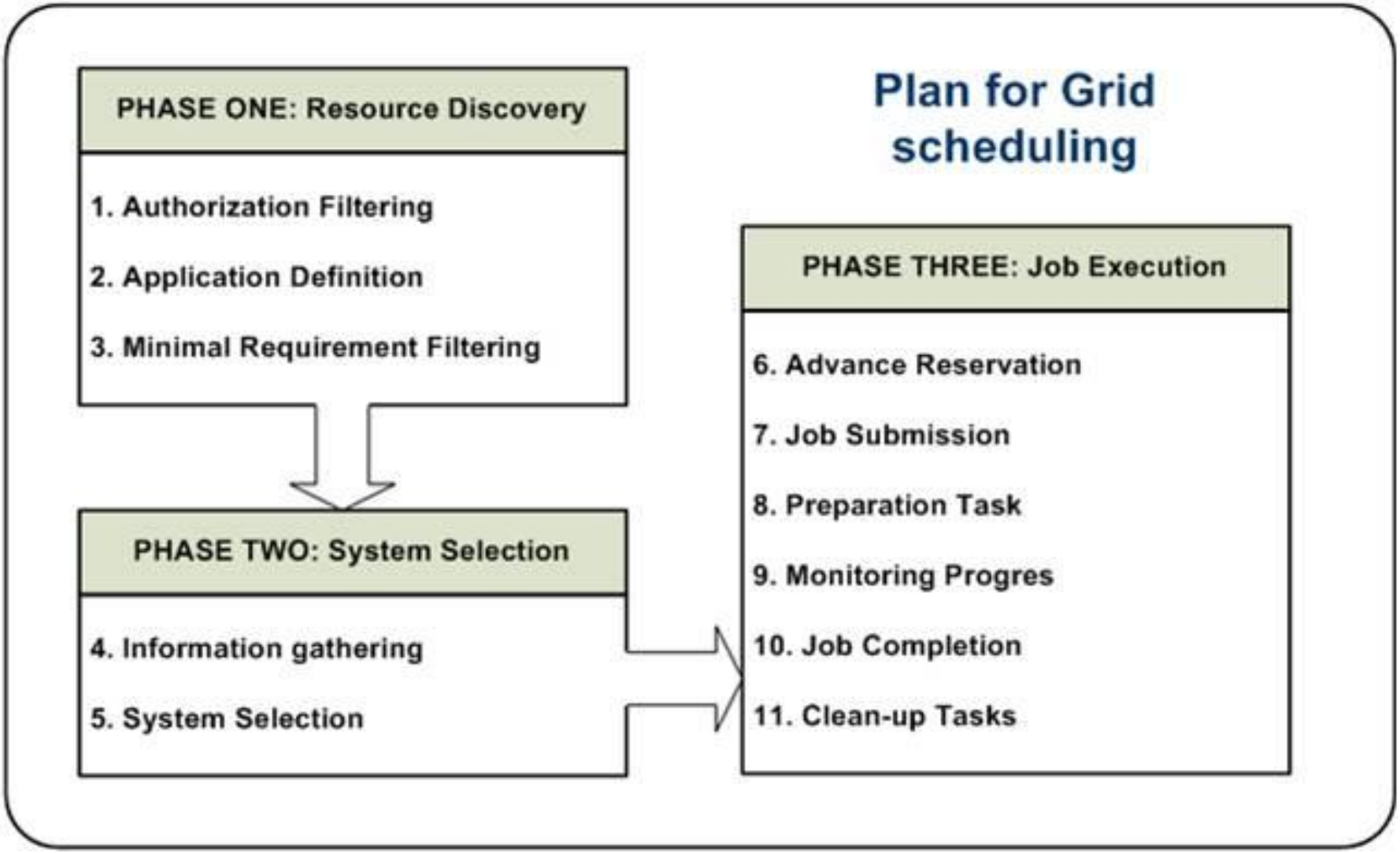
Distributed System Levels



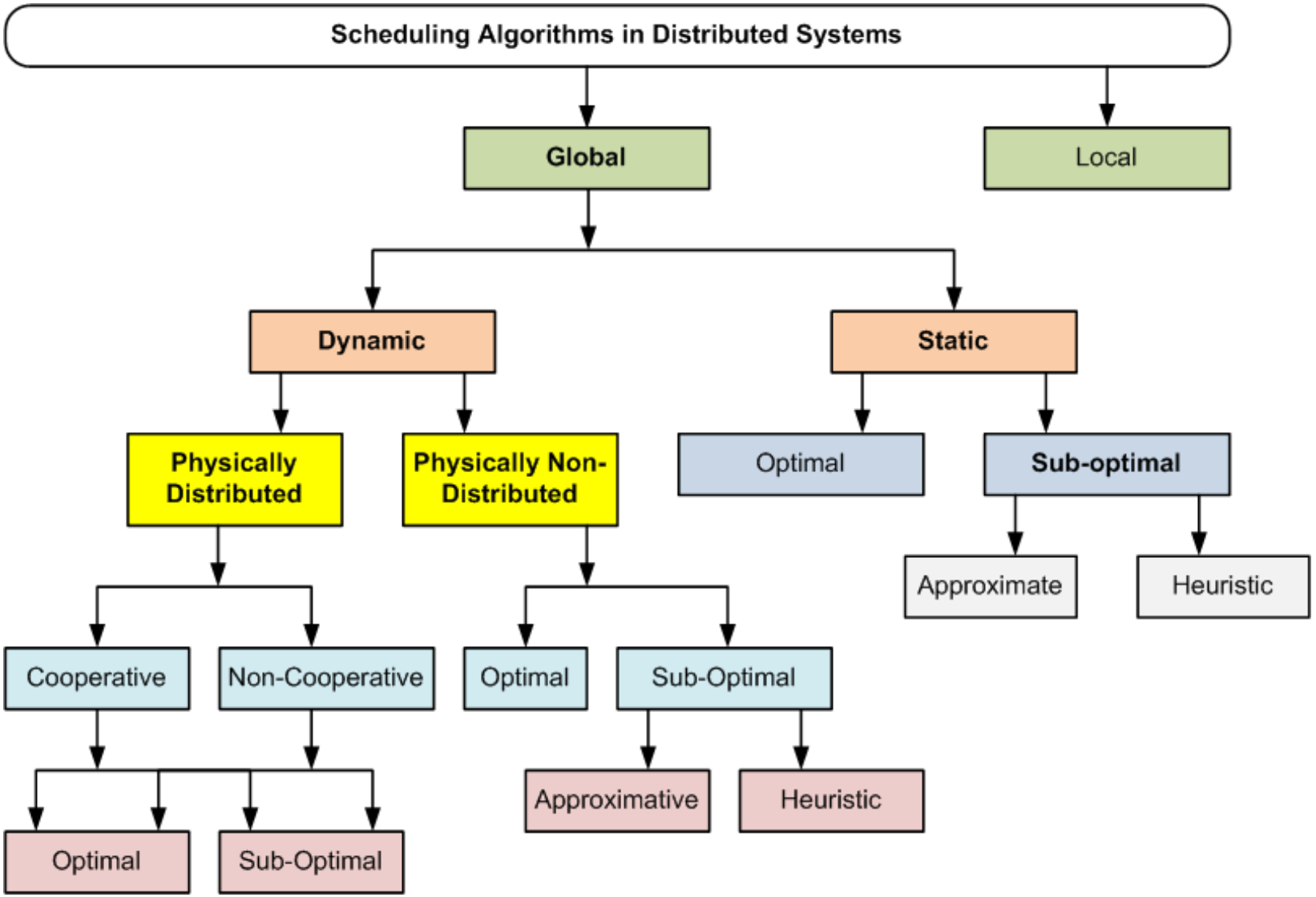


Q: Scheduler's position ? A: ……

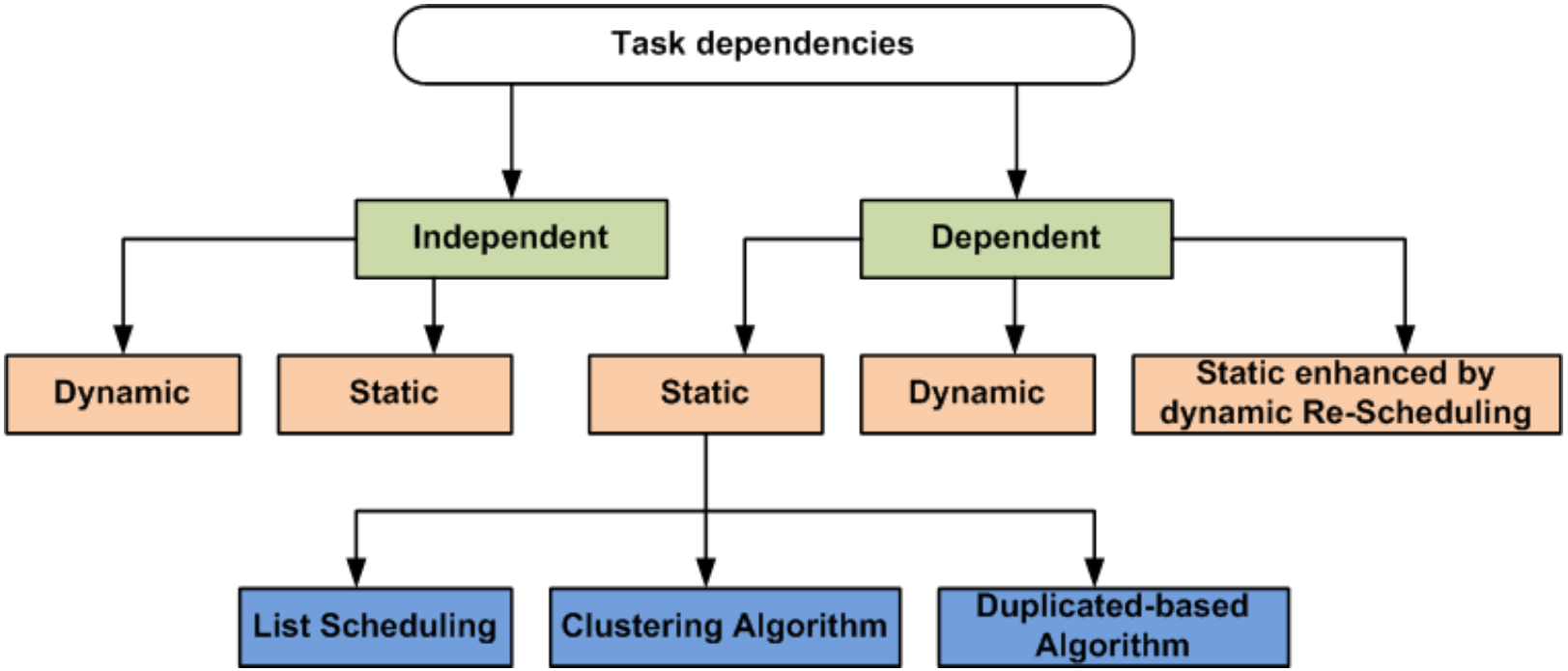
“Ten actions”


Plan for Grid scheduling
PHASE ONE: Resource Discovery
1. Authorization Filtering
2. Application Definition
3. Minimal Requirement Filtering
PHASE TWO: System Selection
4. Information gathering
5. System Selection
PHASE THREE: Job Execution
6. Advance Reservation
7. Job Submission
8. Preparation Task
9. Monitoring Progres
10. Job Completion
11. Clean-up Tasks

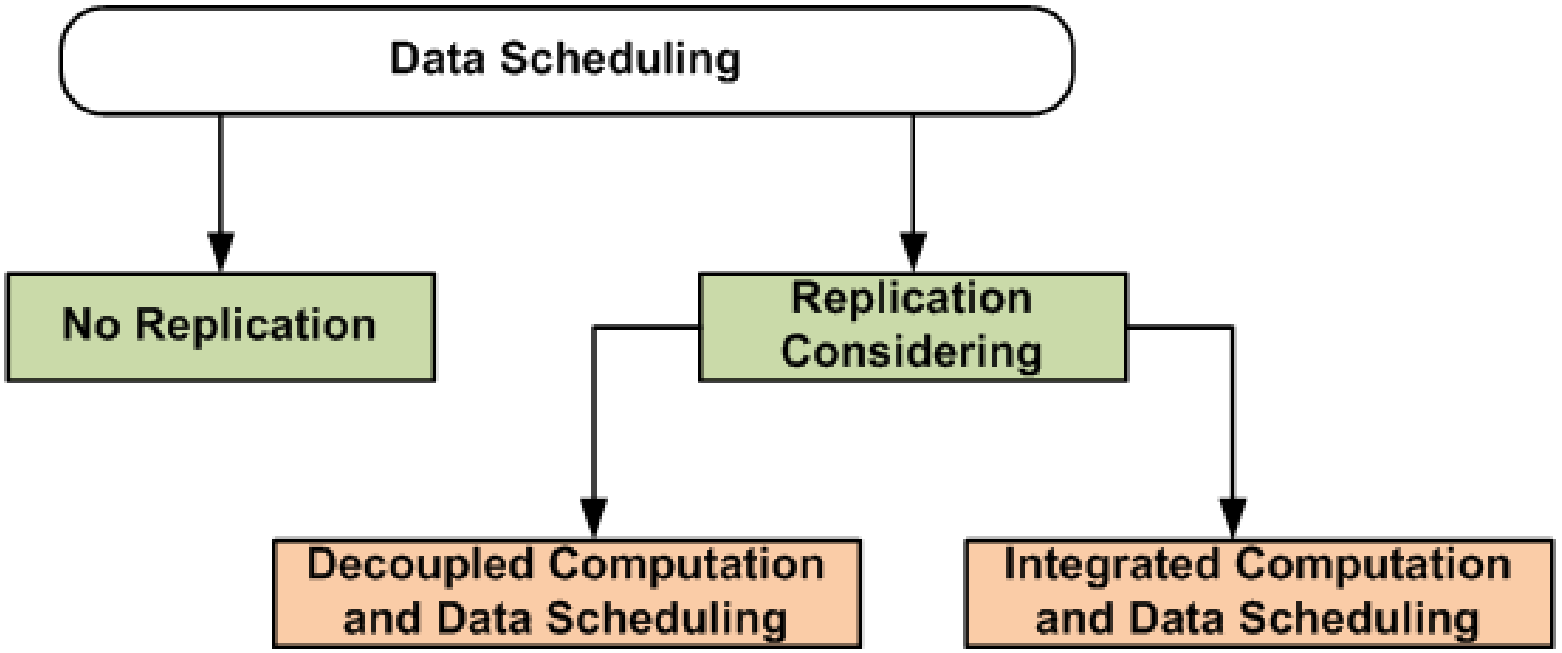
Taxonomy


Scheduling Algorithms in Distributed Systems
Global
Local
Dynamic
Static
Physically Distributed
Physically Non-Distributed
Optimal
Sub-optimal
Approximate
Heuristic
Cooperative
Non-Cooperative
Optimal
Sub-Optimal
Approximative
Heuristic
Optimal
Sub-Optimal

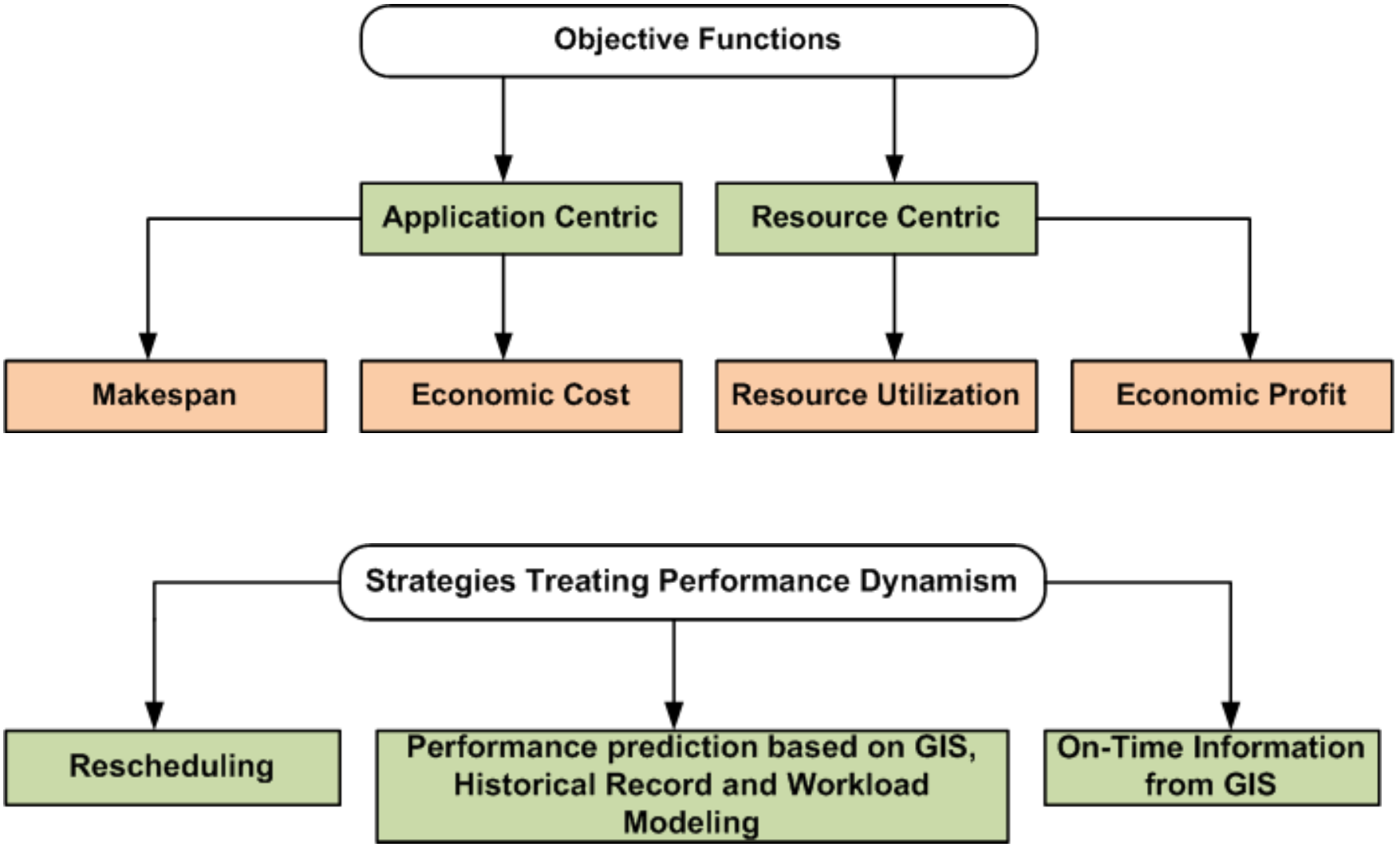
Taxonomy


Task dependencies
Independent
Dependent
Dynamic
Static
Static
Dynamic
Static enhanced by dynamic Re-Scheduling
List Scheduling
Clustering Algorithm
Duplicated-based Algorithm

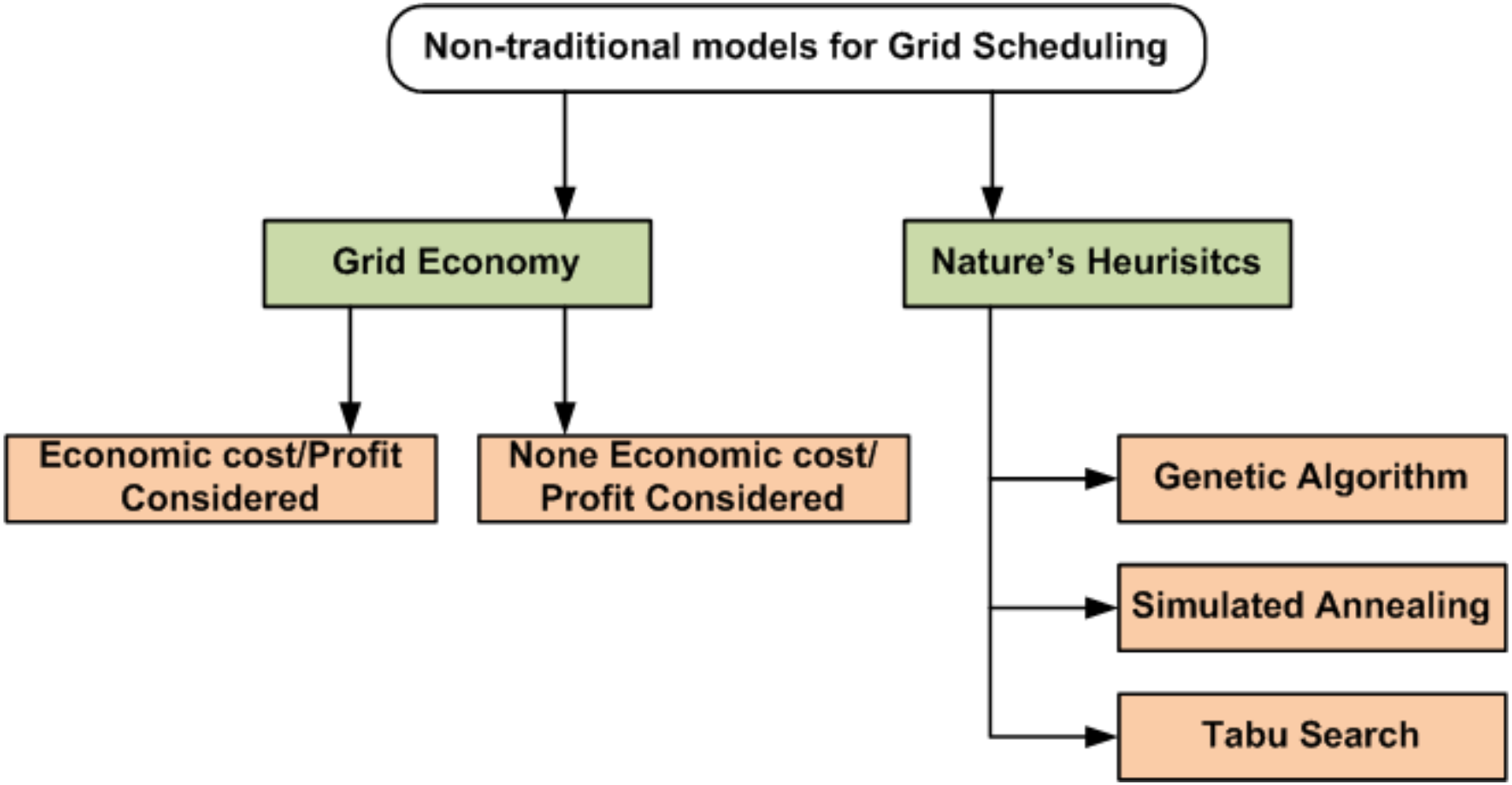
Taxonomy


Data Scheduling
No Replication
Replication Considering
Decoupled Computation and Data Scheduling
Integrated Computation and Data Scheduling

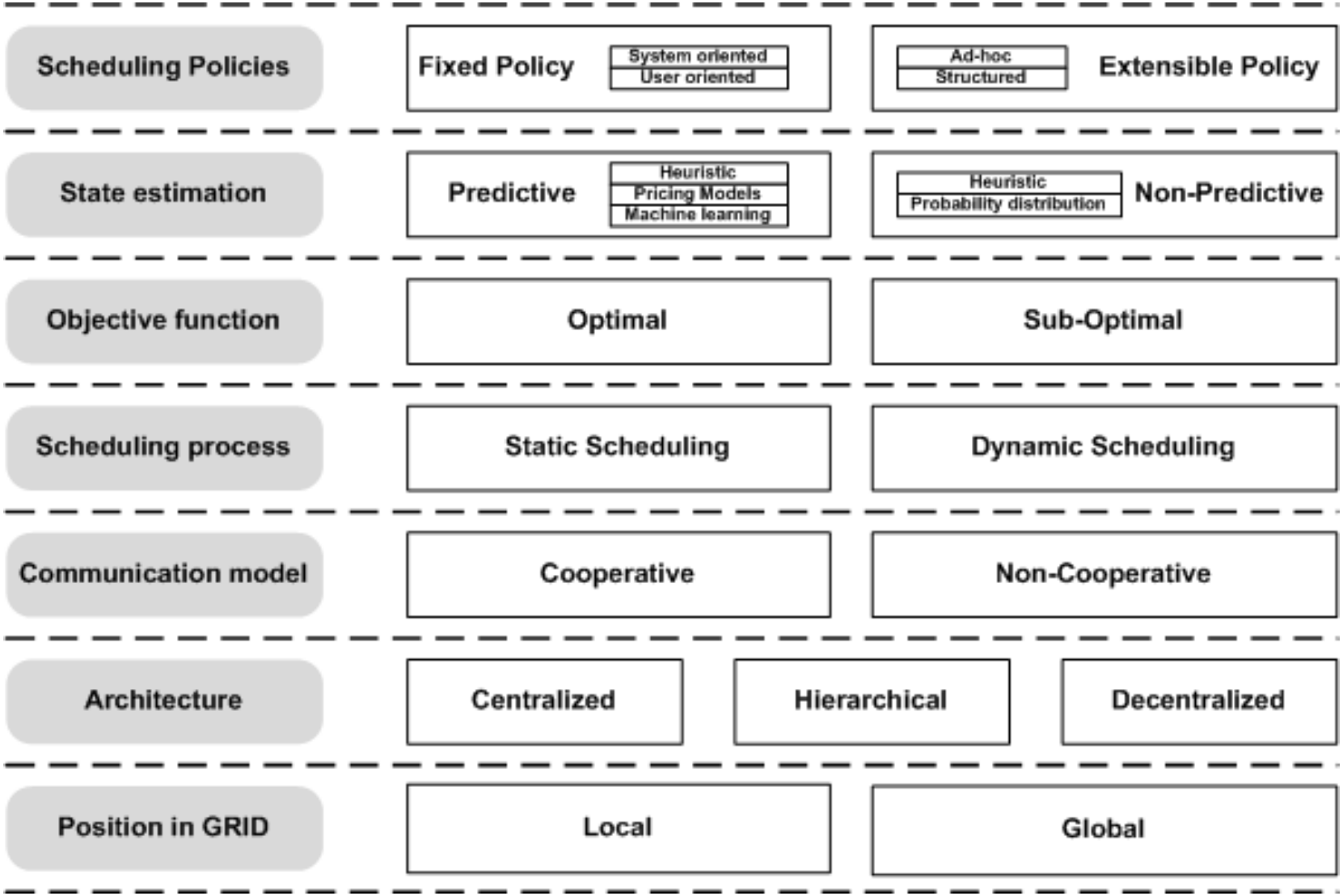
Taxonomy


Objective Functions
Application Centric
Resource Centric
Makespan
Economic Cost
Resource Utilization
Economic Profit
Strategies Treating Performance Dynamism
Rescheduling
Performance prediction based on GIS, Historical Record and Workload Modeling
On-Time Information from GIS

Taxonomy


Non-traditional models for Grid Scheduling
Grid Economy
Nature's Heurisitcs
Economic cost/Profit Considered
None Economic cost/ Profit Considered
Genetic Algorithm
Simulated Annealing
Tabu Search

Scheduling Taxonomy



Scheduling Taxonomy

GRID Communication	Node arbitrary conected	Node fully conected
GRID Resources	Limited number of processors	Unlimited number of processors
Fault tolerance strategy	With duplication	Without duplication
Communication costs	Uniform communication costs	Arbitrary communication costs
Communication model	With communication	Without communication
Graph stucture	Arbitrary graph structure	Restricted graph structure
Tasks dependencies	Depended tasks	Independent tasks
Scheduling model	Centralized Scheduling	Decentralized Scheduling

Scheduling methods

- Push Model
  - Manager pushes jobs from Q to a resource.
  - Used in Clusters, Grids
- Pull Model
  - P2P Agent request for a job for processing from job-pool
  - Commonly used in P2P systems such as SETI@Home
- Hybrid Model (both push and pull)
  - Broker deploys an agent on resources, which pulls jobs from a resource.
  - May use in Grid (e.g., Nimrod-G system).
  - Broker also pulls data from user host or separate data host (distributed datasets) (e.g., Gridbus Broker)

Taxonomy of Systems

- **Computational Grid**:
  - *distributed supercomputing* (parallel application execution on multiple machines)
  - *high throughput* (stream of jobs)
- **Data Grid**: provides the way to solve large scale data management problems
- **Service Grid:** systems that provide services that are not provided by any single local machine.
  - *on demand:* aggregate resources to enable new services
  - *Collaborative:* connect users and applications via a virtual workspace
  - *Multimedia:* infrastructure for real-time multimedia applications

Taxonomy of Resources

- **β classification**: 1, P, Q, MPM, R, J, F, O

- Time-shared vs. Non time-shared
- Dedicated vs. Non-dedicated
- Preemptive vs. Non-preemptive

Resource representation

```
<Node>
<Id>1</Id>
  <FarmName>DIOGENESFarm</FarmName>
  <ClusterName>DIOGENESCluster</ClusterName>
  <NodeName>P01</NodeName>
  <Parameters>
      <CPUPower>2730.8MHZ</CPUPower>
      <Memory>512MB</Memory>
      <CPU_idle>93.7</CPU_idle>
      ...
  </Parameters>
</Node>
```

Taxonomy of Applications

- ***Distributed supercomputing*** consume CPU cycles and memory.
- ***High-Throughput Computing*** unused processor cycles
- ***On-Demand Computing*** meet short-term requirements for resources that cannot be cost-effectively or conveniently located locally.
- ***Data-Intensive Computing***: e.g. satellite image processing
- ***Collaborative Computing*** enabling and enhancing human-to-human interactions (e.g.: CAVE5D system supports remote, collaborative exploration of large geophysical data sets and the models that generated them).

Task description

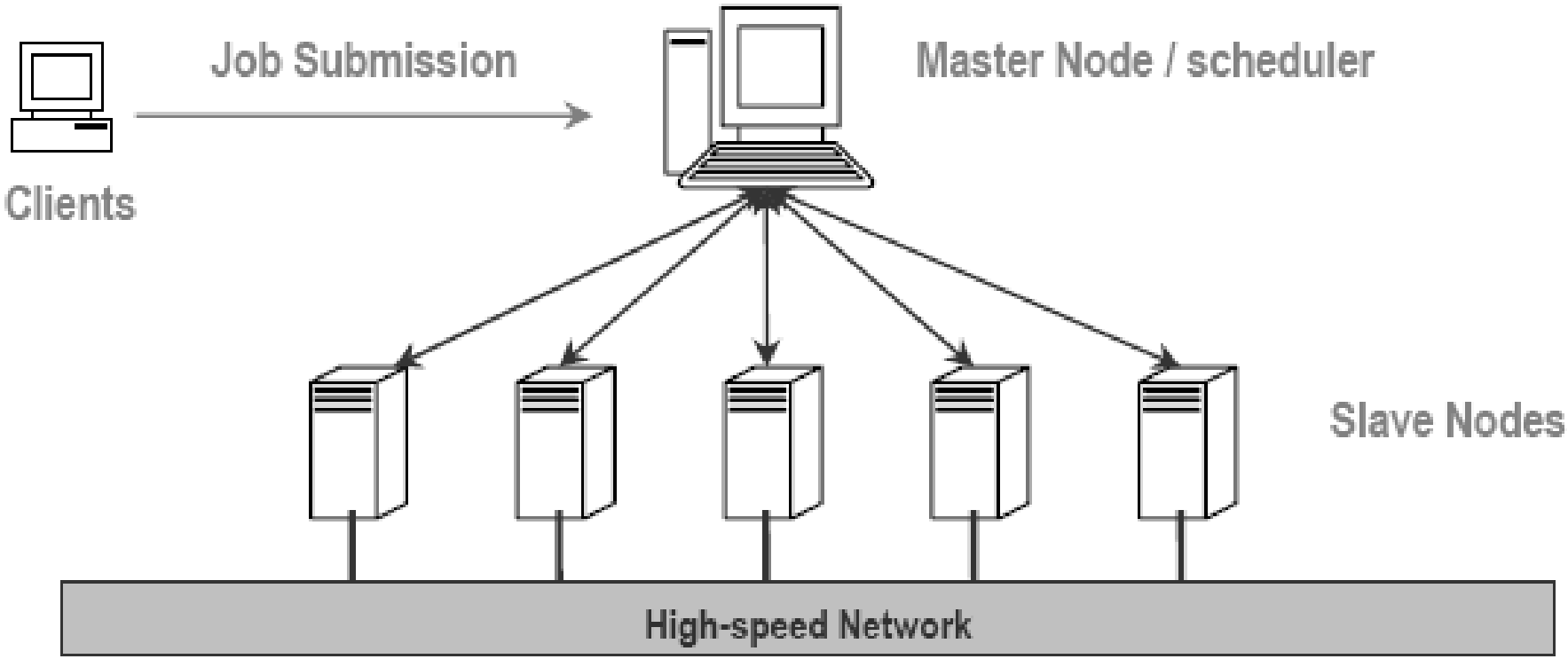
```
<task>
   <taskId>3</taskId>
   <path>/home/student/executabile/loop100.sh</path>
   <arrivingDate>2009/06/04</arrivingDate>
   <arrivingTime>01:20:00</arrivingTime>
   <arguments></arguments>

   <requirements>
        <memory>0.0MB</memory>
         <cpuPower>2745.9404MHZ</cpuPower>
         <processingTime>1</processingTime>
         <deadlineTime>2008/09/10 20:59:30</deadlineTime>
         <schedulePriority>-1</schedulePriority>

         ...
   </requirements>
   <nrexec>1</nrexec>


   <child>
        <Id>5</Id>
        <Cost>10</Cost>
   </child>
</task>
```

Example: Cluster Scheduling



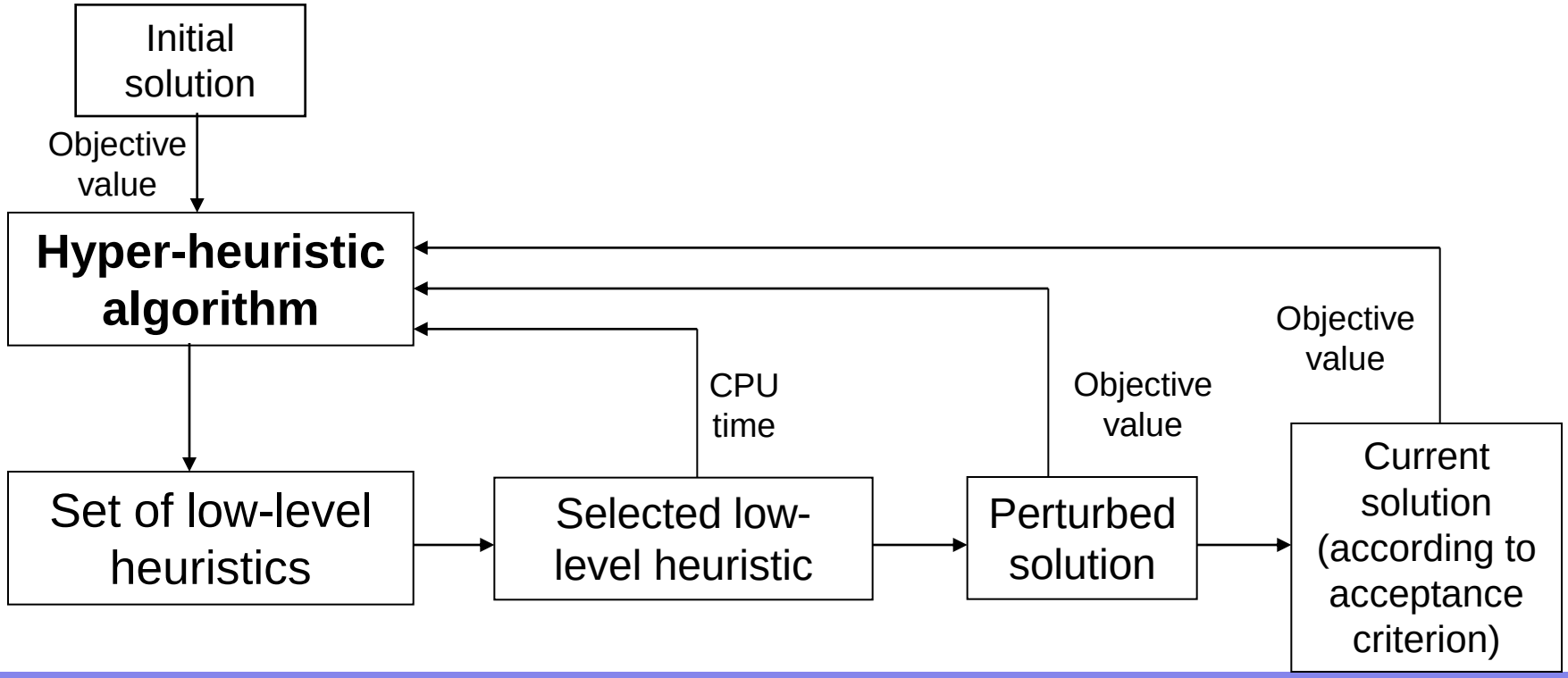
Characteristics of Cluster Scheduling

- **P | r_j; d_j; p_j; pmtn; prec | C_{max}** $(\Sigma\omega\varphi X\varphi)$
- **Q | r_j; d_j; p_j; pmtn; prec | C_{max}** $(\Sigma\omega_j C_j)$

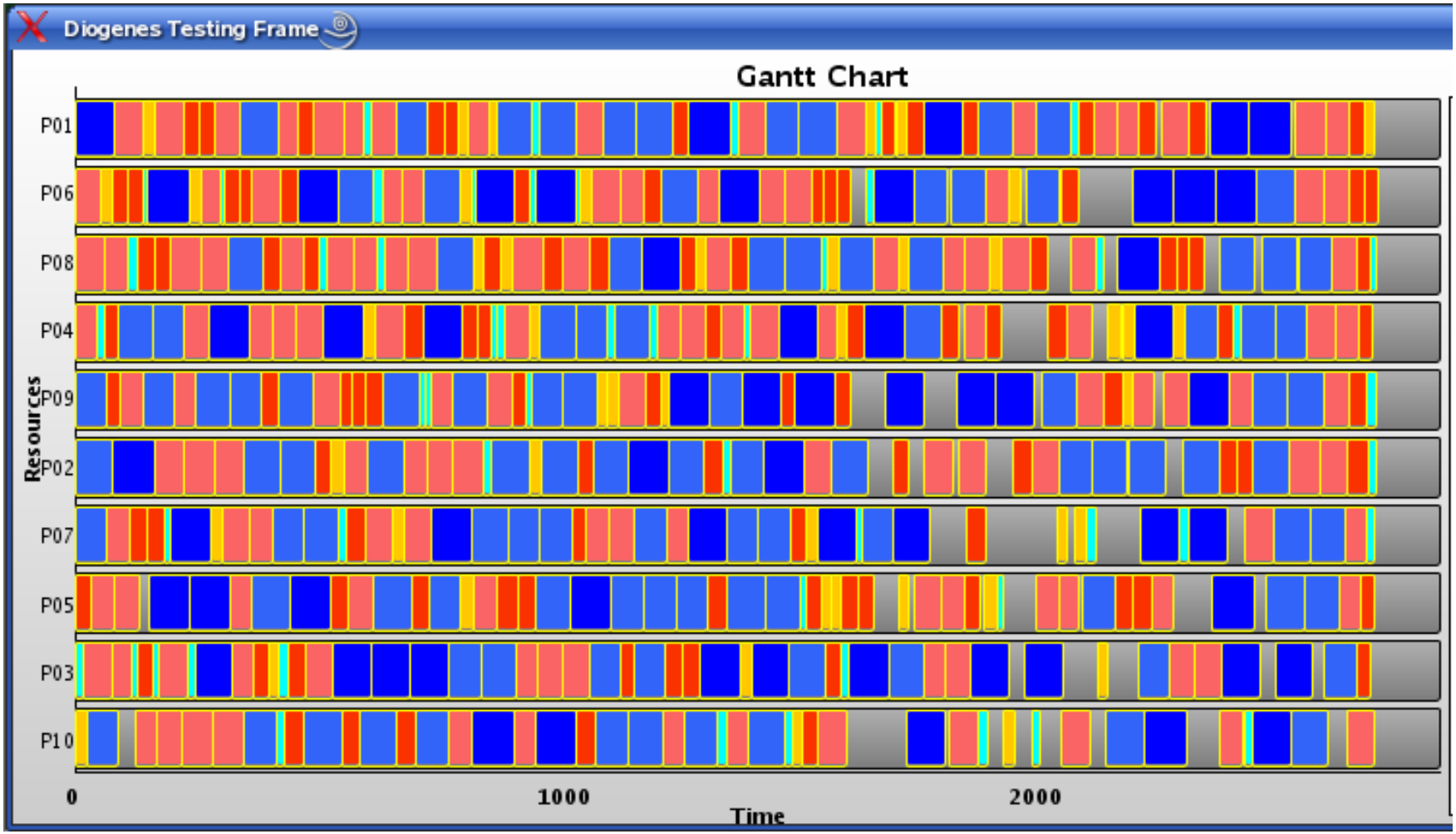
- Homogeneity of resource and application
- Dedicated resource
- Centralized scheduling architecture
- High-speed interconnection network
- Monotonic performance goal

Building a Schedule using a hyper-heuristic

- Advantages of hyperheuristics for clusters
  - Cheap and fast to implement & Produce solutions of good quality
  - Require limited domain-specific knowledge
  - Robustness: can be effectively applied to a wide range of problems and problem instances



Schedule results



Exam's quizzes

- **1.** Care sunt etapele realizării unei planificări într-un sistem distribuit?
- **2.** Descrieţi pe scurt taxonomia algoritmilor de planificare.
- **3.** Care sunt tipurile de sisteme şi resurse specifice unui sistem distribuit?
- **4.** Care sunt tipurile de aplicaţii specifice pentru calului distribuit?
- **5.** Descrieţi pe scurt problema planificării pentru un cluster de calculatoare.